


```

177 font-family:monospace, Courier New !important;
178 font-size:9pt !important;
179 color:#f84;
180 top:-20px;
181 }
182 #GshPerfMon:hover {
183 z-index:3 !important;
184 }
185 #GshSidebar:hover {
186 z-index:2;
187 overflow-x-visible !important;
188 background-color:rgba(255,255,255,0.7);
189 width:50%;
190 }
191 #GshIndexer {
192 z-index:1;
193 position:relative;
194 resize:both !important;
195 height:100%;
196 line-height:1px;
197 top:0px;
198 scroll-behavior: overflow !important;
199 padding-left:4pt;
200 font-size:0.5em;
201 white-space: nowrap;
202 xxx-background-color:rgba(64,160,64,0.6) !important;
203 color:#7794c6;xxx-PBlue;
204 xxbackground-color:#FEE3EB;xxx-PBlue;
205 background-color:#eeeeee;xxx-PBlue;
206 }
207 #GshIndexer:hover {
208 z-index:1000000;
209 overflow-x-visible !important;
210 color:#000000 !important;xxx-PBlue;
211 xxxbackground-color:#FFFFFF;xxx-PBlue;
212 background-color:rgba(255,255,255,0.7);
213 padding-right:0px;
214 width:50%;
215 }
216 #GshIndexer:select {
217 color:#000000 !important;xxx-PBlue;
218 background-color:#FFFFFF;xxx-PBlue;
219 }
220 .IndexLine {
221 font-size:8pt !important;
222 font-family:Georgia;
223 display:inline-block;
224 xxx-color:#fffff;
225 xxx-color:#ef1f1f;xxx-PBlue;
226 xxx-color:#41516d;xxx-PBlue;
227 xxx-color:#7794c6;xxx-PBlue;
228 padding-right:4pt;
229 }
230 .IndexLine:hover {
231 font-size:0pt!important;
232 xxx-color:#228;
233 xxx-background-color:#fff;
234 xxxcolor:#ffff;xxx-PBlue;
235 color:#516487;xxx-PBlue;
236 background-color:rgba(220,220,255,1.0);xxx-PBlue;
237 xxtext-shadow:1px 1px #eee;
238 xxbackground-color:#516487;xxx-PBlue;
239 xxtext-decoration:underline !important;
240 }
241 }
242 
```

```
<script id="Indexer_WorkScript">
```

```
243 function Indexer_openWorkCodeView(){
244     function Indexer_showWorkCode(){
245         showHTMLcode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
246     }
247     Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
248 }
249 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
250 Indexer_openWorkCodeView();
```

```
251 var startPerfDate = new Date();
252 var prevPerfDate = startPerfDate;
253 function ShowResourceUsage(){
254     d = new Date();
255     perf += '<'+'font color="gray">UA:' + window.navigator.userAgent + '</'+'font><br>\n';
256     perf += 'DateShort0(' + startPerfDate) + '<br>\n';
257     perf += 'DateShort1() + '<br>\n';
258     elps = d.getTime() - startPerfDate.getTime();
259     itvl = d.getTime() - prevPerfDate.getTime();
260     perf += 'Elapsed: +' + elps/1000 + ' ms<br>\n';
261     perf += 'Skew: +' + (itvl-1000) + ' ms<br>\n';
262     prevPerfDate = d;
263 }
264 if( performance.memory !== undefined ){
265     if( performance.memory ){
266         mu0 = (m0.usedJSHeapSize / 1000000.0); //toFixed(6);
267         perf += 'Memory: +' + mu0 + ' MB<br>\n';
268     }
269 }
270 perf += '<br>\n';
271 //GshSidebar.innerHTML = perf;
272 GshPerfMon.innerHTML = perf;
273 //GshIndexer.innerHTML = 'Memory: +' + mu0 + ' MB';
274 //console.log('-- Perfmon heap: +' + mu0 + '/' + m0.totalHeapSize + '/' + m0.jsHeapSizeLimit);
275 if( true ){
276     GshSidebar.style.zIndex = 1000;
277     GshIndexer.style.zIndex = 0;
278     GshPerfMon.style.zIndex = 1;
279     //GshSidebar.appendChild(GshPerfMon);
280     if( document.getElementById('primary') == null ){ // not in WordPress
281         GshPerfMon.style.position = 'absolute';
282     }
283     GshPerfMon.style.display = 'block';
284     GshPerfMon.style.marginLeft = '4px';
285     //GshPerfMon.style.top = '45px';
286     GshPerfMon.style.position = 'relative';
287     //top = GshPerfMon.getBoundingClientRect().top;
288     //top = parseInt(top) + 40;
289     //GshPerfMon.style.top = top + 'px';
290     GshPerfMon.style.left = '0px';
291     GshMain.style.top = -GshPerfMon.getBoundingClientRect().height;
292 }
293 }
```

```
300 function ResetPerfMon(){
301     GshPerfMon.removeAttribute('style');
302     GshSidebar.removeAttribute('style');
303 }
```

```
304 var iserno = 0;
305 var GeneratedId = 0;
306 function generateIndex(n,e,chv,nch,ht){
307     // https://developer.mozilla.org/en-US/docs/Web/API/Element
308     c = '';
309     if( e.classList != null ){
310         c = e.classList.value;
311     }
312     //console.log('-- <' + e.nodeName + '> #' + e.id + ' .' + c + ' ' + e.attributes);
313     if( e.nodeName == '#text' ){ return '';}
314     if( e.nodeName == '#comment' ){ return ''}
315     if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
316         id = e.innerHTML;
317         GeneratedId += 1;
318         eid = 'GeneratedId-' + GeneratedId;
319         e.id = eid;
320     }else{
321         if( e.nodeName == 'SUMMARY' ){
322             id = e.innerHTML;
323             GeneratedId += 1;
324             eid = 'GeneratedId-' + GeneratedId;
325             e.id = eid;
326         }else{
327             if( (e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxxentry-content') ){
328                 console.log('-- DIV entry-content begin');
329                 id = e.innerHTML;
330                 GeneratedId += 1;
331                 eid = 'GeneratedId-' + GeneratedId;
332                 e.id = eid;
333                 console.log('-- DIV entry-content end hash-children',e.hasChildNodes());
334             }else{
335                 if( e.id == '' || e.id == 'undefined' ){
336                     return '';
337                 }else{
338                     id = '#' + e.id;
339                     eid = e.id;
340                 }
341             }
342             iserno += 1;
343             ht = '<' + 'div id="GeneratedRef_' + iserno + '" class="IndexLine" href="#">' +
344             '+iserno+' + 'ni+' + e.nodeName + ' : ' + id;
345             if( e.id == '' || e.id == 'undefined' ) {return ht + '</' + 'div>';}
346             if( !e.hasChildNodes() ) {return ht + '</' + 'div>';}
347             chv = e.childNodes;
348             nch = e.childNodes.length;
349             ht += '<' + 'ncn+' + 'ni+' + '>' + '</' + 'div>';
350             for( let i = 0; i < chv.length; i++ ){
351                 sec = ni + '.';
352                 if( ni == '' ){ sec = i; }
353             }
354         }
355     }
356 }
```

```

354     ht += generateIndex(sec, chv[i], null, 0);
355   }
356   return ht;
357 }
358 function onClickIndex(e){
359   tid = e.target.id;
360   tge = document.getElementById(tid);
361   eid = tge.getAttribute('eid');
362   rx = tge.getBoundingClientRect().left.toFixed(0)
363   ry = tge.getBoundingClientRect().top.toFixed(0)
364   if( false ){
365     alert('index clicked mouse(x="'+e.x+', y="'+e.y+')'
366       + '\ntid='+tid+ ' rx=' + rx + ',ry=' + ry
367       + '\neid=' + eid + '\n'
368       + '\nhtml=' + tge.outerHTML );
369   }
370   ee = document.getElementById(eid);
371   sx = 'NaN';
372   sy = 'NaN';
373   ee.style.top = sx+'px'+sy+'px';
374   ee.scrollIntoView();
375   window.scrollTo(sx,sy);
376   //window.scrollTo(left,'Non',top:sy,behavior:'smooth');
377 }
378 function Indexer_afterLoaded(){
379   sideIndex = document.getElementById('GshIndexer');
380   ht = '<'+ 'h3>G-Index</' + 'h3>';
381   ht += generateIndex("",document.getElementById('gsh'),null,0,'');
382   if( (pri = document.getElementById('primary')) != null ){
383     ht += generateIndex("",pri,null,0,'');
384   }
385   ht += '<'+ 'br>';
386   ht += '<'+ 'br>';
387   ht += '<'+ 'br>';
388   ht += '<'+ 'br>';
389   sideIndex.innerHTML = ht;
390   sideIndex.addEventListener('click',onClickIndex);
391 }
392 if( (pri = document.getElementById('primary')) != null ){
393   console.log('-- Seems in WordPress');
394   pri.style.zIndex = 2000;
395 }
396 GshSidebar.style.setProperty('position','relative','important');
397 GshSidebar.style.top = '-1400px';
398 //GshSidebar.style.setProperty('position','absolute','important');
399 //GshSidebar.style.top = '0px';
400
401 GshSidebar.style.setProperty('width','200px','important');
402 GshSidebar.style.setProperty('overflow','scroll','important');
403 GshSidebar.style.resize = 'both';
404
405 GshSidebar.style.left = '-100px';
406 GshIndexer.style.left = '100px';
407 GshIndexer.style.height = '1400px';
408 gsh.appendChild(GshSidebar); // change parent
409 }else{
410   console.log('-- Seems not in Wordpress');
411   GshSidebar.style.setProperty('position','fixed','important');
412 }
413
414 //document.addEventListener('load',Indexer_afterLoaded);
415
416 DestroyIndexBar = function(){
417   sideIndex = document.getElementById('GshIndexer');
418   sideIndex.innerHTML = "";
419   sideIndex.style = "";
420 }
421 </script>
422
423 <!-- Indexer_WorkCodeSpan -->
424 </!-->
425 //<!------- Work j ----->
426
427
428 /*
429 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
430 <p>
431 <note>
432 It is a shell for myself, by myself, of myself. --SatoxITS(^~^)
433 <a href="gsh-0.6.2.go.html">prev</a>
434 <a href="#">next</a>
435 </p>
436 <div id="GjFactory_x"></div>
437
438 <div>
439 <span id="GshMenu" class="GshMenu">
440   <span class="GshMenu1" id="GshMenuEdit" onclick="html_edit();">Edit</span>
441   <span class="GshMenu1" id="GshMenuSave" onclick="html_save();">Save</span>
442   <span class="GshMenu1" id="GshMenuLoad" onclick="html_load();">Load</span>
443   <span class="GshMenu1" id="GshMenuVer" onclick="html_ver0();">Vers</span>
444   <span id="gsh-Wind" onclick="win_jump(0.1)";>0</span>
445   <span class="GshMenu1" id="GshMenuFork" onclick="html_fork();">Pork</span>
446   <span class="GshMenu1" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
447   <span class="GshMenu1" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
448   <span class="GshMenu1" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
449   <span class="GshMenu1" id="GshMenuSign" onclick="html_sign(this); style="">Source</span>
450   <span id="gsh-menu-pure" onclick= html_pure(this);>Pure</span> -->
451 </span>
452 </div>
453 */
454
455 /*
456 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
457 <h3>Fun to create a shell</h3>
458 <p>For a programmer, it must be far easy and fun to create his own simple shell
459 rightly fitting to his favor and necessities, than learning existing shells with
460 complex full features that he never use.
461 as one of programmers, am writing this tiny shell for my own real needs,
462 totally from scratch, with fun.
463 </p><p>
464 For a programmer, it is fun to learn new computer languages. For long years before
465 writing this software, I had been specialized to C and early HTML2 :).
466 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
467 now demand as a novice of these, with fun.
468 </p><p>
469 This single file "gsh.go", that is executable by Go, contains all of the code written
470 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
471 HTML file that works as the viewer of the code of itself, and as the "home page" of
472 this software.
473 </p>
474 Because this HTML file is a Go program, you may run it as a real shell program
475 on your computer.
476 But you must be aware that this program is written under situation like above.
477 Needless to say, there is no warranty for this program in any means.
478 </p>
479 <address>Aug 2020, SatoxITS (satoto@its-more.jp)</address>
480 </details>
481 */
482
483 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
484 </p>
485 <h3>Cross-browser communication</h3>
486 <p>
487 ... to be written ...
488 </p>
489 <h3>Vi compatible command line editor</h3>
490 </p>
491 The command line of GShell can be edited with commands compatible with
492 <a href="https://www.washington.edu/computing/unix/vi.html">vi</a>.
493 As in vi, you can enter <b><code>command mode</code></b></b> key,
494 then move around in the history by <b><code>j k h l f w b 0 $ &</code></b> or so.
495 </p>
496 </details>
497 */
498 <details id="gsh-gindex">
499 <summary>Index</summary><div class="gsh-src">
500 Documents
501 <span class="gsh-link" onclick="jumpToJavaScriptView();">Command summary</span>
502 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
503 Package structures
504   <a href="#import">import</a>
505   <a href="#struct">struct</a>
506 Main functions
507   <a href="#comexpansion">str-expansion</a> // macro processor
508   <a href="#findr">findr</a> // builtin find + du
509   <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
510   <a href="#plusd">plusd</a> // plusd commands
511   <a href="#commands">systemc</a> // external commands
512   <a href="#builtin">builtin</a> // builtin commands
513   <a href="#network">network</a> // socket handler
514   <a href="#remote_sh">remote-sh</a> // remote shell
515   <a href="#redirect">redirect</a> // StdIn/Out/redirection
516   <a href="#history">history</a> // command history
517   <a href="#usage">usage</a> // resource usage
518   <a href="#encode">encode</a> // encode / decode
519   <a href="#IME">IME</a> // command line IME
520   <a href="#getline">getline</a> // line editor
521   <a href="#scanf">scanf</a> // string decomposer
522   <a href="#interpreter">interpreter</a> // command interpreter
523   <a href="#main">main</a>
524 </span>
525 JavaScript part
526   <a href="#script-src-view" class="gsh-link" onclick="jumpToJavaScriptView();">Source</a>
527 
```

```

531     <a href="#gsh-data-frame" class="gsh-link" onclick="jmpTo_DataView();">Builtin data</a>
532     CSS part
533     <a href="#style-src-view" class="gsh-link" onclick="jmpTo_StyleView();">Source</a>
534     References
535     <a href="#" class="gsh-link" onclick="jmpTo_WholeView();">Internal</a>
536     <a href="#" class="gsh-link" onclick="jmpTo_ReferenceView();">External</a>
537     Whole parts
538     <a href="#whole-src-view" class="gsh-link" onclick="jmpTo_WholeView();">Source</a>
539     <a href="#whole-src-view" class="gsh-link" onclick="jmpTo_WholeView();">Download</a>
540     <a href="#whole-src-view" class="gsh-link" onclick="jmpTo_WholeView();">Dump</a>
541 
542   </div>
543   </details>
544 
545 //<details id="gsh-gocode">
546 //<summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
547 // gsh - Go lang based Shell
548 // (c) 2020 ITS more Co., Ltd.
549 // 2020-0807 created by SatoxITS (sato@its-more.jp)
550 package main // gsh main
551 
552 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
553 import (
554     _           // <a href="https://golang.org/pkg/fmt/">fmt</a>
555     "errors"
556     "strings"  // <a href="https://golang.org/pkg/strings/">strings</a>
557     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
558     "sort"    // <a href="https://golang.org/pkg/sort/">sort</a>
559     "time"    // <a href="https://golang.org/pkg/time/">time</a>
560     "bufio"   // <a href="https://golang.org/pkg/bufio/">bufio</a>
561     "io/ioutil" // <a href="https://golang.org/pkg/ioutil/">ioutil</a>
562     "os"       // <a href="https://golang.org/pkg/os/">os</a>
563     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
564     "plugin"  // <a href="https://golang.org/pkg/plugin/">plugin</a>
565     "net"      // <a href="https://golang.org/pkg/net/">net</a>
566     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
567     "html"    // <a href="https://golang.org/pkg/html/">html</a>
568     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
569     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
570     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
571     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
572     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
573     //"gshdata" // gsh's logo and source code
574     "hash/crc32" // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
575     "golang.org/x/net/websocket"
576     "runtime"
577 )
578 
579 
580 */
581 #include <stdio.h> // </stdio.h> to be closed as HTML tag :-p
582 #ifdef WIN32
583 #include <windows.h> // </windows.h>
584 // 2020-1022 added -- terminal mode on Windows
585 // https://docs.microsoft.com/en-us/windows/console/setconsolemode
586 // https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
587 int setTermRaw(){
588     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
589     DWORD tmode = 0;
590     if( GetConsoleMode(hStdin,&tmode) ){
591         DWORD xmode = tmode;
592         xmode |= ~ENABLE_ECHO_INPUT;
593         xmode |= ~ENABLE_LINE_INPUT;
594         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
595         if( SetConsoleMode(hStdin,xmode) ){
596             return tmode;
597         }
598     }
599     return 0;
600 }
601 int setTermMode(int tmode){
602     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
603     SetConsoleMode(hStdin,tmode);
604     return 0;
605 }
606 #else
607 int setTermRaw(){
608     return -1;
609 }
610 int setTermMode(int tmode){
611     return 0;
612 }
613 #endif
614 */
615 import "C"
616 
617 /*
618 // 2020-0906 added,
619 // <a href="https://golang.org/cmd/cgo/">CGo</a>
620 // #include "poll.h" // </poll.h> to be closed as HTML tag :-p
621 // typedef struct { struct pollfd fdv[8]; } pollFd;
622 // int pollx(pollFd *fdv, int nfds, int timeout);
623 // 
624 import "C"
625 
626 // 2020-1021 replaced poll() with channel/select
627 // 2020-0906 added
628 func CpollInn(fp*os.File, timeoutUs int)(ready uintptr){
629     var fdv = C.pollFd{ }
630     var nfds = 1
631     var timeout = timeoutUs/1000
632     fdv.fd[0].fd = C.int(fp.Fd())
633     fdv.events[0] = C.POLLIN
634     if( 0 < EventRecvFd ){
635         fdv.fd[1].fd = C.int(EventRecvFd)
636         fdv.events[1] = C.POLLIN
637         nfds += 1
638     }
639     r := C.polix(&fdv,C.int(nfds),C.int(timeout))
640     if( r <= 0 ){
641         return 0
642     }
643     if( (int(fdv.fd[1].events) & int(C.POLLIN)) != 0 {
644         //fprintf(stderr, "--De- got Event\n");
645         return uintptr(EventFdOffset + fdv.fd[1].fd)
646     }
647     if( (int(fdv.fd[0].events) & int(C.POLLIN)) != 0 {
648         return uintptr(NormalFdOffset + fdv.fd[0].fd)
649     }
650     return 0
651 }
652 */
653 
654 const {
655     NAME = "gsh"
656     VERSION = "0.8.1"
657     DATE = "2020-11-11"
658     AUTHOR = "SatoxITS(^_~)//"
659 }
660 
661 var {
662     GSH_HOME = ".gsh" // under home directory
663     GSH_PORT = 9999
664     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
665     PROMPT = "> "
666     LINESEP = "\r\n" // (8*1024)
667     PATHSEP = ":" // should be ";" in Windows
668     DIRSEP = "\\" // canbe \ in Windows
669     OnWindows = false;
670 }
671 
672 func initGshEnv(){
673     if( runtime.GOOS == "windows" ){
674         PATHSEP = ";";
675         DIRSEP = "\\";
676         OnWindows = true;
677     }else{
678     }
679 }
680 
681 // -X logging control
682 // --A-- all
683 // --I-- info.
684 // --D-- debug
685 // --T-- time and resource usage
686 // --W-- warning
687 // --E-- error
688 // --F-- fatal error
689 // --Xn-- network
690 
691 // <a name="struct">Structures</a>
692 
693 // 2020-1022 Unix/Windows
694 // -----
695 // type astat_t syscall.Stat_t;
696 type astat_t struct { syscall.Stat_t }
697 type astat_t_struct {
698     Size int64
699     Mode os.FileMode
700     Rdev int64
701     Blocks int64
702     Nlink int64
703 }
704 
705 func alstat(path string, astat *aStat_t)(error){
706     /*
707     sstat := syscall.Stat_t{ }
708     */
709 }

```

```

708     err := syscall.Lstat(path,&sstat);
709     /*astat = astat_t{sstat};
710     */
711     fi,err := os.Stat(path);
712     if( err == nil ){
713         astat.Mode = fi.Mode();
714         astat.Size = fi.Size();
715     }
716     return err;
717 }
718 func aFstat(fd int, astat *astat_t)(error){
719     /*
720     astat := syscall.Stat_t{};
721     err := syscall.Fstat(fd,&sstat);
722     *astat = aStat_t(sstat);
723     */
724     err := errors.New("NotImplemented-Fstat");
725     //fmt.Printf("---E--- fstat(%v)(%v)\n",fd,err);
726     return err;
727 }
728 func aAccess(path string, mode uint32)(error){
729     /*
730     /err := syscall.Access(path,mode);
731     //err := errors.New("NotImplemented-Access");
732     fi,err := os.Stat(path);
733     //fmt.Printf("%v Access(%v,%v)\n",path,mode,err);
734     if( err == nil ){
735         fmode := fi.Mode();
736         if fmode.IsRegular() ){
737             perm := fmode.Perm();
738             if (uint32(perm) & mode) != 0 ){
739                 return nil;
740             }
741             return errors.New("NotAccessible");
742         }
743         return errors.New("NotRegularFile");
744     }
745     return err;
746 }
747 // 2020-1022 Unix/Windows
748 // -----
750 type aRusage struct {
751     syscall.Rusage
752     Utime time.Duration
753     Stime time.Duration
754     //Sys interface{}
755 }
756 */
757 const aRUSAGE_SELF = syscall.RUSAGE_SELF
758 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
759
760 const aRUSAGE_SELF = 0
761 const aRUSAGE_CHILDREN = 1
762 func aGetrusage(sel int, ru *aRusage){
763     /*
764     sysru := syscall.Rusage();
765     sysru.Getrusage(sel,&sysru);
766     ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
767     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
768     */
769     func aSetrusage(ru *aRusage, ps *os.ProcessState){
770         ru.Utime = ps.UserTime();
771         ru.Stime = ps.SystemTime();
772     }
773     func showRusage(what string,argv []string, ru *aRusage){
774         fmt.Printf("%s", what);
775         //fmt.Printf(" %s",ru.Utime.Sec,".",ru.Utime.Usec)
776         fmt.Printf(" Sys=%d.%06ds ",ru.Utime/1000000000,(ru.Utime/1000000)%1000000);
777         fmt.Printf(" Sys=%d.%06ds ",ru.Stime/1000000000,(ru.Stime/1000000)%1000000);
778         /*
779         fmt.Println(" Rss=%v",ru.Maxrss)
780         if len(argv)-1>=0{
781             fmt.Printf(" Minflt=%v",ru.Minflt)
782             fmt.Printf(" Majflt=%v",ru.Majflt)
783             fmt.Printf(" IxRSS=%v",ru.Ixrss)
784             fmt.Printf(" IdRSS=%v",ru.Idrss)
785             fmt.Printf(" Nswap=%v",ru.Nswap)
786             fmt.Printf(" Read=%v",ru.Inblock)
787             fmt.Printf(" Write=%v",ru.Oublock)
788         }
789         fmt.Println(" Snd=%v",ru.Msgsnd)
790         fmt.Println(" Rcv=%v",ru.Mgrcv)
791         //if isn("-l",argv){
792         //    fmt.Println(" Sig=%v",ru.Nsignals)
793         //}
794     }
795     fmt.Println("\n");
796 }
797
801 type GCommandHistory struct {
802     StartAt time.Time // command line execution started at
803     EndAt time.Time // command line execution ended at
804     ResCode int // exit code of (external command)
805     CmdError error // error string
806     CmdLine string // full path of the command
807     FoundFile [1]string // output of ufind
808     Rusageev [2]aRusage // Resource consumption, CPU time or so
809     CmdId int // maybe with identified with arguments or impact
810     Workdir string // redirecton commands should not be the Cmdid
811     WorkdirX int // index in ChdirHistory
812     CmdLine string // command line
813 }
814
815 type GChdirHistory struct {
816     Dir string
817     MovedAt time.Time
818     CmdIndex int
819 }
820 type CmdMode struct {
821     BackGround bool
822 }
823 type Event struct {
824     when time.Time
825     event int
826     evarg int64
827     CmdIndex int
828 }
829 var CmdIndex int
830 var Events []Event
831 type PluginInfo struct {
832     Spec *plugin.Plugin
833     Addr plugin.Symbol
834     Name string
835     Path string // this is in Plugin but hidden
836 }
837 type GServer struct {
838     host string
839     port string
840 }
841
842 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
843 const ( // SumType
844     SUM_ITEMS = 0x000001 // items count
845     SUM_SIZE = 0x000000 // data length (simply added)
846     SUM_SIZEHASH = 0x000000 // data length (hashed sequence)
847     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
848     // also envelope attributes like time stamp can be a part of digest
849     // hashed value of sizes or mod-date of files will be useful to detect changes
850
851     SUM_WORDS = 0x0000010 // word count is a kind of digest
852     SUM_LINES = 0x0000020 // line count is a kind of digest
853     SUM_SUM64 = 0x0000040 // simple add of bytes, useful for human too
854
855     SUM_SUM32_BITS = 0x000100 // the number of true bits
856     SUM_SUM32_2BYTE = 0x000000 // 16bits words
857     SUM_SUM32_4BYTE = 0x0000400 // 32bits words
858     SUM_SUM32_8BYTE = 0x000800 // 64bits words
859
860     SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
861     SUM_SUM16_SYS = 0x020000 // UNIXsum -sum -sysv
862     SUM_UNIXFILE = 0x040000
863     SUM_CRCIEEE = 0x080000
864 )
865 type CheckSum struct {
866     Files int64 // the number of files (or data)
867     Size int64 // content size
868     Words int64 // word count
869     Lines int64 // line count
870     SumType int
871     Sum64 uint64
872     Crc32Table crc32.Table
873     Crc32Val uint32
874     Sum int64
875     Ctime time.Time
876     Atime time.Time
877     Mtime time.Time
878     Start time.Time
879     Done time.Time
880     RusgatStart [2]aRusage
881     RusgatEnd [2]aRusage
882 }
883 type ValueStack [][]string
884 type GshContext struct {

```

```

885     StartDir    string // the current directory at the start
886     GetLine    string // gsh-getline command as a input line editor
887     CmdHistory []GChdirHistory // the 1st entry is wd at the start
888     /gshPA    syscall.ProcAttr
889     gshPA    os.ProcAttr
890     CommandHistory []GCommandHistory
891     CmdCurrent GCommandHistory
892     BackGround bool
893     BackGroundable []os.ProcessState; //[]int
894     LastRusage arUsage
895     GshHomeDir string
896     TerminalId int
897     CmdTrace bool // should be [map]
898     CmdRun bool // should be [map]
899     PluginFuncs []pluginInfo
900     iValues []string
901     iDelimiter string // field separator of print out
902     iFormat   string // default print format (of integer)
903     iValStack ValStack
904     ListenerServer ListenerServer
905     RSPV      string // [gsh://]host[:port]
906     RWD       string // remote (target, there) working directory
907     lastCheckSum CheckSum
908   }
909
910   func nsleep(ns time.Duration){
911     time.Sleep(ns)
912   }
913   func usleep(ns time.Duration){
914     nsleep(ns*1000)
915   }
916   func msleep(ns time.Duration){
917     nsleep(ns*100000)
918   }
919   func sleep(ns time.Duration){
920     nsleep(ns*1000000000)
921   }
922
923   func strBegins(str, pat string)(bool){
924     if len(pat) <= len(str){
925       yes := str[0:len(pat)] == pat
926       //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
927       return yes
928     }
929     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
930     return false
931   }
932   func isin(what string, list []string) bool {
933     for v := range list {
934       if v == what {
935         return true
936       }
937     }
938     return false
939   }
940   func isinX(what string,list[]string)(int){
941     for i,v := range list {
942       if v == what {
943         return i
944       }
945     }
946     return -1
947   }
948
949   func env(opts []string) {
950     env := os.Environ()
951     if isin("-s", opts){
952       sort.Slice(env, func(i,j int) bool {
953         return env[i] < env[j]
954       })
955     }
956     for v := range env {
957       fmt.Printf("%v\n",v)
958     }
959   }
960
961 // - rewriting should be context dependent
962 // - should postpone until the real point of evaluation
963 // - should rewrite only known notation of symbols
964 func scanInt(str string)(val int,leng int){
965   leng = -1
966   for i,ch := range str {
967     if '0' <= ch && ch <= '9' {
968       leng = i+1
969     }else{
970       break
971     }
972   }
973   if 0 < leng {
974     ival := strconv.Atoi(str[0:leng])
975     return ival,leng
976   }else{
977     return 0,0
978   }
979 }
980 func subHistroy(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
981   if len(str[i+1:]) == 0 {
982     return 0,rstr
983   }
984   hi := 0
985   histlen := len(gshCtx.CommandHistory)
986   if str[i+1] == '!' {
987     hi = histlen - 1
988     leng = 1
989   }else{
990     hi,leng = scanInt(str[i+1:])
991     if leng == 0 {
992       return 0,rstr
993     }
994     if hi < 0{
995       hi = histlen + hi
996     }
997   }
998   if 0 <= hi && hi < histlen {
999     var ext byte
1000    if 1 < len(str[i+leng:]) {
1001      ext = str[i+leng:][1]
1002    }
1003    //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
1004    if ext == ' ' {
1005      leng += 1
1006      xlist := []string()
1007      list := gshCtx.CommandHistory[hi].FoundFile
1008      for ,v := range list {
1009        if !list[i].escapeWhiteSP(v) {
1010          xlist = append(xlist,escapeWhiteSP(v))
1011        }
1012        //rstr += strings.Join(list, " ")
1013        rstr += strings.Join(xlist, " ")
1014      }else{
1015        if ext == '!' || ext == 'd' {
1016          //IN... workdir at the start of the command
1017          leng += 1
1018          rstr += gshCtx.CommandHistory[hi].WorkDir
1019        }else{
1020          rstr += gshCtx.CommandHistory[hi].CmdLine
1021        }
1022      }else{
1023        leng = 0
1024      }
1025      return leng,rstr
1026    }
1027    func escapeWhiteSP(str string)(string){
1028      if len(str) == 0 {
1029        return "\z" // empty, to be ignored
1030      }
1031      rstr := ""
1032      for ,ch := range str {
1033        switch ch {
1034        case '\': rstr += "\\\\""
1035        case ' ': rstr += "\\s"
1036        case '\t': rstr += "\\t"
1037        case '\r': rstr += "\\r"
1038        case '\n': rstr += "\\n"
1039        default: rstr += string(ch)
1040      }
1041    }
1042    return rstr
1043  }
1044  func unescapeWhiteSP(str string)(string){ // strip original escapes
1045    rstr := ""
1046    for i:= 0; i < len(str); i++ {
1047      ch := str[i]
1048      if ch == '\\' {
1049        if i+1 < len(str) {
1050          switch str[i+1] {
1051            case 'z':
1052              continue;
1053            }
1054        }
1055        rstr += string(ch)
1056      }
1057    }
1058    return rstr
1059  }
1060  func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
1061    usrv := []string{}

```

```

1062     for _,v := range strv {
1063         ustrv = append(ustrv,unescapeWhiteSP(v))
1064     }
1065     return ustrv
1066 }
1067 // <a name="cosexpansion">str-expansion</a>
1068 // - this should be a macro processor
1069 func strSubst(gshCtx *GshContext,str string,histonly bool) string {
1070     rbuf := []byte{}
1071     if false {
1072         //@# Unicode should be cared as a character
1073         return str
1074     }
1075     //rstr := "" // escape character mode
1076     inEsc := 0 // escape character mode
1077     for i := 0; i < len(str); i++ {
1078         //fmt.Printf("-D--Subst %v:%v\n",i,str[i:])
1079         ch := str[i]
1080         if ch == '\0' {
1081             if ch == '\'' {
1082                 if len(xrstr) == 0 {
1083                     xrstr := substHistory(gshCtx,str,i,rstr)
1084                     len,rs := substHistory(gshCtx,str,i,"")
1085                     if 0 < len {
1086                         if _,rs := substHistory(gshCtx,str,i,"") {
1087                             rbuf = append(rbuf,[byte(rs)])
1088                             i += len
1089                         } //rstr = xrstr
1090                         continue
1091                     }
1092                 }
1093                 switch ch {
1094                     case '\\': inEsc = '\\'; continue
1095                     case '$': inEsc = '$'; continue
1096                     case '$':
1097                 }
1098             switch inEsc {
1099                 case '\\':
1100                     switch ch {
1101                         case '\\': ch = '\\'
1102                         case 'g': ch = ' '
1103                         case 't': ch = '\t'
1104                         case 'r': ch = '\r'
1105                         case 'n': ch = '\n'
1106                         case 'z': inEsc = 0; continue // empty, to be ignored
1107                     }
1108                     inEsc = 0
1109                 case '$':
1110                     switch {
1111                         case ch == '%': ch = '%'
1112                         case ch == 'T':
1113                             //rstr = rstr + time.Now().Format(time.Timestamp)
1114                             rs := time.Now().Format(time.Timestamp)
1115                             rbuf = append(rbuf,[byte(rs)])
1116                             inEsc = 0
1117                             continue;
1118                         default:
1119                             //rstr = rstr + " " + string(ch)
1120                             rbuf = append(rbuf,ch)
1121                             inEsc = 0
1122                             continue;
1123                         }
1124                     }
1125                     inEsc = 0
1126                 }
1127             //rstr = rstr + string(ch)
1128             rbuf = append(rbuf,ch)
1129         }
1130     }
1131     //fmt.Printf("-D--subst(%s)(%s)\n",str,string(rbuf))
1132     return string(rbuf)
1133 }
1134 }
1135 func showFileInfo(path string, opts []string) {
1136     if isn("-l",opts) || isn("-ls",opts) {
1137         fi, err := os.Stat(path)
1138         if err != nil {
1139             fmt.Printf("----- ((%v))\n",err)
1140         } else{
1141             mod := fi.ModTime()
1142             date := mod.Format(time.Timestamp)
1143             fmt.Printf("%v %v %s ",fi.Mode(),fi.Size(),date)
1144         }
1145     }
1146     fmt.Printf("%s",path)
1147     if isn("-sp",opts) {
1148         fmt.Printf(" ")
1149     } else{
1150         if isn("-n",opts) {
1151             fmt.Printf("\n")
1152         }
1153     }
1154 }
1155 func userHomeDir()(string,bool){
1156     /*
1157     homedir,_ = os.UserHomeDir() // not implemented in older Golang
1158     */
1159     homedir,found := os.LookupEnv("HOME")
1160     //fmt.Printf("-I-- HOME=%v(%v)\n",homedir,found)
1161     if !found {
1162         return "/tmp",found
1163     }
1164 }
1165 func toFullpath(path string) (fullpath string) {
1166     if path[0] == '/' {
1167         return path
1168     }
1169     pathv := strings.Split(path,DIRSEP)
1170     switch {
1171     case pathv[0] == ".": // all ones should be interpreted
1172         pathv[0],_ = os.Getwd()
1173     case pathv[0] == "..": // all ones should be interpreted
1174         cwd, _ := os.Getwd()
1175         cwdv := strings.Split(cwd,DIRSEP)
1176         pathv[0] = strings.Join(ppathv,DIRSEP)
1177     case pathv[0] == "-":
1178         pathv[0],_ = userHomeDir()
1179     default:
1180         cwd, _ := os.Getwd()
1181         pathv[0] = cwd + DIRSEP + pathv[0]
1182     }
1183     return strings.Join(pathv,DIRSEP)
1184 }
1185 }
1186 func IsRegFile(path string)(bool){
1187     fi,err := os.Stat(path)
1188     if err == nil {
1189         fm := fi.Mode()
1190         return fm.IsRegular()
1191     }
1192     return false
1193 }
1194 }
1195 // <a name="encode">Encode / Decode</a>
1196 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1197 func (gshCtx *GshContext)Enc(argv[]string){
1198     file := os.Stdin
1199     buff := make([]byte,LINESIZE)
1200     li := 0
1201     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1202     for li = 0; ; li++ {
1203         count, err := file.Read(buff)
1204         if count <= 0 {
1205             break
1206         }
1207         if err != nil {
1208             break
1209         }
1210         encoder.Write(buff[0:count])
1211     }
1212     encoder.Close()
1213 }
1214 func (gshCtx *GshContext)Dec(argv[]string){
1215     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1216     li := 0
1217     buff := make([]byte,LINESIZE)
1218     for li = 0; ; li++ {
1219         count, err := decoder.Read(buff)
1220         if count <= 0 {
1221             break
1222         }
1223         if err != nil {
1224             break
1225         }
1226         os.Stdout.Write(buff[0:count])
1227     }
1228 }
1229 // lns[N] [-crlf][-\C \ ]
1230 func (gshCtx *GshContext)SplitLine(argv[]string){
1231     strRep := isn("-stt",argv) //...+
1232     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1233     ni := 0
1234     toi := 0
1235     for ni = 0; ; ni++ {
1236         line,err := reader.ReadString('\n')
1237         if len(line) <= 0 {

```

```

1239     if err != nil {
1240         fmt.Fprintf(os.Stderr,"--I-- lnsP $d to $d (%v)\n",ni,toi,err)
1241         break
1242     }
1243 }
1244 off := 0
1245 ilen := len(line)
1246 remlen := len(line)
1247 i := strRep { os.Stdout.Write([]byte("\n")) }
1248 for oin := 0; 0 < remlen; oin++ {
1249     olen := remlen
1250     addnl := false
1251     if 72 < olen {
1252         olen = 72
1253         addnl = true
1254     }
1255     fmt.Fprintf(os.Stderr,"--D-- write $d (%d.%d) $d %d/%d/%d\n",
1256                 toi,ni,oi,off,olen,remlen,ilen)
1257     toin := byte(0)
1258     os.Stdout.Write([]byte(line[0:olen]))
1259     if addnl {
1260         if strRep {
1261             os.Stdout.Write([]byte("\n+\n"))
1262         }else{
1263             //os.Stdout.Write([]byte("\r\n"))
1264             os.Stdout.Write([]byte("\r"))
1265             os.Stdout.Write([]byte("\n"))
1266         }
1267     }
1268     line = line[olen:]
1269     off += olen
1270     remlen -= olen
1271 }
1272 if strRep { os.Stdout.Write([]byte("\n")) }
1273 }
1274 fmt.Fprintf(os.Stderr,"--I-- lnsP $d to $d(%n),ni,toi)
1275 }
1276
1277 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
1278 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1279 var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
1280 var CRC32IEEE uint32 = uint32(0xEDB88320)
1281 func byteCRC32add(crc uint32, str[]byte, len uint64)(uint32){
1282     var oin uint64
1283     for oin = 0; oin < len; oin++ {
1284         var oct = str[oin]
1285         for bi := 0; bi < 8; bi++ {
1286             //fmt.Printf("sherr,-~CRC32 %d %X (%d.%d)\n",crc,oct,oin,bi)
1287             ovf1 := (crc & 0x80000000) != 0
1288             ovf2 := (oct & 0x80) != 0
1289             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1290             oct <<= 1
1291             crc <<= 1
1292             if ovf { crc ^= CRC32UNIX }
1293         }
1294     }
1295     //fprintf(stderr,"--CRC32 return %d %d\n",crc,len)
1296     return crc;
1297 }
1298
1299 func byteCRC32end(crc uint32, len uint64)(uint32){
1299     var slen = make([]byte,4)
1300     var li = 0
1301     for li = 0; li < 4; {
1302         slen[li] = byte(len)
1303         li += 1
1304         len >>= 8
1305         if( len == 0 ) {
1306             break
1307         }
1308     }
1309     crc = byteCRC32add(crc,slen,uint64(li))
1310     crc ^= 0xFFFFFFFF
1311     return crc
1312 }
1313 func strCRC32(str string, len uint64)(crc uint32){
1314     crc = byteCRC32add(0,[]byte(str),len)
1315     crc = byteCRC32end(crc,len)
1316     //fmt.Printf(stderr,"--CRC32 %d %d\n",crc,len)
1317     return crc
1318 }
1319 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
1320     var slen = make([]byte,4)
1321     var li = 0
1322     for li = 0; li < 4; {
1323         slen[li] = byte(len & 0xFF)
1324         li += 1
1325         len >>= 8
1326         if( len == 0 ) {
1327             break
1328         }
1329     }
1330     crc = crc32.Update(crc,table,slen)
1331     crc = 0xFFFFFFFF
1332     return crc
1333 }
1334
1335 func (gsh*GshContext)xChecksum(path string,argv[]string, sum*CheckSum)(int64{
1336     if isn("-type/f",argv) && !IsRegFile(path){
1337         return 0
1338     }
1339     if isn("-type/d",argv) && IsRegFile(path){
1340         return 0
1341     }
1342     file, err := os.OpenFile(path,os.O_RDONLY,0)
1343     if err != nil {
1344         fmt.Printf("--E-- cksum %v (%v)\n",path,err)
1345         return -1
1346     }
1347     defer file.Close()
1348     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n",path,argv) }
1349
1350     bi := 0
1351     var buff = make([]byte,32*1024)
1352     var total int64 = 0
1353     var initTime = time.Time{}
1354     if sum.Start == initTime{
1355         sum.Start = time.Now()
1356     }
1357     for bi = 0; ; bi++ {
1358         count,err := file.Read(buff)
1359         if count <= 0 || err!= nil {
1360             break
1361         }
1362         if (sum.SumType & SUM_SUM64) != 0 {
1363             s := sum.Sum64
1364             for c := range buff[0:count] {
1365                 s += uint64(c)
1366             }
1367             sum.Sum64 = s
1368         }
1369         if (sum.SumType & SUM_UNIXFILE) != 0 {
1370             sum.Crc32Val = byteCRC32add(sum.Crc32Val,buff,uint64(count))
1371         }
1372         if (sum.SumType & SUM_CRCIEEE) != 0 {
1373             sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
1374         }
1375         // <a href="https://en.wikipedia.org/wiki/BSB\_checksum">BSB checksum</a>
1376         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1377             s := sum.Sum16
1378             for c := range buff[0:count] {
1379                 s = (s >> 1) + ((s & 1) << 15)
1380                 s += int(c)
1381                 s &= 0xFFFF
1382             }
1383             //fmt.Printf("BSDsum: %d(%d) %d\n",sum.Size+int64(i),i,s)
1384             sum.Sum16 = s
1385         }
1386         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1387             for bj := 0; bj < count; bj++ {
1388                 sum.Sum16 += int(buff[bj])
1389             }
1390         }
1391         total += int64(count)
1392     }
1393     sum.Done = time.Now()
1394     sum.File += 1
1395     sum.Size += total
1396     if isn("-s",argv) {
1397         fmt.Printf("%v ",total)
1398     }
1399 }
1400
1401
1402 // <a name="grep">grep</a>
1403 // "lines", "lin" or "inp" for "text" line processor" or "scanner"
1404 // "a", "ab", "c", ... sequential combination of patterns
1405 // "LINE" .. is should be definable
1406 // generic line-by-line processing
1407 // grep [-v]
1408 // cat -n -v
1409 // uniq [-c]
1410 // tail [-c]
1411 // sed s/x/y/ or awk
1412 // grep with line count like wc
1413 // rewrite contents if specified
1414 func (gsh*GshContext)xGrep(path string,rexpv[]string)(int{
1415     file, err := os.OpenFile(path,os.O_RDONLY,0)

```

```

1416     if err != nil {
1417         fmt.Printf("--E-- grep %v\n", path,err)
1418         return -1
1419     }
1420     defer file.Close()
1421     if gsh.CmdTrace { fmt.Printf("--I-- grep %v\n",path,rexpv) }
1422     //reader := bufio.NewReaderSize(file,LINESIZE)
1423     reader := bufio.NewReaderSize(file,80)
1424     li := 0
1425     found := 0
1426     for li = 0; li++ {
1427         line, err := reader.ReadString('\n')
1428         if len(line) < 0 {
1429             break
1430         }
1431         if l50 < len(line) {
1432             // maybe binary
1433             break
1434         }
1435         if err != nil {
1436             break
1437         }
1438         if 0 < strings.Index(string(line),rexpv[0]) {
1439             found += 1
1440             fmt.Printf("%s:%d: %s",path,li,line)
1441         }
1442     }
1443     //fmt.Printf("total %d lines %s\n",li,path)
1444     //if 0 < found { fmt.Printf("(found %d lines %s)\n",found,path); }
1445     return found
1446 }
1447 // <a name="finder">Finder</a>
1448 // finding files with its name and contents
1449 // file names are ORed
1450 // show the content with %x fmt list
1451 // -R
1452 // tar command by adding output
1453 type fileSum struct {
1454     Err int64 // access error or so
1455     Size int64 // content size
1456     DupSize int64 // content size from hard links
1457     Blocks int64 // number of blocks (of 512 bytes)
1458     DupBlocks int64 // Blocks pointed from hard links
1459     HLinks int64 // hard links
1460     Words int64
1461     Lines int64
1462     Flats int64
1463     Dirs int64 // the num. of directories
1464     SymLink int64
1465     Plats int64 // the num. of flat files
1466     MaxDepth int64
1467     MaxNameLen int64 // max. name length
1468     nextRepo time.Time
1469 }
1470 func showUsage(dir string,fusage *fileSum){
1471     bsume := float64((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1472     //bsumup := float64((fusage.Blocks/2)*1024)/1000000.0
1473     //fusage.Total := bsume
1474     fmt.Printf("%v %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1475         dir,
1476         fusage.Files,
1477         fusage.Dirs,
1478         fusage.SymLink,
1479         fusage.HLinks,
1480         float64(fusage.Size)/1000000.0,bsume);
1481 }
1482 const {
1483     S_IFMT   = 0170000
1484     S_IFCHR  = 0060000
1485     S_IFDIR  = 0040000
1486     S_IFREG  = 0100000
1487     S_IFLNK  = 0120000
1488     S_IFSOCK = 0140000
1489 }
1490 func cumInfo(fsum *fileSum, path string, staterr error, fstat aStat_t, argv[]string,verb bool)(*fileSum){
1491     now := time.Now()
1492     if time.Second < now.Sub(fsum.nextRepo) {
1493         if !time.Second {
1494             if !fsum.nextRepo.IsZero(){
1495                 tstamp := now.Format(time.Stamp)
1496                 showUsage(tstamp,fsum)
1497             }
1498             fsum.nextRepo = now.Add(time.Second)
1499         }
1500         if staterr != nil {
1501             fsum.Err += 1
1502             return fsum
1503         }
1504         fsum.Files += 1
1505         if 1 < fstat.Nlink {
1506             // must count only once...
1507             // at least ignores files in the same directory
1508             //if (fstat.Mode & os.ModeRegular) {
1509             if (fstat.Mode & S_IFMT) == S_IFREG {
1510                 fsum.HLinks += 1
1511                 fsum.DupBlocks += int64(fstat.Blocks)
1512                 //fmt.Printf("--Dup HardLink %s\n",fstat.Nlink,path)
1513             }
1514             //fsum.Size += finfo.Size()
1515             fsum.Size += fstat.Size
1516             fsum.Blocks += int64(fstat.Blocks)
1517             //if verb { fmt.Printf("(#dblk) %s",fstat.Blocks/2,path) }
1518             if isn("-ls",argv) {
1519                 if verb {
1520                     fmt.Printf("%d %d ",fstat.Blksize,fstat.Blocks)
1521                 }
1522             }
1523             //if finfo.IsDir()
1524             if (fstat.Mode & S_IFDIR) {
1525                 fsum.Dirs += 1
1526             }
1527             //if (finfo.Mode() & os.ModeSymlink) != 0
1528             if (fstat.Mode & S_IFMT) == S_IFLINK {
1529                 //if verb { fmt.Println("symlink(%v,%s)\n",fstat.Mode,finfo.Name()) }
1530                 //fmt.Println("symlink(%v,%s)\n",fstat.Mode,finfo.Name())
1531                 fsum.Symlink += 1
1532             }
1533         }
1534     }
1535     func (gsh*GshContext)xxFindEnv(depth int,total *fileSum,dir string, dstat aStat_t, ei int, entv []string,npatv[]string,argv[]string)(*fileSum){
1536         nols := isn("-grep",argv)
1537         // sort entv
1538         if isn("-t",argv){
1539             sort.Slice(filev, func(i,j int) bool {
1540                 return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1541             })
1542         }
1543     }
1544     /*
1545     if isn("-u",argv){
1546         sort.Slice(filev, func(i,j int) bool {
1547             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1548         })
1549     }
1550     if isn("-U",argv){
1551         sort.Slice(filev, func(i,j int) bool {
1552             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1553         })
1554     */
1555     /*
1556     if isn("-S",argv){
1557         sort.Slice(filev, func(i,j int) bool {
1558             return filev[j].Size() < filev[i].Size()
1559         })
1560     */
1561     for ,filename := range entv {
1562         for ,npat := range npatv {
1563             match := true
1564             if npat == "*" {
1565                 match = true
1566             }else{
1567                 match, _ = filepath.Match(npat,filename)
1568             }
1569             if match {
1570                 path := dir + DIRSEP + filename
1571                 continue
1572             }
1573             Path := dir + DIRSEP + filename
1574             if !match {
1575                 var fstat aStat_t
1576                 staterr := alstat(path,&fstat)
1577                 if staterr != nil {
1578                     if isn("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
1579                     continue;
1580                 }
1581             }
1582             if isn("-du",argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1583                 // should not show size of directory in "-du" mode ...
1584             }else{
1585                 if isn("-s",argv) && (!isn("-du",argv) || isn("-a",argv)) {
1586                     if isn("-du",argv) {
1587                         fmt.Printf("%d(%d",fstat.Blocks/2
1588                     }
1589                     showFileInfo(path,argv)
1590                 }
1591             }
1592             if true { // && isn("-du",argv)
1593                 total = cumInfo(total,path,staterr,fstat,argv,false)
1594             }
1595         }
1596     }
1597 }

```

```

1593     */
1594     if (isin("-wc", argv)) {
1595     }
1596     /*
1597     if gsh.lastCheckSum.SumType != 0 {
1598         gsh.xCksum(path, argv, &gsh.lastCheckSum);
1599     }
1600     */
1601     x := isin("-grep", argv); // -grep will be convenient like -ls
1602     if 0 <= x && x+1 < len(argv) { // -grep will be convenient like -ls
1603         if IsRegFile(path) {
1604             found := gsh.XGrep(path, argv[x+1:])
1605             if 0 < found {
1606                 foundv := gsh.CmdCurrent.FoundFile
1607                 if len(foundv) < 10 {
1608                     gsh.CmdCurrent.FoundFile =
1609                         append(gsh.CmdCurrent.FoundFile, path)
1610                 }
1611             }
1612         }
1613     }
1614     if (isin("-r0", argv)) { // -d 0 in du, -depth n in find
1615         /total.Depth += 1
1616         if (fstat.Mode & S_IFMT) == S_IFLNK {
1617             continue
1618         }
1619         if fstat.RDev != fstat.Rdev {
1620             fmt.Println("-I-- don't follow differnet device %v(%v) %v(%v)\n",
1621                         dir, fstat.Rdev, path, fstat.Rdev)
1622         }
1623         if (fstat.Mode & S_IFMT) == S_IFDIR {
1624             total = gsh.xxFind(depth+1, total, path, npatv, argv)
1625         }
1626     }
1627 }
1628 }
1629 return total
1630 }
1631 func (gsh*GshContext)xxFind(depth int, total *fileSum, dir string, npatv []string, argv[]string)(*fileSum){
1632     nols := isin("-grep", argv)
1633     dirfile, err := os.OpenFile(dir, os.O_RDONLY, 0)
1634     if err != nil {
1635         //fmt.Printf("-I-- %v(%v)[%d]\n", dir, dirfile, dirfile.Fd())
1636         defer dirfile.Close()
1637     }else{
1638     }
1639     prev := *total
1640     var dstat aStat_t
1641     staterr := alstat(dir, &dstat) // should be flstat
1642     if staterr != nil {
1643         if isin("-w", argv){ fmt.Printf("ufind: %v\n", staterr) }
1644         return total
1645     }
1646     //iflev,err := ioutil.ReadDir(dir)
1647     //,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1648     /*
1649     if err != nil {
1650         if isin("-w", argv){ fmt.Printf("ufind: %v\n", err) }
1651         return total
1652     }
1653     */
1654     if depth == 0 {
1655         total = cumInfo(total, dir, staterr, dstat, argv, true)
1656         if !nols && !isin("-s", argv) && (!isin("-du", argv) || isin("-a", argv)) {
1657             showFileInfo(dir, argv)
1658         }
1659     }
1660     // it is not a directory, just scan it and finish
1661     for ei := 0; ; ei++ {
1662         entv,rderr := dirfile.Readdirnames(8*1024)
1663         if len(entv) == 0 || rderr != nil {
1664             //if rderr != nil { fmt.Printf("%d len=%d (%v)\n",ei,len(entv),rderr) }
1665             break
1666         }
1667         if 0 < ei {
1668             fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1669         }
1670         total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1671     }
1672     if isin("-du", argv) {
1673         // if in "du" mode
1674         fmt.Printf("%d\t%s\n", (total.Blocks-prev.Blocks)/2, dir)
1675     }
1676     return total
1677 }
1678 }
1679 // {ufind|fu|ls} [Files] [-- Names] [-- Expressions]
1680 // Files is "-" by default
1681 // Names is "-" by default
1682 // Expressions is "-" by default for "ufind", or -du for "fu" command
1683 func (gsh*GshContext)xFind(argv[]string){
1684     if 0 < len(argv) && strBegin(argv[0], "?"){
1685         showFound(gsh,argv)
1686         return
1687     }
1688     if isin("-cksum", argv) || isin("-sum", argv) {
1689         gsh.lastCheckSum = CheckSum()
1690         if isin("-sum", argv) && isin("-add", argv) {
1691             gsh.lastCheckSum.SumType |= SUM_SUM64
1692         }else{
1693             if isin("-sum", argv) && isin("-size", argv) {
1694                 gsh.lastCheckSum.SumType |= SUM_SIZE
1695             }else{
1696                 if isin("-sum", argv) && isin("-bsd", argv) {
1697                     gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
1698                 }else{
1699                     if isin("-sum", argv) && isin("-sysv", argv) {
1700                         gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
1701                     }else{
1702                         if isin("-sum", argv) {
1703                             gsh.lastCheckSum.SumType |= SUM_SUM64
1704                         }else{
1705                             if isin("-unix", argv) {
1706                                 gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1707                             }else{
1708                                 if isin("-ieee", argv){
1709                                     gsh.lastCheckSum.SumType |= SUM_CRCIEEE
1710                                     gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
1711                                 }else{
1712                                     gsh.lastCheckSum.RusgAtStart = Getrusagev()
1713                                 }
1714                             }
1715                         }
1716                     }
1717                 }
1718             }
1719             var total = fileSum{}
1720             npats = []string{}
1721             for v < range argv {
1722                 if 0 < len(v) && v[0] != '-' {
1723                     npats = append(npats,v)
1724                 }
1725                 if v == "/" { break }
1726                 if v == ":" { break }
1727                 if v == "-grep" { break }
1728                 if v == "-ls" { break }
1729             }
1730             if len(npats) == 0 {
1731                 npats = []string{"*"}
1732             }
1733             cwd := "."
1734             // if to be fullpath :: cwd, := os.Getwd()
1735             if len(npats) == 0 { npats = []string{"*"} }
1736             fusage := gsh.xxFind(&total,cwd,npats,argv)
1737             if gsh.lastCheckSum.SumType != 0 {
1738                 sum := uint64(0)
1739                 if (sum.SumType & SUM_SIZE) != 0 {
1740                     sum = uint64(sum.Size)
1741                 }
1742                 if (sum.SumType & SUM_SUM64) != 0 {
1743                     sum = sum.Sum64
1744                 }
1745                 if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1746                     s := uint32(sum.Sum16)
1747                     r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
1748                     sum = uint64((r & 0xFFFF) + (r >> 16))
1749                     sum.Crc32Val = uint32(s)
1750                     sum = uint64(s)
1751                 }
1752                 if (sum.SumType & SUM_SUM16_BSD) != 0 {
1753                     sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
1754                     sum = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
1755                 }
1756                 if (sum.SumType & SUM_UNIXFILE) != 0 {
1757                     sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
1758                     sum = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
1759                 }
1760             if l < sum.Files {
1761                 fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1762                         sum.Size,
1763                         absSize(sum.Size),sum.Files,
1764                         absSize(sum.Size/sum.Files))
1765             }else{
1766                 fmt.Printf("%v %v %v\n",
1767                         sum.Size,npats[0])
1768             }
1769         }

```

```

1770     if lisin("-grep",argv) {
1771         showUsage("total",fusage)
1772     }
1773     if lisin("-s",argv){
1774         hits := len(gsh.CmdCurrent.FoundFile)
1775         if 0 < hits {
1776             fmt.Printf("--I-- %d files hits // can be refered with !%df\n",
1777                 hits,len(gsh.CommandHistory))
1778         }
1779     }
1780     if gsh.lastCheckSum.SumType != 0 {
1781         if lisin("-ru",argv) {
1782             sum := gsh.lastCheckSum
1783             sum.Done.time.Now()
1784             gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1785             elps := sum.Done.SubSum.Start
1786             fmt.Printf("--cksun-size: %v (%v) / %v files, %v/file\r\n",
1787                 sum.Size,abssize(sum.Size),sum.Files,abssize(sum.Size/sum.Files))
1788             nanos := int64(elps)
1789             dnanos := float64(nanos)
1790             diff := UsageSub(sum.RusgAtEnd,sum.RusgAtStart)
1791             fmt.Printf("--cksun-time: %v/total, %v/file, %.1f files/s, %v\r\n",
1792                 abbtme(dnanos),
1793                 abbtme(time.Duration(nanos)/sum.Files),
1794                 (float64(sum.Files)*1000000000.0)/float64(nanos),
1795                 absbe(sum.Size,nanos))
1796             diff := UsageSub(sum.RusgAtEnd,sum.RusgAtStart)
1797             fmt.Printf("--cksun-rusg: %v\n",sUsage(argv,diff))
1798         }
1799     }
1800     return
1801 }
1802 func showFiles(files[]string){
1803     sp := ""
1804     for i,file := range files {
1805         if 0 < i { sp = " " } else { sp = "" }
1806         fmt.Printf(sp+"%s",escapeWhiteSP(file))
1807     }
1808 }
1809 func showFound(gshCtx *GshContext, argv[]string){
1810     for i,v := range gshCtx.CommandHistory {
1811         if 0 < i { v.Name() }
1812         fmt.Printf("%d (%d) ",i,len(v.FoundFile))
1813         if lisin("-ls",argv){
1814             for _file := range v.FoundFile {
1815                 for _file := range v.FoundFile {
1816                     fmt.Printf("%n") //sub number?
1817                     showFileInfo(file,argv)
1818                 }
1819             }
1820         }
1821         showFiles(v.FoundFile)
1822         fmt.Printf("\n")
1823     }
1824 }
1825 }
1826 func showMatchFile(filev []os.FileInfo, npat,dir string, argv[]string)(string,bool){
1827     fname := ""
1828     found := false
1829     for _v := range filev {
1830         match := filepath.Match(npat,(v.Name()))
1831         if match {
1832             fname = v.Name()
1833             found = true
1834             //fmt.Printf("%d) %s\n",i,v.Name())
1835             showIfExecutable(fname,dir,argv)
1836         }
1837     }
1838     return fname,found
1839 }
1840 func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
1841     var fullpath string
1842     if strBegins(name,DIRSEP){
1843         fullpath = name
1844     }else{
1845         if len(dir) == 0 ){
1846             fullpath = name;
1847         }else{
1848             fullpath = dir + DIRSEP + name
1849         }
1850         fi, err := os.Stat(fullpath)
1851         //fmt.Println("-dp-- `iv`\n-- %v\n",fullpath,err);
1852         if err != nil {
1853             fullpath += ".exe";
1854             fi, err = os.Stat(fullpath)
1855         }
1856         if err != nil {
1857             fullpath = dir + DIRSEP + name + ".go"
1858             fi, err = os.Stat(fullpath)
1859         }
1860         if err == nil {
1861             fm := fi.Mode()
1862             if fm.IsRegular() {
1863                 // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1864                 if aAccess(fullpath,5) == nil {
1865                     fullpath = fullpath
1866                     ffound = true
1867                     if ! lisin("-s", argv) {
1868                         showFileInfo(fullpath,argv)
1869                     }
1870                 }
1871             }
1872         }
1873     }
1874     return ffullpath,ffound
1875 }
1876 func which(list string, argv []string) (fullpathv []string, itis bool){
1877     if len(argv) <= 1 {
1878         fmt.Printf("Usage: which command [-s] [-a] [-ls]\n")
1879         return []string{}, false
1880     }
1881     path := argv[1]
1882     if strBegins(path,"/") {
1883         // should check if executable?
1884         exOK := showIfExecutable(path,"/",argv)
1885         fmt.Printf("-D- %v %v\n",path,exOK)
1886         return []string(path),exOK
1887     }
1888     pathenv, efound := os.LookupEnv(list)
1889     if ! efound {
1890         fmt.Printf("--E-- which: no `\$` in environment\n",list)
1891         return []string{}, false
1892     }
1893     //fmt.Println("PATH=%v\n",pathenv);
1894     showall := lisin("-a",argv) || 0 <= strings.Index(path,"*")
1895     dirv := strings.Split(pathenv,PATHSEP)
1896     ffound := false
1897     ffullpath := path
1898     for dir := range dirv {
1899         if 0 <= strings.Index(path,"*") { // by wild-card
1900             list,_ := ioutil.ReadDir(dir)
1901             ffullpath,ffound = showMatchFile(list,path,dir,argv)
1902         }else{
1903             ffullpath,ffound = showIfExecutable(path,dir,argv)
1904         }
1905         //if ffound && lishowall {
1906         if ffound && lishowall {
1907             break;
1908         }
1909     }
1910     return []string{ffullpath},ffound
1911 }
1912 }
1913 func stripLeadingNSParg(argv[]string)([]string){
1914     for i< len(argv); {
1915         if len(argv[0]) == 0 {
1916             argv = argv[1:]
1917         }else{
1918             break;
1919         }
1920     }
1921     return argv
1922 }
1923 func xEval(argv []string, nlend bool){
1924     argv = stripLeadingNSParg(argv)
1925     if len(argv) == 0 {
1926         fmt.Printf("eval [%vformat] [Go-expression]\n",
1927             return
1928     }
1929     pfmt := "%v"
1930     if argv[0][0] == '%' {
1931         pfmt = argv[0]
1932         argv = argv[1:]
1933     }
1934     if len(argv) == 0 {
1935         return
1936     }
1937     gocode := strings.Join(argv," ")
1938     //fmt.Println("eval [%v]\n",pfmt,gocode)
1939     fset := token.NewFileSet()
1940     rval, := types.Eval(fset,nil,token.NoPos,gocode)
1941     fmt.Println(pfmt,rval.Value)
1942     if nlend { fmt.Println("\n") }
1943 }
1944 func getval(name string) (found bool, val int) {
1945     /* should expand the name here */

```

```

1947     if name == "gsh.pid" {
1948         return true, os.Getpid()
1949     }else{
1950         if name == "gsh.ppid" {
1951             return true, os.Getppid()
1952         }
1953         return false, 0
1954     }
1955 }
1956 func echo(argv []string, nlen bool){
1957     for ai := 1; ai < len(argv); ai++ {
1958         if 1 < ai {
1959             fmt.Printf(" ")
1960         }
1961         arg := argv[ai]
1962         found, val := getval(arg)
1963         if found {
1964             fmt.Printf("%d",val)
1965         }else{
1966             fmt.Printf("%s",arg)
1967         }
1968     }
1969     if nlen {
1970         fmt.Printf("\n");
1971     }
1972 }
1973 func resfile() string {
1974     return "gsh.tmp"
1975 }
1976 //var resP *file
1977 func resmap(){
1978     // , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1979     // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1980     // , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1981     If err != nil {
1982         fmt.Printf("refF could not open: %s\n",err)
1983     }else{
1984         fmt.Printf("refF opened\n")
1985     }
1986 }
1987 }
1988 // @@2020-0821
1989 func gshScanArg(str string,strip int)(argv []string){
1990     var si = 0
1991     var sb = 0
1992     var inBracket = 0
1993     var arg = make([]byte,LINESIZE)
1994     var ax = 0
1995     debug := false
1996
1997     for ; si < len(str); si++ {
1998         if str[si] != ' ' {
1999             break
2000         }
2001     }
2002     sb = si
2003     for ; si < len(str); si++ {
2004         if sb < si {
2005             if debug {
2006                 fmt.Printf("-Da- %d %d %s\n",
2007                     inBracket,sb,si,argl[0:ax],str[si:])
2008             }
2009         }
2010     }
2011     ch := str[si]
2012     if ch == '{' {
2013         inBracket += 1
2014         if 0 < strip && inBracket <= strip {
2015             //fmt.Println("stripLEV %d <= %d\n",inBracket,strip)
2016             continue
2017         }
2018         if 0 < inBracket {
2019             if ch == ')' {
2020                 inBracket -= 1
2021                 if 0 < strip && inBracket < strip {
2022                     //fmt.Println("stripLEV %d < %d\n",inBracket,strip)
2023                     continue
2024                 }
2025             }
2026             argl[ax] = ch
2027             ax += 1
2028             continue
2029         }
2030     if str[si] == ' ' {
2031         argv = append(argv,string(argl[0:ax]))
2032         if debug {
2033             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2034                 -1:len(argv),sb,si,str[sb:si],string(str[si:]))
2035         }
2036         sb = si+1
2037         ax = 0
2038         continue
2039     }
2040     argl[ax] = ch
2041     ax += 1
2042     }
2043     if sb < si {
2044         argv = append(argv,string(argl[0:ax]))
2045         if debug {
2046             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2047                 -1+len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
2048         }
2049     }
2050     if debug {
2051         fmt.Printf("--Da- %d [%s] => [%d]@%v\n",strip,str,len(argv),argv)
2052     }
2053     return argv
2054 }
2055 }
2056 // should get stderr (into tmpfile ?) and return
2057 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
2058     //var pv = ([]int{-1,-1})
2059     //syscall.Pipe(pv)
2060
2061     xarg := gshScanArg(name,1)
2062     name = strings.Join(xarg, " ")
2063
2064     //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-{"+name+"}")
2065     //pout = os.NewFile(uintptr(pv[1]),"StdinOf-{"+name+"}")
2066     pin,pout,_ = os.Pipe();
2067
2068     fdir := 0
2069     dir := "="
2070     if mode == "r" {
2071         dir = "<"
2072     }
2073     fdir = 1 // read from the stdout of the process
2074     else{
2075         dir = ">" // write to the stdin of the process
2076     }
2077     gshPA := gsh.gshPA
2078     savfd := gshPA.Files[fdir]
2079
2080     var fd uintptr = 0
2081     if mode == "r" {
2082         //fd = pout.Fd()
2083         //gshPA.Files[fdir] = pout.Fd()
2084         gshPA.Files[fdir] = pout;
2085     }else{
2086         //fd = pin.Fd()
2087         //gshPA.Files[fdir] = pin.Fd()
2088         gshPA.Files[fdir] = pin;
2089     }
2090
2091     // should do this by Goroutine?
2092     if false {
2093         fmt.Printf("-Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2094         fmt.Printf("-REDL [%d,%d,%d]>[%d,%d,%d]\n",
2095             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2096             pin.Fd(),pout.Fd())
2097     }
2098     savi := os.Stdin
2099     savo := os.Stdout
2100     save := os.Stderr
2101     os.Stdout = pin
2102     os.Stderr = pout
2103     os.Stderr = pout
2104     gsh.BackGround = true
2105     gsh.gshellh(name)
2106     gsh.BackGround = false
2107     os.Stdin = savi
2108     os.Stdout = savo
2109     os.Stderr = save
2110
2111     gshPA.Files[fdir] = savfd
2112     return pin,pout,false
2113 }
2114
2115 // <a name="ex-commands">External commands</a>
2116 func (gsh*GshContext)execCommand(exec bool, argv []string) (notf bool,exit bool) {
2117     if gsh.CmdTrace {fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
2118
2119     gshPA := gsh.gshPA
2120     fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
2121     if itis == false {
2122         return true,false
2123     }

```

```

2124     fullPath := fullPathv[0]
2125     args := strings.Split(fullPathv[1], "/")
2126     if 0 < len(args) & args[0] == "gsh" {
2127         argv := args[1:]
2128         goFullPath, itis := which("PATH", argv)
2129         if itis == false {
2130             fmt.Printf("-F-- Go not found\n")
2131             return false, true
2132         }
2133         goFullPath := goFullPathv[0]
2134         argv = []string{goFullPath, "run", fullPath}
2135         fmt.Printf("-I-- %s %s\n", argv[0], argv[1])
2136         argv[0], argv[1], argv[2] = argv[1], argv[2], argv[0]
2137         if exec {
2138             syscall.Exec(goFullPath, argv, os.Environ())
2139         } else {
2140             //pid, _ := syscall.ForkExec(goFullPath, argv, &gshPA)
2141             proc, _ := os.StartProcess(goFullPath, argv, &gshPA);
2142             Pid := proc.Pid();
2143             Pid := pstat.Pid();
2144             if gsh.BackGround {
2145                 fmt.Fprintf(stderr, "-Ip- in Background pid[%d]\n", pid, len(argv), argv)
2146                 //gsh.BackGroundJobs = append(gsh.BackGroundJobs, pid)
2147                 gsh.BackGroundJobs = append(gsh.BackGroundJobs, *mPid)
2148             } else {
2149                 usage := aUsage {}
2150                 syscall.Wait4(pid, nil, 0, &usage)
2151                 gsh.LastUsage = usage
2152                 gsh.CmdCurrent.Rusagev[1] = usage
2153             }
2154         }
2155     /*
2156     gsh.LastUsage = *pstat.SysUsage().(*aUsage);
2157     gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aUsage);
2158     */
2159     aStrUsage(&gsh.LastUsage, pstat);
2160     gsh.CmdCurrent.Rusagev[1] = gsh.LastUsage;
2161 }
2162 */
2163 else{
2164     if exec {
2165         syscall.Exec(fullPath, argv, os.Environ())
2166     } else{
2167         //pid, _ := syscall.ForkExec(fullPath, argv, &gshPA)
2168         proc, _ := os.StartProcess(fullPath, argv, &gshPA);
2169         pstat := proc.Wait();
2170         Pid := pstat.Pid();
2171         //fmt.Printf("%d\n", pid); // '-t' to be background
2172     if( false ) {sys=v\n", gshPA.Sys);
2173     if( gshPA.Sys != nil )
2174         //fmt.Println("InFG=v\n", gshPA.Sys.Foreground);
2175     }
2176     if gsh.BackGround {
2177         fmt.Fprintf(stderr, "-Ip- in Background pid[%d]\n", pid, len(argv), argv)
2178         //gsh.BackGroundJobs = append(gsh.BackGroundJobs, pid)
2179         gsh.BackGroundJobs = append(gsh.BackGroundJobs, *sPid)
2180     } else{
2181         usage := aUsage {}
2182         syscall.Wait4(pid, nil, 0, &usage);
2183         gsh.LastUsage = usage
2184         gsh.CmdCurrent.Rusagev[1] = usage
2185     }
2186     /*
2187     gsh.LastUsage = *pstat.SysUsage().(*aUsage);
2188     gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aUsage);
2189     */
2190     aStrUsage(&gsh.LastUsage, pstat);
2191     gsh.CmdCurrent.Rusagev[1] = gsh.LastUsage;
2192 }
2193 }
2194 }
2195 */
2196 return false, false
2197 }
2198 }
2199 }
2200 }
2201 */
2202 // <a name="builtin">Builtin Commands</a>
2203 func (gshCtx *GshContext) sleep(argv []string) {
2204     if len(argv) < 2 {
2205         fmt.Println("Sleep 100ms, 100us, 100ns, ... \n")
2206         return
2207     }
2208     duration := argv[1];
2209     d, err := time.ParseDuration(duration)
2210     if err != nil {
2211         err := time.ParseDuration(duration+"s")
2212         if err != nil {
2213             fmt.Printf("duration ? %s\n", duration,err)
2214             return
2215         }
2216     }
2217     //fmt.Printf("Sleep %v\n",duration)
2218     time.Sleep(d)
2219     if 0 < len(argv[2:]) {
2220         gshCtx.gshellv(argv[2:])
2221     }
2222 }
2223 */
2224 func (gshCtx *GshContext) repeat(argv []string) {
2225     if len(argv) < 2 {
2226         return
2227     }
2228     start0 := time.Now()
2229     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2230         if 0 < len(argv[2:]) {
2231             /start := time.Now()
2232             gshCtx.gshellv(argv[2:])
2233             end := time.Now()
2234             elps := end.Sub(start0);
2235             if 1000000000 < elps {
2236                 fmt.Printf("repeat#%d %v\n", ri,elps);
2237             }
2238         }
2239     }
2240 }
2241 */
2242 func (gshCtx *GshContext) gen(argv []string) {
2243     gshPA := gshCtx.gshPA
2244     if len(argv) < 2 {
2245         fmt.Println("Usage: $ N\n", argv[0])
2246         return
2247     }
2248     // should br repeated by "repeat" command
2249     count := strconv.Atoi(argv[1])
2250     //fd := gshPA.Files[1] // Stdout
2251     //fd := os.NewFile(fd, "file", os.StdOut)
2252     file := gshPA.Files[1]; // Stdout
2253     fmt.Printf("-I-- Gen. Count=%d to %d\n", count,file.Fd())
2254     //buf := [byte]{}
2255     outdata := "0123 5678 0123 5678 0123 5678\r"
2256     for gi := 0; gi < count; gi++ {
2257         file.WriteString(outdata)
2258     }
2259     //file.WriteString("\n")
2260     fmt.Printf("\n%d B)\n",count*len(outdata));
2261     //file.Close()
2262 }
2263 */
2264 // <a name="rexec">Remote Execution</a> // 2020-0820
2265 func ElapsedFromTimeTime(string){
2266     elps := time.Now().Sub(frm)
2267     if 1000000000 < elps {
2268         return fmt.Sprintf("%d.%02ds", elps/1000000000,(elps%100000000)/10000000)
2269     }
2270     if 1000000 < elps {
2271         return fmt.Sprintf("%d.%03dms", elps/1000000,(elps%1000000)/1000)
2272     }
2273     else{
2274         return fmt.Sprintf("%d.%03dus", elps/1000,(elps%1000))
2275     }
2276 }
2277 */
2278 func abbtme(nanos int64)(string){
2279     if 1000000000 < nanos {
2280         return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%100000000)/10000000)
2281     }
2282     if 1000000 < nanos {
2283         return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
2284     }
2285     else{
2286         return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
2287     }
2288 }
2289 */
2290 func abbsize(size int64)(string){
2291     fsize := float64(size)
2292     if 1024*1024*1024 < size {
2293         return fmt.Sprintf("%2fGiB",fsize/(1024*1024*1024))
2294     }
2295     if 1024*1024 < size {
2296         return fmt.Sprintf("%3fMiB",fsize/(1024*1024))
2297     }
2298     else{
2299         return fmt.Sprintf("%3fKiB",fsize/1024)
2300     }
2301 }
2302 */

```

```

2301 }else{
2302     if 1024*1024 < size {
2303         return fmt.Sprintf("%8.3fMiB",fsize/(1024*1024))
2304     }else{
2305         return fmt.Sprintf("%8.3fKiB",fsize/1024)
2306     }
2307 }
2308 func abspeed(totalB int64,ns int64)(string){
2309     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2310     if 1000 <= MBs {
2311         return fmt.Sprintf("%6.3fGB/s",MBs/1000)
2312     }else{
2313         var dsize int64 = 32*1024*1024
2314         var bszie int64 = 64*1024
2315         var fname string = ""
2316         var in *os.File = nil
2317         var pseudoEOF = false
2318     }
2319     func abspeed(totalB int64,ns time.Duration)(string){
2320         MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2321         if 1000 <= MBs {
2322             return fmt.Sprintf("%6.3fGbps",MBs/1000)
2323         }else{
2324             if 1 <= MBs {
2325                 return fmt.Sprintf("%6.3fMBps",MBs)
2326             }else{
2327                 return fmt.Sprintf("%6.3fKBps",MBs*1000)
2328             }
2329     }
2330     func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
2331         Start0 := time.Now()
2332         buff := make([]byte,bsiz)
2333         var total int64 = 0
2334         var rem int64 = size
2335         nio := 0
2336         Prev := time.Now()
2337         var PrevSize int64 = 0
2338         fmt.Printf(Elapsed(Start0)+"--In- X: %s (%v/%v/%v) START\n",
2339             what,absize(total),size,nio)
2340         for i:= 0 ; i< len {
2341             var len = bsiz
2342             if int(rem) < len {
2343                 len = int(rem)
2344             }
2345             Now := time.Now()
2346             Elps := Now.Sub(Prev);
2347             if 1000000000 < Now.Sub(Prev) {
2348                 fmt.Printf(Elapsed(Start0)+"--In- X: %s (%v/%v/%v) %s\n",
2349                     what,absize(total),size,nio,
2350                     abspeed((total-PrevSize),Elps))
2351                 Prev = Now;
2352                 PrevSize = total
2353             }
2354             rlen := len
2355             if in != nil {
2356                 // should watch the disconnection of out
2357                 err,rcc,err,in.Read(buff[0:rlen])
2358                 if err != nil {
2359                     fmt.Printf(Elapsed(Start0)+"--En- X: %s read(%v,%v)<\n",
2360                         what,rcc,err,in.Name())
2361                     break
2362                 }
2363                 rlen = rcc
2364                 if string(buff[0:10]) == "((SoftEOF "
2365                     var ecc int64 = 0
2366                     fmt.Sscanf(string(buff),"((SoftEOF %v",&ecc)
2367                     fmt.Println(Elapsed(Start0)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
2368                         what,ecc,total)
2369                     if ecc == total {
2370                         break
2371                     }
2372                 }
2373             }
2374         }
2375         wlen := rlen
2376         if out != nil {
2377             wcc,err := out.Write(buff[0:rlen])
2378             if err != nil {
2379                 fmt.Printf(Elapsed(Start0)+"--En- X: %s write(%v,%v)>%v\n",
2380                     what,wcc,err,out.Name())
2381                 break
2382             }
2383             wlen = wcc
2384         }
2385         if wlen < rlen {
2386             fmt.Println(Elapsed(Start0)+"--En- X: %s incomplete write (%v/%v)\n",
2387                 what,wlen,rlen)
2388             break;
2389         }
2390         nio += 1
2391         total += int64(rlen)
2392         rem -= int64(rlen)
2393         if rem <= 0 {
2394             break
2395         }
2396     }
2397     done := time.Now()
2398     Elps := float64(Done.Sub(Start0))/1000000000 //Seconds
2399     TotalMB := float64(total)/1000000 //MB
2400     MBps := TotalMB / Elps
2401     fmt.Printf(Elapsed(Start0)+"--In- X: %s (%v/%v/%v) %v %.3fMB/s\n",
2402         what,total,size,nio,absize(total),MBps)
2403     return total
2404 }
2405 func tcppush(cint *os.File){
2406     // shrink socket buffer and recover
2407     usleep(100);
2408 }
2409 func (gsh*GshContext)RexecServer(argv[]string){
2410     debug := true
2411     Start0 := time.Now()
2412     Start := Start0
2413     // if local ":" { local = "0.0.0.0:9999" }
2414     local := "0.0.0.0:9999"
2415     if 0 < len(argv) {
2416         if argv[0] == "-s" {
2417             debug = false
2418             argv = argv[1:]
2419         }
2420         if 0 < len(argv) {
2421             argv = argv[1:]
2422         }
2423     }
2424     port, err := net.ResolveTCPAddr("tcp",local);
2425     if err != nil {
2426         fmt.Println("-En- S: Address error: %s (%s)\n",local,err)
2427         return
2428     }
2429     sconn, err := net.ListenTCP("tcp",port)
2430     if err != nil {
2431         fmt.Println(Elapsed(Start0)+"--In- S: Listen error: %s (%s)\n",local,err)
2432         return
2433     }
2434     reqbuf := make([]byte,LINESIZE)
2435     res := ""
2436     for {
2437         fmt.Println(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2438         accon,err := sconn.AcceptTCP()
2439         Start = time.Now()
2440         if err != nil {
2441             fmt.Println(Elapsed(Start0)+"--En- S: Accept error: %s (%s)\n",local,err)
2442             return
2443         }
2444         clnt,_ := accon.File()
2445         fd := Clnt.Fd()
2446         ar := accon.RemoteAddr()
2447         if debug { fmt.Println(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2448             local,fd,ar)
2449         }
2450         res,_ := fd.Read("220 GShell/%s Server\r\n",VERSION)
2451         fmt.Println("%s",res)
2452         if debug { fmt.Println(Elapsed(Start0)+"--In- S: %s",res) }
2453         count, err := clnt.Read(reqbuf)
2454         if err != nil {
2455             fmt.Println(Elapsed(Start0)+"--En- C: (%v %v) %v",
2456                 count,err,string(reqbuf))
2457         }
2458         req := string(reqbuf[:count])
2459         if debug { fmt.Println(Elapsed(Start0)+"--In- C: %v",string(req)) }
2460         reqv := strings.Split(string(req),"r")
2461         cmdv := gshSshParseReqv0();
2462         //cmdv := strings.Split(cmdv[0]," ")
2463         switch cmdv[0] {
2464             case "HELLO":
2465                 res = fmt.Sprintf("250 %v",req)
2466             case "GET":
2467                 if reqv[0] == "load {remotefile|~N} [localfile]"
2468                     var dsize int64 = 32*1024*1024
2469                     var bszie int64 = 64*1024
2470                     var fname string = ""
2471                     var in *os.File = nil
2472                     var pseudoEOF = false
2473             }
2474         }
2475     }
2476 }
```

```

2478     if 1 < len(cmdv) {
2479         fname = cmdv[1]
2480         if strBegin(fname, "-z") {
2481             fmt.Sscanf(fname[2:], "%d", &dsize)
2482         }else{
2483             if strBegin(fname, "(") {
2484                 xin,xout,err := gsh.Popen(fname, "r")
2485                 if err != nil {
2486                     defer xin.Close()
2487                     defer xout.Close()
2488                     in = xin
2489                     dsize = MaxStreamSize
2490                     pseudoEOF = true
2491                 }
2492             }else{
2493                 xin,err := os.Open(fname)
2494                 if err != nil {
2495                     fmt.Printf("--En- GET (%v)\n",err)
2496                 }else{
2497                     defer xin.Close()
2498                     in = xin
2499                     fi,_ := xin.Stat()
2500                     dsize = fi.Size()
2501                 }
2502             }
2503         }
2504     }/fmt.Printf(Elapsed(Start)+"--In- GET %v:\n",dsize,bsize)
2505     res = fmt.Sprintf("200 %v\r\n",dsize)
2506     fmt.Fprintf(clnt,"%v",res)
2507     bsize := 1024 // send data be separated as line in receiver
2508     fmt.Println(Elapsed(Start)+"--In- S: %v\n",res)
2509     wcount := fileRelay("SendGET",in,clnt,dsize,bsize)
2510     if pseudoEOF {
2511         in.Close() // pipe from the command
2512         // show end of stream data (its size) by OOB?
2513         SoftEOF := fmt.Sprintf("(%dEOF %v)",wcount)
2514         fmt.Println(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2515     }
2516     tcpPush(clnt); // to let SoftEOF data appear at the top of received data
2517     fmt.Fprintf(clnt,"%v\r\n",SoftEOF)
2518     tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2519     // with client generated random?
2520     //fmt.Println("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2521     }
2522     res = fmt.Sprintf("200 GET done\r\n")
2523 case "PUT":
2524     upload := os.Getenv("ZNAME") [dstfile]
2525     var dsize int64 = 32*1024*1024
2526     var bsize int = 64*1024
2527     var fname string = ""
2528     var out *os.File = nil
2529     if 1 < len(cmdv) { // localfile
2530         fmt.Sscanf(cmdv[1],"%d", &dsize)
2531     }
2532     if 2 < len(cmdv) {
2533         fname = cmdv[2]
2534         if fname == "_" {
2535             // nul dev
2536         }else{
2537             if strBegin(fname,"(") {
2538                 xin,xout,err := gsh.Popen(fname, "w")
2539                 if err != nil {
2540                     defer xin.Close()
2541                     out = xout
2542                 }
2543             }else{
2544                 // should write to temporary file
2545                 // should suppress ^C on tty
2546                 xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2547                 if err != nil {
2548                     fmt.Printf("--En- PUT (%v)\n",err)
2549                 }else{
2550                     out = xout
2551                 }
2552             }
2553             fmt.Println(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2554                 fname,local,err)
2555         }
2556     }
2557     fmt.Println(Elapsed(Start)+"--In- P: 200 %v OK\r\n",dsize)
2558     fmt.Fprintf(clnt,"%v\r\n",dsize)
2559     fileRelay("RecvPUT",clnt,out,dsize,bsize)
2560     res = fmt.Sprintf("200 PUT done\r\n")
2561     default:
2562     res = fmt.Sprintf("400 What? %v",req)
2563 }
2564 swcc,err := clnt.Write([]byte(res))
2565 if err != nil {
2566     fmt.Println(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,err,res)
2567 }else{
2568     fmt.Println(Elapsed(Start)+"--In- S: %v",res)
2569 }
2570 aconn.Close();
2571 clnt.Close();
2572 }
2573 sconn.Close();
2574 }
2575 func (gsh*GshContext)ReexecClient(argv[]string)(int,string){
2576     debug := true
2577     Start := time.Now()
2578     if len(argv) == 1 {
2579         return -1,"EmptyARG"
2580     }
2581     argv = argv[1:]
2582     if argv[0] == "-serv" {
2583         gsh.ReexecServer(argv[1:])
2584         return 0,"Server"
2585     }
2586     remote := "0.0.0.0:9999"
2587     if argv[0][0] == '@' {
2588         remote = argv[0][1:];
2589         argv = argv[1:];
2590     }
2591     if argv[0] == "-s" {
2592         debug = false
2593         argv = argv[1:];
2594     }
2595     dport, err := net.ResolveTCPAddr("tcp",remote);
2596     if err != nil {
2597         fmt.Println(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
2598         return -1,"AddressError"
2599     }
2600     fmt.Println(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
2601     serv, err := net.DialTCP("tcp",nil,dport)
2602     if err != nil {
2603         fmt.Println(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
2604         return -1,"CannotConnect"
2605     }
2606     if debug {
2607         al := serv.LocalAddr()
2608         fmt.Println(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
2609     }
2610     res := ""
2611     req := make([]byte,LINESIZE)
2612     count,err := serv.Read(res)
2613     if err != nil {
2614         fmt.Printf("--En- S: (%d,%v) %v",count,err,string(res))
2615     }
2616     if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",string(res)) }
2617     if argv[0] == "GET" {
2618         savPA := gsh.gshPA
2619         var bsize int = 64*1024
2620         req := fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2621         fmt.Println(Elapsed(Start)+"--In- C: %v",req)
2622         fmt.Fprintf(serv,req)
2623         count,err = serv.Read(res)
2624         if err != nil {
2625             if err != nil {
2626                 var dsize int64 = 0
2627                 var out *os.File = nil
2628                 var out_tobeclosed *os.File = nil
2629                 var fname string = ""
2630                 var rcode int = 0
2631                 var pid int = 0
2632                 fmt.Println(Elapsed(Start),"fd %d",rcode,&dsize)
2633                 fmt.Println(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
2634                 if 3 < len(argv) {
2635                     fname = argv[2]
2636                     if strBegin(fname,"(") {
2637                         xin,xout,err := gsh.Popen(fname, "w")
2638                         if err != nil {
2639                             xin.Close()
2640                             defer xout.Close()
2641                             out = xout
2642                             out_tobeclosed = xout
2643                             pid = 0 // should be its pid
2644                         }
2645                     }else{
2646                         // should write to temporary file
2647                         // should suppress ^C on tty
2648                     }
2649                 }
2650             }
2651         }
2652     }
2653 }
```

```

2655         xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2656         if err != nil {
2657             fmt.Println("--En- %v\n",err)
2658         }
2659         out = xout
2660         //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2661     }
2662     in,_ := serv.File()
2663     fileRelay("RecvGET",in,out,dsize,bsize)
2664     if 0 <= pid {
2665         gsh.gshPA = savPA // recovery of Fd(), and more?
2666         fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2667         out_toBeClosed.Close()
2668         //syscall.Wait4(pid,nil,0,nil) //@@
2669     }
2670 }
2671 }
2672 if argv[0] == "PUM" {
2673     remote, _ := serv.File()
2674     var local *os.File = nil
2675     var dsize int64 = 32*1024*1024
2676     var bsize int = 64*1024
2677     var ofile string = ""
2678     //fmt.Printf("--In- %s Rex %v\n",argv)
2679     if len(argv) < len(argv) {
2680         fname := argv[1]
2681         if strBegins(fname,"-z") {
2682             fmt.Sscanf(fname[2:],"%d",&dsize)
2683         }else{
2684             if strBegins(fname,"{") {
2685                 xin,xout,err := gsh.Popen(fname,"r")
2686                 if err != nil {
2687                     if else{
2688                         xout.Close()
2689                         defer xin.Close()
2690                         /in/xin
2691                         local = xin
2692                         fmt.Println("--In- [%d] < Upload output of %v\n",
2693                             local.Fd(),fname)
2694                         ofile = ".from."+fname
2695                         dsize = MaxStreamsize
2696                     }
2697                 }else{
2698                     xlocal,err := os.Open(fname)
2699                     if err != nil {
2700                         fmt.Println("--En- (%s)\n",err)
2701                         local = nil
2702                     }else{
2703                         local = xlocal
2704                         fi,_ := local.Stat()
2705                         dsize = fi.Size()
2706                         defer local.Close()
2707                         //fmt.Printf("--In- Rex in(%v / %v)\n",ofile,dsize)
2708                     }
2709                     ofile = fname
2710                     fmt.Println(Elapsed(Start)+"--In- L: open(%v,r=%v %v (%v)\n",
2711                         fname,dsize,local,err)
2712                 }
2713             }
2714             if 2 < len(argv) && argv[2] != "" {
2715                 ofile = argv[2]
2716                 //fmt.Println("(d)%s B.ofile=%v\n",len(argv),argv,ofile)
2717             }
2718             //fmt.Printf("Elapsed(Start)+"--In- Rex out(%v)\n",ofile)
2719             fmt.Println(Elapsed(Start)+"--In- PUT %v (%v)\n",dsize,bsize)
2720             req = fmt.Sprintf("PUT %v (%v)",dsize,ofile)
2721             if debug { fmt.Println(Elapsed(Start)+"--In- C: %v",req) }
2722             fmt.Fprintf(serv,"%v",req)
2723             count,err = serv.Read(res)
2724             if debug { fmt.Println(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2725             fileRelay("SendPUT",local,remote,dsize,bsize)
2726         }else{
2727             req = fmt.Sprintf("%v\r\n",strings.Join(argv, " "))
2728             if debug { fmt.Println(Elapsed(Start)+"--In- C: %v",req) }
2729             fmt.Fprintf(serv,"%v",req)
2730             //fmt.Println("--In- sending RexRequest(%v)\n",len(req))
2731             if debug { fmt.Println(Elapsed(Start)+"--In- waiting RexResponse...\n") }
2732             count,err = serv.Read(res)
2733             res := ""
2734             if count == 0 {
2735                 res = "(nil)\r\n"
2736             }else{
2737                 res = string(res[:count])
2738             }
2739             if err != nil {
2740                 fmt.Println(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
2741             }else{
2742                 fmt.Println(Elapsed(Start)+"--In- S: %v",res)
2743             }
2744             serv.Close()
2745             //conn.Close()
2746             var stat string
2747             var rcode int
2748             fmt.Sscanf(res,"%d %s",&rcode,&stat)
2749             //fmt.Println("--D-- Client: %v (%v)",rcode,stat)
2750             return rcode,res
2751         }
2752     }
2753     // <a name="remote-sh">Remote Shell</a>
2754     // gcp file [...] { [host:[port:]]|dir | dir } // -p | -no-p
2755     func (gsh *GshContext) FileCopy(argv[]string){
2756         var host = ""
2757         var port = ""
2758         var upload = false
2759         var download = false
2760         var xargv = []string{"rex-gcp"}
2761         var srcv = []string{}
2762         var dstv = []string{}
2763         argv = argv[1:]
2764         for _v := range argv {
2765             /* if v[0] == '-' { // might be a pseudo file (generated date)
2766                continue
2767            }*/
2768             obj := strings.Split(v,":")
2769             //fmt.Println(*v+" "+obj[0]+obj[1])
2770             if 1 < len(obj) {
2771                 host = obj[0]
2772                 file := "-"
2773                 if 0 < len(host) {
2774                     gsh.LastServer.host = host
2775                 }else{
2776                     host = gsh.LastServer.host
2777                     port = gsh.LastServer.port
2778                 }
2779                 if 2 < len(obj) {
2780                     port = obj[1]
2781                     if 0 < len(port) {
2782                         gsh.LastServer.port = port
2783                     }else{
2784                         port = gsh.LastServer.port
2785                     }
2786                     file = obj[2]
2787                 }else{
2788                     file = obj[1]
2789                 }
2790                 if len(srcv) == 0 {
2791                     download = true
2792                     srcv = append(srcv,file)
2793                     continue
2794                 }
2795                 upload = true
2796                 dstv = append(dstv,file)
2797             continue
2798             }*/
2799             idx := strings.Index(v,":")
2800             if 0 <= idx {
2801                 remote := v[0:idx]
2802                 if len(remote) == 0 {
2803                     download = true
2804                     srcv = append(srcv,v[idx+1:])
2805                 }else{
2806                     upload = true
2807                     dstv = append(dstv,v[idx+1:])
2808                 }
2809             */
2810             if download {
2811                 dstv = append(dstv,v)
2812             }else{
2813                 srcv = append(srcv,v)
2814             }
2815         }
2816         hostport := "@" + host + ":" + port
2817         if upload != "" { xargv = append(xargv,hostport) }
2818         xargv = append(xargv,"PUT")
2819         xargv = append(xargv,srcv[0]...)
2820         xargv = append(xargv,dstv[0]...)
2821         //fmt.Printf("--In-- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)
2822     }
2823 }
```

```

2832 fmt.Printf("--I-- FileCopy PUT gsh://$s/$v < $v\n", hostport,dstv,srcv)
2833 gsh.RexecClient(xargv)
2834 }
2835 if download != "" { xargv = append(xargv,hostport) }
2836 xargv = append(xargv,"GET")
2837 xargv = append(xargv,srcv[0]...)
2838 xargv = append(xargv,dstv[0]...)
2839 //fmt.Println("--I-- FileCopy GET gsh://$v/$v > $v // $v\n",hostport,srcv,dstv,xargv)
2840 fmt.Printf("--I-- FileCopy GET gsh://$v/$v > $v // $v\n",hostport,srcv,dstv)
2841 gsh.RexecClient(xargv)
2842 }
2843 }
2844 }
2845 }
2846 }
2847 // target
2848 func (gsh*GshContext)Trepath(rloc string)(string){
2849 cwd,_ := os.Getwd()
2850 os.Cchdir(gsh.RWD)
2851 os.Cchdir(rloc)
2852 twd,_ := os.Getwd()
2853 os.Cchdir(cwd)
2854 tpath := twd + "/" + rloc
2855 return tpath
2856 }
2857 // join to remote GShell - [user@host[:port] or cd host[:port]:path
2858 func (gsh*GshContext)Rjoin(argv[]string){
2859 if len(argv) <= 1 {
2860 fmt.Printf("--I-- current server = $v\n",gsh.RSERV)
2861 return
2862 }
2863 serv := argv[1]
2864 servv := strings.Split(serv,":")
2865 if 1 <= len(servv) {
2866 if servv[0] == "lo" {
2867 servv[0] = "localhost"
2868 }
2869 }
2870 }
2871 switch len(servv) {
2872 case 1:
2873 if strings.Index(servv[0]) < 0 {
2874 serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2875 "/"}
2876 case 2: // host:port
2877 serv = strings.Join(servv,:)
2878 }
2879 xargv := []string{"rex-join","@"+serv, "HELLO"}
2880 rcode,stat := gsh.RexecClient(xargv)
2881 if (rcode / 100) == 2 {
2882 fmt.Printf("--I-- OK Joined ($v) $v\n",rcode,stat)
2883 gsh.RSERV = serv
2884 }else{
2885 fmt.Printf("--I-- NG, could not joined ($v) $v\n",rcode,stat)
2886 }
2887 }
2888 func (gsh*GshContext)Rexec(argv[]string){
2889 if len(argv) <= 1 {
2890 fmt.Printf("--I-- reexec command [ | {file || {command} ]\n",gsh.RSERV)
2891 return
2892 }
2893 /*
2894 nargv := gshScanArgv(strings.Join(argv, " "),0)
2895 fmt.Printf("--D-- nargc=%d $v\n",len(nargv),nargv)
2896 if nargv[1][0] != '{' {
2897 nargv[1] = "(" + nargv[1] + ")"
2898 fmt.Printf("--D-- nargc=%d ($v)\n",len(nargv),nargv)
2899 argv = nargv
2900 */
2901 argv = nargv
2902 /*
2903 nargv := []string{}
2904 nargv = append(nargv,""+strings.Join(argv[1:], " ")+"")
2905 fmt.Printf("--D-- nargc=%d $v\n",len(nargv),nargv)
2906 argv = nargv
2907 */
2908 xargv := []string{"rex-exec","@"+gsh.RSERV,"GET"}
2909 xargv = append(xargv,argv...)
2910 xargv = append(xargv,"/dev/tty")
2911 rcode,stat := gsh.RexecClient(xargv)
2912 if (rcode / 100) == 2 {
2913 fmt.Printf("--I-- OK Exec ($v) $v\n",rcode,stat)
2914 }else{
2915 fmt.Printf("--I-- NG Exec ($v) $v\n",rcode,stat)
2916 }
2917 }
2918 func (gsh*GshContext)Rchdir(argv[]string){
2919 if len(argv) <= 1 {
2920 return
2921 }
2922 cwd,_ := os.Getwd()
2923 os.Cchdir(gsh.RWD)
2924 os.Cchdir(argv[1])
2925 twd,_ := os.Getwd()
2926 gsh.RWD = twd
2927 fmt.Printf("--I-- JWD=$v\n",twd)
2928 os.Cchdir(cwd)
2929 }
2930 func (gsh*GshContext)Rpwd(argv[]string){
2931 fmt.Println("$v\n",gsh.RWD)
2932 }
2933 func (gsh*GshContext)Rls(argv[]string){
2934 cwd,_ := os.Getwd()
2935 os.Cchdir(gsh.RWD)
2936 argv[0] = "-ls"
2937 gsh.XFind(argv)
2938 os.Cchdir(cwd)
2939 }
2940 func (gsh*GshContext)Rput(argv[]string){
2941 var local string = ""
2942 var remote string = ""
2943 if 1 < len(argv) {
2944 local = argv[1]
2945 remote = local // base name
2946 }
2947 if 2 < len(argv) {
2948 remote = argv[2]
2949 }
2950 fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trepath(remote))
2951 }
2952 func (gsh*GshContext)Rget(argv[]string){
2953 var remote string = ""
2954 var local string = ""
2955 if 1 < len(argv) {
2956 remote = argv[1]
2957 local = remote // base name
2958 }
2959 if 2 < len(argv) {
2960 local = argv[2]
2961 }
2962 fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trepath(remote),local)
2963 }
2964 //}
2965 // <a name="network">network</a>
2966 // -s, -si, -so // bi-directional, source, sync (maybe socket)
2967 func (gshCtx*GshContext)connect(inTCP bool, argv []string) {
2968 gshPA := gshTx,gshPA
2969 if len(argv) < 2 {
2970 fmt.Printf("Usage: -s [host]:[port].udp]\n")
2971 return
2972 }
2973 remote := argv[1]
2974 if remote == ":"{ remote = "0.0.0.0:9999" }
2975 if inTCP {
2976 dport, err := net.ResolveTCPAddr("tcp",remote);
2977 if err != nil {
2978 fmt.Printf("Address error: %s (%s)\n",remote,err)
2979 return
2980 }
2981 conn, err := net.DialTCP("tcp",nil,dport)
2982 if err != nil {
2983 fmt.Printf("Connection error: %s (%s)\n",remote,err)
2984 return
2985 }
2986 file,_ := conn.File();
2987 //fd := file.Fd()
2988 //fmt.Println("Socket: connected to $s, socket($d)\n",remote,fd)
2989 fmt.Println("Socket: connected to $s, socket($d)\n",remote,file.Fd())
2990 }
2991 safd := gshPA.Files[1]
2992 //gshPA.Files[1] = fd;
2993 gshPA.Files[1] = file;
2994 gshCtxt.gshellv(argv[2]);
2995 gshPA.Files[1] = safd
2996 file.Close()
2997 conn.Close()
2998 }
2999 else{
3000 //dport, err := net.ResolveUDPAddr("udp4",remote);
3001 dport, err := net.ResolveUDPAddr("udp",remote);
3002 if err != nil {
3003 fmt.Printf("Address error: %s (%s)\n",remote,err)
3004 return
3005 }
3006 //conn, err := net.DialUDP("udp4",nil,dport)
3007 conn, err := net.DialUDP("udp",nil,dport)
3008 if err != nil {
}

```

```

3009     fmt.Printf("Connection error: %s (%s)\n",remote,err)
3100     return
311 }
312 file,_ := conn.File();
313 //fd,_ := file.Fd()
314
315 ar := conn.RemoteAddr()
316 //ar := conn.LocalAddr()
317 fmt.Printf("Socket: connected to %s, socket[%d]\n",
318     "/remote",ar.String(),fd)
319 remote,ar.String(),file.Fd())
320
321 savfd := gshPA.Files[1]
322 //gshPA.Files[1] = fd;
323 gshPA.Files[1] = file;
324 gshCtx.gshellv(argv[2:])
325 gshPA.Files[1] = savfd
326 file.Close()
327 conn.Close()
328 }
329
330 func (gshCtx*GshContext)accept(inTCP bool, argv []string) {
331     gshPA := gshCtx.gshPA
332     if len(argv) < 2 {
333         fmt.Printf("Usage: -ac [host]:[port].udp]\n")
334         return
335     }
336     local := argv[1]
337     if local == ":" { local = "0.0.0.0:9999" }
338     if inTCP { // TCP
339         port := net.ResolveTCPAddr("tcp",local);
340         if err != nil {
341             fmt.Printf("Address error: %s (%s)\n",local,err)
342             return
343         }
344         fmt.Println("Listen at %s..\n",local);
345         sconn, err := net.ListenTCP("tcp", port)
346         if err != nil {
347             fmt.Printf("Listen error: %s (%s)\n",local,err)
348             return
349         }
350         //fmt.Println("Accepting at %s..\n",local);
351         sconn, err := sconn.AcceptTCP()
352         if err != nil {
353             fmt.Printf("Accept error: %s (%s)\n",local,err)
354             return
355         }
356         file,_ := sconn.File()
357         //fd,_ := file.Fd()
358         //fmt.Println("Accepted TCP at %d\n",local,fd)
359         fmt.Println("Accepted TCP at %d\n",local,file.Fd())
360
361         savfd := gshPA.Files[0]
362         //gshPA.Files[0] = fd;
363         gshPA.Files[0] = file;
364         gshCtx.gshellv(argv[2:])
365         gshPA.Files[0] = savfd
366
367         sconn.Close();
368         sconn.Close();
369         file.Close();
370     }else{
371         //port, err := net.ResolveUDPAddr("udp4",local);
372         port, err := net.ResolveUDPEndPoint("udp",local);
373         if err != nil {
374             fmt.Printf("Address error: %s (%s)\n",local,err)
375             return
376         }
377         fmt.Println("Listen UDP at %s..\n",local);
378         //uconn,err := net.ListenUDP("udp4", port)
379         uconn, err := net.ListenUDP("udp", port)
380         if err != nil {
381             fmt.Printf("Listen error: %s (%s)\n",local,err)
382             return
383         }
384         file,_ := uconn.File()
385         //fd,_ := file.Fd()
386         ar := uconn.RemoteAddr()
387         remote := ""
388         if ar != nil { remote = ar.String() }
389         if remote == "" { remote = "?" }
390
391         // not yet received
392         //fmt.Println("Accepted at %d <- %s\n",local,fd,"")
393
394         savfd := gshPA.Files[0]
395         //gshPA.Files[0] = fd;
396         gshPA.Files[0] = file;
397         savenv := gshPA.Env
398         gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
399         gshCtx.gshellv(argv[2:])
400         gshPA.Env = savenv
401         gshPA.Files[0] = savfd
402
403         uconn.Close();
404         file.Close();
405     }
406 }
407
408 // empty line command
409 func (gshCtx*GshContext)xPwd(argv[]string){
410     // execute context command, pwd + date
411     // context notation, representation scheme, to be resumed at re-login
412     cwd := os.Getwd()
413     switch argv[0] {
414     case isn("-a",argv):
415         gshCtx.ShowChdirHistory(argv)
416     case isn("-ls",argv):
417         showFileInfo(cwd,argv)
418     default:
419         fmt.Println("$s\n",cwd)
420     case isn("-v",argv): // obsolete empty command
421         t := time.Now()
422         date := t.Format(time.UnixDate)
423         exe := os.Executable()
424         host := runtime.GOOS
425         fmt.Printf("PWD=%s\n", cwd)
426         fmt.Printf("HOST=%s\n", host)
427         fmt.Printf("DATE=%s\n",date)
428         fmt.Printf("TIME=%s\n",t.String())
429         fmt.Printf("PID=%d\n",os.Getpid())
430         fmt.Printf("EXE=%s\n",exe)
431         fmt.Println("}\n")
432     }
433 }
434
435 // <a name="history">History</a>
436 // there should be browsed and edited by HTTP browser
437 // show the time of command with -t and directory with -ls
438 // openfile-history, sort by -a -m -c
439 // sort by elapsed time by -t -s
440 // search like "more" like interface
441 // history
442 // sort history, and wc or uniq
443 // CPU and other resource consumptions
444 // limit showing range (by time or so)
445 // export / import history
446 func (gshCtx*GshContext)xHistory(argv []string){
447     atWorkDirX := 1
448     if 1 < len(argv) && strBegins(argv[1],"") {
449         atWorkDirX,_ = strconv.Atoi(argv[1][1:1])
450     }
451     //fmt.Println("-D-- showHistory(%v)\n",argv)
452     for i:=1;i<range gshCtx.CommandHistory; {
453         // exclude commands not to be listed by default
454         // internal commands may be suppressed by default
455         if v.Cmdline == "" && !isin("-a",argv) {
456             continue;
457         }
458         if 0 <= atWorkDirX {
459             if v.WorkDirX != atWorkDirX {
460                 continue
461             }
462         }
463         if !isin("-n",argv){ // like "fc"
464             fmt.Println("1%-2d ",i)
465         }
466         if isn("-v",argv){ // should be with it date
467             if isn("-l",argv) || isn("-10",argv) {
468                 elps := v.EndAt.Sub(v.StartAt).Format(time.Stamp)
469                 start := v.StartAt.Format(time.Stamp)
470                 fmt.Printf("%d ",v.WorkDirX)
471                 fmt.Println("[v] ilv/t ",start,elps)
472             }
473             if isn("-1",argv) && isn("-lo",argv){
474                 fmt.Println("%v",v.Rusage)
475             }
476             if isn("-at",argv) { // isn("-ls",argv){
477                 dhi := v.WorkDirX // workdir history index
478                 fmt.Println("%d %t",dhi,v.WorkDirX)
479             }
480             // show the FileInfo of the output command??
481         }
482         fmt.Println("%s",v.Cmdline)
483         fmt.Println("\n")
484     }
485 }
```

```

3186     }
3187   // In - history index
3188   func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
3189     if gline[0] == '!' {
3190       hix, err := strconv.Atoi(gline[1:])
3191       if err != nil {
3192         fmt.Printf("-E-- (%s : range)\n",hix)
3193         return "", false, true
3194       }
3195       if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3196         fmt.Printf("-E-- (%d : out of range)\n",hix)
3197         return "", false, true
3198       }
3199       return gline, false, false
3200     }
3201   }
3202   // search
3203   //for i, v := range gshCtx.CommandHistory {
3204   //}
3205   return gline, false, false
3206 }
3207 func (gsh*GshContext)xCmdStringInHistory(hix int)(cmd string, ok bool){
3208   if 0 <= hix && hix < len(gsh.CommandHistory) {
3209     return gsh.CommandHistory[hix].CmdLine,true
3210   }
3211   return "",false
3212 }
3213
3214 // temporary adding to PATH environment
3215 // cdname -l lib for LD_LIBRARY_PATH
3216 // cdname -L lib for LIBRARY_PATH
3217 // for sort option (by visit date or so)
3218 func (gsh*GshContext>ShowChdirHistory(i int,v GChdirHistory, argv []string){
3219   fmt.Printf("%-2d ",v.CmdIndex) // the first command at this WorkDir
3220   fmt.Printf("%d ",i)
3221   fmt.Printf("(%) ",v.MovedAt.Format(time.Stamp))
3222   showFileInfo(v.Dir,argv)
3223 }
3224 func (gsh*GshContext>ShowChdirHistory(argv []string){
3225   for i, v := range gsh.ChdirHistory {
3226     gsh.ShowChdirHistory(i,v,argv)
3227   }
3228 }
3229 func skipOpts(argv[]string)(int){
3230   for i,v := range argv {
3231     if strbegins(v,"-") {
3232       }else{
3233         return i
3234     }
3235   }
3236   return -1
3237 }
3238 func (gshCtx*GshContext)xChdir(argv []string){
3239   osHist := gshCtx.ChdirHistory
3240   if isn("?",argv) || isn("-t",argv) || isn("-a",argv) {
3241     gshCtx.ShowChdirHistory(argv)
3242     return
3243   }
3244   pwd, _ := os.Getwd()
3245   dir := ""-path)
3246   if len(argv) <= 1 {
3247     dir = toFullPath("-")
3248   }else{
3249     i := skipOpts(argv[1:])
3250     if i < 0 {
3251       dir = toFullPath("-")
3252     }else{
3253       dir = argv[i+1]
3254     }
3255   }
3256   if strbegins(dir,"@") {
3257     if dir == "@0" { // obsolete
3258       dir = gshCtx.Startdir
3259     }else{
3260       if dir == "@i" {
3261         index := len(cdhist) - 1
3262         if 0 < index { index -= 1 }
3263         dir = cdhist[index].Dir
3264       }else{
3265         index, err := strconv.Atoi(dir[1:])
3266         if err != nil {
3267           fmt.Printf("-E-- xChdir(%s)\n",err)
3268           dir = "?"
3269         }else{
3270           if len(gshCtx.ChdirHistory) <= index {
3271             fmt.Printf("-E-- xChdir(history range error)\n")
3272             dir = "?"
3273           }else{
3274             dir = cdhist[index].Dir
3275           }
3276         }
3277       }
3278     if dir != "?" {
3279       err := os.Chdir(dir)
3280       if err != nil {
3281         fmt.Printf("-E-- xChdir(%s)(%s)\n",argv[1],err)
3282       }else{
3283         cwd, cwd := os.Getwd()
3284         if cwd == cwd {
3285           hist1 := GChdirHistory( )
3286           hist1.Dir = cwd
3287           hist1.MovedAt = time.Now()
3288           hist1.CmdIndex = len(gshCtx.CommandHistory)+1
3289           gshCtx.ChdirHistory.append(cdhist,hist1)
3290           if isn("s",argv){
3291             cwd, cwd := os.Getwd()
3292             cwd := cwd"\n"
3293             cwd := len(gshCtx.ChdirHistory)-1
3294             gshCtx.ShowChdirHistory(i,hist1,argv)
3295           }
3296         }
3297       }
3298     if isn("-ls",argv){
3299       cwd, cwd := os.Getwd()
3300       showFileInfo(cwd,argv);
3301     }
3302   }
3303 }
3304 func TimeValSub(tv1 *time.Duration, tv2 *time.Duration){
3305   //tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
3306   *tv1 -= *tv2;
3307 }
3308 func RusageSub(rul, ru2 [2]aRusage)((2)aRusage){
3309   TimeValSub(&rul[0].Utime,&ru2[0].Utime)
3310   TimeValSub(&rul[0].Stime,&ru2[0].Stime)
3311   TimeValSub(&rul[1].Utime,&ru2[1].Utime)
3312   TimeValSub(&rul[1].Stime,&ru2[1].Stime)
3313   return ru1
3314 }
3315 func TimeValAdd(tv1 time.Duration, tv2 time.Duration)(time.Duration){
3316   //tv1 := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
3317   tvs := tv1 + tv2;
3318   return tvs;
3319 }
3320 func RusageAdd(rul, ru2 [2]aRusage)((2)aRusage){
3321   TimeValAdd(rul[0].Utime,ru2[0].Utime)
3322   TimeValAdd(rul[0].Stime,ru2[0].Stime)
3323   TimeValAdd(rul[1].Utime,ru2[1].Utime)
3324   TimeValAdd(rul[1].Stime,ru2[1].Stime)
3325   return rul
3326 }
3327 */
3328
3329 // <a name="rusage">Resource Usage</a>
3330 func Rusagef(fmtspec string, argv []string, ru [2]aRusage)(string){
3331   // ru[0] self , ru[1] children
3332   ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3333   st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3334   su := int64(ut.Sec)*1000000 + int64((ut.Usec)) * 1000
3335   //su := (int64(st.Sec)*1000000 + int64(st.Usec)) * 1000
3336   uu := ut // in nano sec
3337   uu := st // in nano sec
3338   uu := uu + su
3339   ret := fmt.Sprintf("%v/sum", abbtme(tu))
3340   ret += fmt.Sprintf("%v/us", abbtme(uu))
3341   ret += fmt.Sprintf("%v/sys", abbtme(su))
3342   return ret
3343 }
3344 func Rusagef(fmtspec string, argv []string, ru [2]aRusage)(string{
3345   ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3346   st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3347   su := fmt.Sprintf("%d.%06ds",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3348   //fmt.Printf("%d.%06ds/%d.%06ds",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
3349   //fmt.Println("%d.%06ds/%d.%06ds",ut/1000000000,ut/1000000000)
3350   fmt.Println("%d.%06ds/%d.%06ds",st/1000000000,st/1000000000);
3351   return ret
3352 }
3353
3354 func Getrusagev()(2)aRusage{
3355   var ruv = [2]aRusage{
3356     aGetusage(aRUSAGE_SELF,&ruv[0])
3357     aGetusage(aRUSAGE_CHILDREN,&ruv[1])
3358   }
3359   return ruv
3360 }
3361 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3362   if 2 <= len(argv){

```

```

3363     gshCtx.LastUsage = aRusage{
3364         usageev := GetUsage{
3365             fin := gshCtx.gshellv(argv[1:])
3366             rusagev2 := GetUsagev()
3367             showUsage(argv[1], argv, &gshCtx.LastRusage)
3368             rusage := RusageSubv(rusagev2, rusagev1)
3369             showUsage("self", argv, &rusage[0])
3370             showUsage("child", argv, &rusage[1])
3371         }
3372     }
3373     if(usagev == nil) {
3374         aGetUsage(&RUSAGE_SELF, &rusage)
3375         showUsage("self", argv, &rusage)
3376         aGetUsage(&RUSAGE_CHILDREN, &rusage)
3377         showUsage("child", argv, &rusage)
3378     }
3379     return fin
3380 }
3381 func (gshCtx *GshContext)xJobs(argv[]string){
3382     fmt.Printf("%d Jobs\n", len(gshCtx.BackgroundJobs))
3383     for ji, pid := range gshCtx.BackgroundJobs {
3384         //wstat := syscall.WaitStatus{0}
3385         rusage := aRusage {
3386             //wpid, err := syscall.Wait4(pid, &wstat, syscall.WNOHANG, &rusage)
3387             //wpid, err := syscall.Wait4(pid, nil, syscall.WNOHANG, &rusage);
3388
3389             wpid := pid.Pid();
3390             err := errors.New("stab_NoError");
3391
3392             if err != nil {
3393                 fmt.Printf("-E-- %d (%d) (%v)\n", ji, wpid, err)
3394             }else{
3395                 fmt.Printf("%d%d\n", ji, wpid)
3396                 showUsage("chld", argv, &rusage)
3397             }
3398         }
3399     }
3400 }
3401 func (gsh*GshContext)inBackground(argv[]string)(bool){
3402     if gsh.CmdTrace { fmt.Println("--I-- inBackground(%v)\n", argv) }
3403     gsh.BackGround = true // set background option
3404     xfin := false
3405     xfin = gsh.gshellv(argv)
3406     gsh.Background = false
3407     return xfin
3408 }
3409 // -o file without command means just opening it and refer by #N
3410 // should be listed by "files" command
3411 func (gshCtx*GshContext)xOpen(argv[]string){
3412     //var pv = &int(-1,-1)
3413     //err := syscall.Pipe(pv)
3414     //fmt.Printf("-I-- pipe(%d,%d)(%v)\n", pv[0],pv[1],err)
3415     pin,pout,err := os.Pipe()
3416     pin.Fd(),pout.Fd(),err
3417 }
3418 func (gshCtx*GshContext)fromPipe(argv[]string){
3419 }
3420 func (gshCtx*GshContext)xClose(argv[]string){
3421 }
3422 // <a name="redirection">redirection</a>
3423 func (gshCtx*GshContext)redirect(argv[]string)(bool){
3424     if len(argv) < 2 {
3425         return false
3426     }
3427     cmd := argv[0]
3428     fname := argv[1]
3429     var file *os.File = nil
3430
3431     fdix := 0
3432     mode := os.O_RDONLY
3433
3434     switch {
3435     case cmd == "-i" || cmd == "<":
3436         fdix = 0
3437         mode = os.O_RDONLY
3438     case cmd == "-o" || cmd == ">":
3439         fdix = 1
3440         mode = os.O_RDWR | os.O_CREATE
3441     case cmd == "-a" || cmd == ">>":
3442         fdix = 1
3443         mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3444     }
3445     if fname[0] == '#' {
3446         fd, err := strconv.Atoi(fname[1:])
3447         if err != nil {
3448             fmt.Printf("--E-- (%v)\n",err)
3449             return false
3450         }
3451         file = os.NewFile(uintptr(fd),"MaybePipe")
3452     }else{
3453         xfile, err := os.OpenFile(argv[1], mode, 0600)
3454         if err != nil {
3455             fmt.Printf("--E-- (%s)\n",err)
3456             return false
3457         }
3458         file = xfile
3459     }
3460
3461     gshPA := gshCtx.gshPA
3462     safd := gshPA.Files[fdix]
3463     //gshPA.Files[fdix] = file.Fd()
3464     gshPA.Files[fdix] = file;
3465     fmt.Printf("-I-- Opened [%d] %s\n",file.Fd(),argv[1])
3466     gshCtx.gshellv(argv[2:])
3467     gshPA.Files[fdix] = safd
3468
3469     return false
3470 }
3471
3472 //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3473 func httpHandler(res http.ResponseWriter, req *http.Request){
3474     path := req.URL.Path
3475     fmt.Println("--I-- Got HTTP Request(%s)\n",path)
3476     {
3477         gshCbuf, i := setupGshContext()
3478         gshCtx := &gshCbuf
3479         fmt.Println("--I-- %s\n",path[1:])
3480         gshCtx.tgshell(path[1:])
3481     }
3482     fmt.Fprintf(res, "Hello(^_~)/\n%s\n",path)
3483 }
3484
3485 func (gshCtx *GshContext)httpServer(argv []string){
3486     http.HandleFunc("/", httpHandler)
3487     accport := "localhost:9999"
3488     fmt.Println("--I-- HTTP Server Start at [%s]\n",accport)
3489     http.ListenAndServe(accport,nil)
3490 }
3491 func (gshCtx *GshContext)xGo(argv[]string){
3492     go gshCtx.gshellv(argv[1]);
3493 }
3494 func (gshCtx *GshContext) xPS(argv[]string)(){
3495 }
3496 // <a name="plugin">Plugin</a>
3497 // plugin [-ls [names]] to list plugins
3498 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3499 func (gshCtx *GshContext)whichPlugin(name string,argv[]string)(pi *PluginInfo){
3500     pi = nil
3501     for i,p := range gshCtx.PluginFuncs {
3502         if p.Name == name && pi == nil {
3503             pi = &p
3504         }
3505         if lisin("s",argv) {
3506             //fmt.Printf("%v %v ",i,p)
3507             if lisin("ls",argv){
3508                 showFileInfo(p.Path,argv)
3509             }else{
3510                 fmt.Printf("%s\n",p.Name)
3511             }
3512         }
3513     }
3514     return pi
3515 }
3516 func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
3517     if len(argv) == 0 || argv[0] == "-ls" {
3518         gshCtx.whichPlugin("",argv)
3519     }
3520     return nil
3521 }
3522 name := argv[0]
3523 Pi := gshCtx.whichPlugin(name,[]string{"-s"})
3524 if Pi != nil {
3525     os.Args = argv // should be recovered?
3526     Pi.Addr.(func())()
3527     return nil
3528 }
3529 sofile := toFullPath(argv[0] + ".so") // or find it by which($PATH)
3530 P, err := plugin.Open(sofile)
3531 if err != nil {
3532     fmt.Printf("--E-- plugin.Open(%s)(%v)\n",sofile,err)
3533     return err
3534 }
3535 fname := "Main"
3536 f, err := p.Lookup(fname)
3537 if( err != nil ){
3538     fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n",fname,err)
3539     return err
3540 }

```

```

3540 }
3541 pin := PluginInfo {p,f,name,sofile}
3542 gshCtx.PluginFuncs.append(gshCtx.PluginFuncs,pin)
3543 fmt.Printf("-- added (%d)\n",len(gshCtx.PluginFuncs))
3544 //fmt.Printf("-- first call(%s:%s)%v\n",sofile, fname, argv)
3545 os.Args = argv
3546 f.Func()()
3547 return err
3548 }
3549 func (gshCtx *GshContext)Args(argv[]string){
3550 for i,v := range os.Args {
3551 fmt.Printf("[%v] %v\n",i,v)
3552 }
3553 }
3554 func (gshCtx *GshContext) showVersion(argv[]string){
3555 if isin("-l",argv) {
3556 fmt.Printf("%v/%v (%v)",NAME,VERSION,DATE);
3557 }else{
3558 fmt.Printf("%v",VERSION);
3559 }
3560 if isin("-a",argv) {
3561 fmt.Printf("%s",AUTHOR)
3562 }
3563 if !isin("-n",argv) {
3564 fmt.Printf("\n")
3565 }
3566 }
3567 }
3568 // <a name="scanf">Scanf</a> // string decomposer
3569 // scanf [1] input
3570 func scanUtil(string)(strv]string{
3571 strv = strings.Split(sstr," ")
3572 return strv
3573 }
3574 func scanUntil(src,end string)(rstr string,len int){
3575 idx := strings.Index(src,end)
3576 if 0 <= idx {
3577 rstr = src[0:idx]
3578 return rstr,idx+1
3579 }
3580 return src,0
3581 }
3582 }
3583 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3584 func (gsh*GshContext)printVal(fmts string, vstr string, optv[]string){
3585 //vint,err := strconv.Atoi(vstr)
3586 //original int64 = 0
3587 n := 0
3588 err := error(nil)
3589 if strBegins(vstr, " ") {
3590 vx,_ := strconv.Atoi(vstr[1:])
3591 if vx < len(gsh.iValues) {
3592 vstr = gsh.iValues[vx]
3593 }else{
3594 }
3595 }
3596 // should use Eval()
3597 if strBegins(vstr,"0x") {
3598 vx,_ = fmt.Sscanf(vstr[2:],"%x",&ival)
3599 }else{
3600 n,err = fmt.Sscanf(vstr,"%d",&ival)
3601 }
3602 //fmt.Printf("-- n=%d err=(%v) %s=%v\n",n,err,vstr, ival)
3603 if n == 1 && err == nil {
3604 //fmt.Printf("-- show formatn(%v) ival(%v)\n",fmts,ival)
3605 fmt.Printf("%s"+fmts,ival)
3606 }else{
3607 if isin("-bn",optv){
3608 fmt.Println("%s"+fmts,filepath.Base(vstr))
3609 }else{
3610 fmt.Printf("%s"+fmts,vstr)
3611 }
3612 }
3613 }
3614 }
3615 func (gsh*GshContext)printf(fmts,div string,argv[]string,optv[]string,list[]string){
3616 //fmt.Printf("%d",len(list))
3617 //curfmt := "%"
3618 outlen := 0
3619 curfmt := gsh.iFormat
3620 if 0 < len(fmts) {
3621 for xi := 0; xi < len(fmts); xi++ {
3622 fch := fmts[xi]
3623 if fch == '%' {
3624 if xi+1 < len(fmts) {
3625 curfmt = string(fmts[xi+1])
3626 xi += 1
3627 if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3628 vals,len := scanUntil(fmts[xi+2:],")")
3629 //fmt.Printf("-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,len)
3630 gsh.printVal(curfmt,vals,optv)
3631 xi += 2+length-1
3632 outlen += 1
3633 }
3634 continue;
3635 }
3636 if fch == ' ' {
3637 hi,len := scanInt(fmts[xi+1:])
3638 if 0 < len {
3639 if hi < len(gsh.iValues) {
3640 gsh.printVal(curfmt,gsh.iValues[hi],optv)
3641 outlen += 1 // should be the real length
3642 }else{
3643 fmt.Printf("(out-range)")
3644 }
3645 xi += len
3646 continue;
3647 }
3648 fmt.Printf("%c",fch)
3649 outlen += 1
3650 }
3651 }else{
3652 //fmt.Printf("-- print (%s)\n")
3653 for i,v := range list {
3654 if 0 < i {
3655 fmt.Println(div)
3656 }
3657 gsh.printVal(curfmt,v,optv)
3658 outlen += 1
3659 }
3660 if 0 < outlen {
3661 fmt.Printf("\n")
3662 }
3663 }
3664 }
3665 if 0 < outlen {
3666 fmt.Printf("\n")
3667 }
3668 }
3669 func (gsh*GshContext)Scanv(argv[]string){
3670 //fmt.Printf("-- Scanv(%v)\n",argv)
3671 if len(argv) == 1 {
3672 return
3673 }
3674 argv = argv[1:]
3675 fmts := ""
3676 if strBegins(argv[0],"-F") {
3677 fmts = argv[0]
3678 gsh.iDelimiter = fmts
3679 argv = argv[1:]
3680 }
3681 input := strings.Join(argv," ")
3682 if fmts == "" { // simple decomposition
3683 v := scanUtil(input)
3684 gsh.iValues[v]
3685 //fmt.Printf("%v\n",strings.Join(v,""))
3686 }else{
3687 v := make([]string,8)
3688 n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
3689 fmt.Printf("-- Scanf ->(%v) n=%d err=(%v)\n",v,n,err)
3690 gsh.iValues = v
3691 }
3692 }
3693 func (gsh*GshContext)Printv(argv[]string){
3694 if false //##0
3695 fmt.Printf("%v\n",strings.Join(argv[1:]," "))
3696 return
3697 }
3698 //fmt.Printf("-- Printv(%v)\n",argv)
3699 //fmt.Printf("%v\n",strings.Join(gsh.iValues,""))
3700 gsh.iDelimiter
3701 fmts := argv[1:]
3702 argv = argv[2:]
3703 if 0 < len(argv) {
3704 if strBegins(argv[0],"-F") {
3705 div = argv[2];
3706 argv = argv[1:]
3707 }
3708 }
3709 optv := []string{}
3710 for ;v := range argv {
3711 if strBegins(v,"-"){
3712 optv = append(optv,v)
3713 argv = argv[1:]
3714 }else{
3715 break;
3716 }
3717 }
3718 }
3719 }
3720 }
3721 }
3722 }
3723 }
3724 }
3725 }
3726 }
3727 }
3728 }
3729 }
3730 }
3731 }
3732 }
3733 }
3734 }
3735 }
3736 }
3737 }
3738 }
3739 }
3740 }
3741 }
3742 }
3743 }
3744 }
3745 }
3746 }
3747 }
3748 }
3749 }
3750 }
3751 }
3752 }
3753 }
3754 }
3755 }
3756 }
3757 }
3758 }
3759 }
3760 }
3761 }
3762 }
3763 }
3764 }
3765 }
3766 }
3767 }
3768 }
3769 }
3770 }
3771 }
3772 }
3773 }
3774 }
3775 }
3776 }
3777 }
3778 }
3779 }
3780 }
3781 }
3782 }
3783 }
3784 }
3785 }
3786 }
3787 }
3788 }
3789 }
3790 }
3791 }
3792 }
3793 }
3794 }
3795 }
3796 }
3797 }
3798 }
3799 }
3800 }
3801 }
3802 }
3803 }
3804 }
3805 }
3806 }
3807 }
3808 }
3809 }
3810 }
3811 }
3812 }
3813 }
3814 }
3815 }
3816 }
3817 }
3818 }
3819 }
3820 }
3821 }
3822 }
3823 }
3824 }
3825 }
3826 }
3827 }
3828 }
3829 }
3830 }
3831 }
3832 }
3833 }
3834 }
3835 }
3836 }
3837 }
3838 }
3839 }
3840 }
3841 }
3842 }
3843 }
3844 }
3845 }
3846 }
3847 }
3848 }
3849 }
3850 }
3851 }
3852 }
3853 }
3854 }
3855 }
3856 }
3857 }
3858 }
3859 }
3860 }
3861 }
3862 }
3863 }
3864 }
3865 }
3866 }
3867 }
3868 }
3869 }
3870 }
3871 }
3872 }
3873 }
3874 }
3875 }
3876 }
3877 }
3878 }
3879 }
3880 }
3881 }
3882 }
3883 }
3884 }
3885 }
3886 }
3887 }
3888 }
3889 }
3890 }
3891 }
3892 }
3893 }
3894 }
3895 }
3896 }
3897 }
3898 }
3899 }
3900 }
3901 }
3902 }
3903 }
3904 }
3905 }
3906 }
3907 }
3908 }
3909 }
3910 }
3911 }
3912 }
3913 }
3914 }
3915 }
3916 }
3917 }
3918 }
3919 }
3920 }
3921 }
3922 }
3923 }
3924 }
3925 }
3926 }
3927 }
3928 }
3929 }
3930 }
3931 }
3932 }
3933 }
3934 }
3935 }
3936 }
3937 }
3938 }
3939 }
3940 }
3941 }
3942 }
3943 }
3944 }
3945 }
3946 }
3947 }
3948 }
3949 }
3950 }
3951 }
3952 }
3953 }
3954 }
3955 }
3956 }
3957 }
3958 }
3959 }
3960 }
3961 }
3962 }
3963 }
3964 }
3965 }
3966 }
3967 }
3968 }
3969 }
3970 }
3971 }
3972 }
3973 }
3974 }
3975 }
3976 }
3977 }
3978 }
3979 }
3980 }
3981 }
3982 }
3983 }
3984 }
3985 }
3986 }
3987 }
3988 }
3989 }
3990 }
3991 }
3992 }
3993 }
3994 }
3995 }
3996 }
3997 }
3998 }
3999 }
3999 }
```

```

3717     }
3718   } if 0 < len(argv) {
3719     fmts = strings.Join(argv, " ")
3720   } gsh.Printf(fmts,div,argv,optv,gsh.iValues)
3721 }
3722 func (gsh*GshContext)Basename(argv[]string){
3723   for i,v := range gsh.iValues{
3724     gsh.iValues[i] = filepath.Base(v)
3725   }
3726 }
3727 func (gsh*GshContext)Sortv(argv[]string){
3728   sv := gsh.iValues
3729   sort.Slice(sv, func(i,j int) bool {
3730     return sv[i] < sv[j]
3731   })
3732 }
3733 func (gsh*GshContext)Shiftv(argv[]string){
3734   w := len(gsh.iValues)
3735   if 0 < w {
3736     if isin("-r",argv) {
3737       top := gsh.iValues[0]
3738       gsh.iValues = append(gsh.iValues[1:],top)
3739     }else{
3740       gsh.iValues = gsh.iValues[1:]
3741     }
3742   }
3743 }
3744 }
3745 func (gsh*GshContext)Eqv(argv[]string){
3746 }
3747 func (gsh*GshContext)Dgv(argv[]string){
3748 }
3749 func (gsh*GshContext)Push(argv[]string){
3750   gsh.iValStack = append(gsh.iValStack,argv[1:])
3751   fmt.Printf("depth=%d\n",len(gsh.iValStack))
3752 }
3753 func (gsh*GshContext)Dump(argv[]string){
3754   for i,v := range gsh.iValStack {
3755     fmt.Printf("%d %v\n",i,v)
3756   }
3757 }
3758 func (gsh*GshContext)Pop(argv[]string){
3759   depth := len(gsh.iValStack)
3760   if 0 < depth {
3761     v := gsh.iValStack[depth-1]
3762     if v == "-cat",argv){
3763       gsh.iValues = append(gsh.iValues,v...)
3764     }else{
3765       gsh.iValues = v
3766     }
3767   }
3768   gsh.iValStack = gsh.iValStack[0:depth-1]
3769   fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3770 }else{
3771   fmt.Printf("depth=%d\n",depth)
3772 }
3773 }
3774 }
3775 // <a name="interpreter">Command Interpreter</a>
3776 func (gshCtxt*GshContext)gshellv(argv []string) (fin bool) {
3777   fin = false
3778
3779   if gshCtxt.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3780   if len(argv) < 0 {
3781     return false
3782   }
3783   argv := []string{}
3784   for ai := 0; ai < len(argv); ai++ {
3785     argv = append(argv,strsubst(gshCtxt,argv[ai],false))
3786   }
3787   argv = argv
3788   if false {
3789     for ai := 0; ai < len(argv); ai++ {
3790       fmt.Printf("%d %s [%d]\n",
3791                   ai,argv[ai],len(argv[ai]),argv[ai])
3792     }
3793   }
3794   cmd := argv[0]
3795   if gshCtxt.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)%v\n",len(argv),argv) }
3796   switch { // https://tour.golang.org/flowcontrol/11
3797   case cmd == "x":
3798     gshCtxt.xPwd([]string{}); // empty command
3799   case cmd == "-x":
3800     gshCtxt.CmdTrace = ! gshCtxt.CmdTrace
3801   case cmd == "-xt":
3802     gshCtxt.CmdTime = ! gshCtxt.CmdTime
3803   case cmd == "-ot":
3804     gshCtxt.sconnect(true, argv)
3805   case cmd == "-ou":
3806     gshCtxt.sconnect(false, argv)
3807   case cmd == "-it":
3808     gshCtxt.saccept(true , argv)
3809   case cmd == "-iu":
3810     gshCtxt.saccept(false, argv)
3811   case cmd == "-i" || cmd == "<":
3812     gshCtxt.redirect(argv)
3813   case cmd == "|":
3814     gshCtxt.xfrPipe(argv)
3815   case cmd == "args":
3816     gshCtxt.Args(argv)
3817   case cmd == "bg" || cmd == "-bg":
3818     rfd := gshCtxt.inBackground(argv[1:])
3819     readFD(rfd)
3820   case cmd == "-bn":
3821     gshCtxt.Basename(argv)
3822   case cmd == "call":
3823     _ = gshCtxt.excommand(false,argv[1:])
3824   case cmd == "cd" || cmd == "chdir":
3825     gshCtxt.XCDir(argv);
3826   case cmd == "-cksum":
3827     gshCtxt.XFind(argv)
3828   case cmd == "-sum":
3829     gshCtxt.XFind(argv)
3830   case cmd == "-sumtest":
3831     _ = ""
3832   if l < len(argv) { str = argv[l] }
3833   ifrc := strCRC32(str,uint64(len(str)))
3834   printf(stderr,"%v\n",crc,len(str))
3835   case cmd == "close":
3836     gshCtxt.XClose(argv)
3837   case cmd == "copy":
3838     gshCtxt.FileCopy(argv)
3839   case cmd == "dec" || cmd == "decode":
3840     gshCtxt.Dec(argv)
3841   case cmd == "#define":
3842     case cmd == "dic" || cmd == "d":
3843       xdic(argv)
3844     case cmd == "dump":
3845       gshCtxt.Dump(argv)
3846   case cmd == "echo" || cmd == "e":
3847     echo(argv,true)
3848   case cmd == "enc" || cmd == "encode":
3849     gshCtxt.Enc(argv)
3850   case cmd == "env":
3851     env(argv)
3852   case cmd == "eval":
3853     xeval(argv[1:true])
3854   case cmd == "ev" || cmd == "events":
3855     dumpEvents(argv)
3856   case cmd == "exec":
3857     _ = gshCtxt.excommand(true,argv[1:])
3858   /* should not return here */
3859   case cmd == "quit" || cmd == "exit":
3860     // write Result code EXIT to >
3861     return true
3862   case cmd == "fdls":
3863     // dump the attributes of fds (of other process)
3864   case cmd == "find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3865     gshCtxt.XFind(argv[1:])
3866   case cmd == "fu":
3867     gshCtxt.XFind(argv[1:])
3868   case cmd == "fork":
3869     /* mainly for a server */
3870   case cmd == "go":
3871     gshCtxt.Go()
3872   case cmd == "go":
3873     gshCtxt.XGo(argv)
3874   case cmd == "grep":
3875     gshCtxt.XFind(argv)
3876   case cmd == "grep":
3877     gshCtxt.XFind(argv)
3878   case cmd == "grep":
3879     gshCtxt.Del(argv)
3880   case cmd == "genq":
3881     gshCtxt.Genq(argv)
3882   case cmd == "gpop":
3883     gshCtxt.Pop(argv)
3884   case cmd == "hi":
3885     gshCtxt.Push(argv)
3886   case cmd == "history" || cmd == "hi": // hi should be alias
3887     gshCtxt.XHistory(argv)
3888   case cmd == "jobs":
3889     gshCtxt.Jobs(argv)
3890   case cmd == "lisp" || cmd == "nisp":
3891     gshCtxt.Splitline(argv)
3892   case cmd == "-ls":
3893     gshCtxt.XFind(argv)
3894   case cmd == "nop":
3895

```

```

3894     // do nothing
3895     case cmd == "xopen":
3896         gshCtx.XOpen(argv)
3897     case cmd == "plug" || cmd == "plugin" || cmd == "pin":
3898         gshCtx.XPlugin(argv[1:])
3899     case cmd == "print" || cmd == "-pr":
3900         // output internal slice // also sprintf should be
3901         gshCtx.Println(argv)
3902     case cmd == "ps":
3903         gshCtx.XPs(argv)
3904     case cmd == "pstitle":
3905         // to be gsh.title
3906     case cmd == "rexcd" || cmd == "rexd":
3907         gshCtx.RexecServer(argv)
3908     case cmd == "rexec" || cmd == "rex":
3909         gshCtx.RexecClient(argv)
3910     case cmd == "repeat" || cmd == "rep": // repeat cond command
3911         gshCtx.repeat(argv)
3912     case cmd == "replay":
3913         gshCtx.XReplay(argv)
3914     case cmd == "scan":
3915         // scan input (or so in fscanf) to internal slice (like Files or map)
3916         gshCtx.Scan(argv)
3917     case cmd == "set":
3918         set name
3919     case cmd == "serv":
3920         gshCtx.HttpServer(argv)
3921     case cmd == "shift":
3922         gshCtx.Shift(argv)
3923     case cmd == "sleep":
3924         gshCtx.Sleep(argv)
3925     case cmd == "sort":
3926         gshCtx.Sort(argv)
3927
3928     case cmd == "j" || cmd == "join":
3929         gshCtx.RJoin(argv)
3930     case cmd == "a" || cmd == "alpa":
3931         gshCtx.Rexec(argv)
3932     case cmd == "jcd" || cmd == "jchdir":
3933         gshCtx.Rchdir(argv)
3934     case cmd == "jget":
3935         gshCtx.RGet(argv)
3936     case cmd == "jls":
3937         gshCtx.RLs(argv)
3938     case cmd == "jput":
3939         gshCtx.RPut(argv)
3940     case cmd == "jpwd":
3941         gshCtx.RPwd(argv)
3942
3943     case cmd == "time":
3944         fin = gshCtx.XTime(argv)
3945     case cmd == "ungets":
3946         if i < len(argv) {
3947             ungets(argv[i]+"\n")
3948         } else{
3949         }
3950     case cmd == "pwd":
3951         gshCtx.XPwd(argv);
3952     case cmd == "show" || cmd == "-ver" || cmd == "version":
3953         gshCtx.ShowVersion(argv)
3954     case cmd == "where":
3955         // data file or so?
3956     case cmd == "which":
3957         which(PATH, argv)
3958     case cmd == "g" && i < len(argv) && argv[i] == "listen":
3959         go g.Server(argv[1:])
3960     case cmd == "g" && i < len(argv) && argv[i] == "serve":
3961         go g.Server(argv[1:])
3962     case cmd == "g" && i < len(argv) && argv[i] == "join":
3963         go g.Join(argv[1:]);
3964     case cmd == "sj":
3965         isend(argv);
3966     case cmd == "jsend":
3967         jsend(argv);
3968     default:
3969         if gshCtx.WhichPlugin(cmd, []string{"-a"}) != nil {
3970             gshCtx.XPlugin(argv)
3971         } else{
3972             notfound, := gshCtx.ExCommand(false, argv)
3973             if notfound {
3974                 fmt.Printf("--E-- command not found (%v)\n", cmd)
3975             }
3976         }
3977     }
3978     return fin
3979 }
3980
3981 func (gsh*GshContext)gshell(gline string) (rfin bool) {
3982     argv := strings.Split(string(gline), " ")
3983     fin := gsh.gshell(argv)
3984     return fin
3985 }
3986
3987 func (gsh*GshContext)tgshell(gline string)(xfin bool){
3988     start := time.Now()
3989     fin := gsh.gshell(gline)
3990     end := time.Now()
3991     elps := end.Sub(start)
3992     if gsh.CmdTime {
3993         fmt.Printf("--T-- %s + time.Now().Format(time.Stamp) + "(%d.%09ds)\n",
3994             elps/1000000000,elps/1000000000)
3995     }
3996     return fin
3997 }
3998
3999 func Ttyid() (int) {
4000     fi, err := os.Stdin.Stat()
4001     if err != nil {
4002         return 0;
4003     }
4004     //fmt.Printf("Stdin: %v Dev=%d\n",
4005     // 0 if it's a terminal, Mode is 0x80000000 if it's a device)
4006     if (fi.Mode() & os.ModeDevice) != 0 {
4007         stat := aStat_t();
4008         err := aFstat(0,&stat)
4009         if err != nil {
4010             //fmt.Println("--I-- Stdin: (%v)\n",err)
4011         }else{
4012             // stat.Rdev&0xFF,stat.RDev;
4013             //fmt.Printf("--I-- Stdin: rdev=%d %d\n",
4014             // 0 if it's a terminal, Mode is 0x80000000 if it's a device)
4015             return int(stat.Rdev & 0xFF)
4016         }
4017     }
4018     return 0
4019 }
4020
4021 func (gshCtx *GshContext) ttyfile() string {
4022     //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
4023     //fmt.Println("gshctx.GshHomeDir = ",gshCtx.GshHomeDir)
4024     //fmt.Sprintf("%03d",gshCtx.TerminalId)
4025     //strconv.Itoa(gshCtx.TerminalId)
4026     //fmt.Printf("--I-- ttyfile=%s\n",ttyfile)
4027     return ttyfile
4028 }
4029
4030 func (gshCtx *GshContext) ttysize()(*os.File){
4031     file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4032     if err != nil {
4033         fmt.Printf("--F-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
4034     }
4035     return file;
4036 }
4037
4038 func (gshCtx *GshContext) getline(hix int, skipping bool, prevline string) (string) {
4039     if( skipping ){
4040         reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4041         line,_ := reader.ReadLine()
4042         return string(line)
4043     }else{
4044         if true {
4045             return xgetline(hix,prevline,gshCtx)
4046         }
4047     }
4048     if( with_exgetline && gshCtx.GetLine != "" ){
4049         //var xhix int64 = int64(hix); // cast
4050         newenv := os.Getenv()
4051         newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
4052
4053         tty := gshCtx.ttyfile()
4054         tty.WriteString(prevline)
4055         Pa := os.ProcAttr {
4056             "", // start dir
4057             newenv, //os.Getenv(),
4058             {os.File(os.Stdin,os.Stdout,os.Stderr,tty}, // args
4059             nil,
4060         }
4061         //fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.GetLine)
4062         proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
4063         if err != nil {
4064             fmt.Printf("--F-- getline process error (%v)\n",err)
4065             // for ; ; {
4066             //     return "exit (getline program failed)"
4067         }
4068         //stat, err := proc.Wait()
4069         proc.Wait()
4070         buff := make([]byte,LINESIZE)
4071         count, err := tty.Read(buff)
4072         //_, err = tty.Read(buff)
4073         //fmt.Printf("--D-- getline (%d)\n",count)
4074     }
4075 }

```

```

4071     if err != nil {
4072         if !(count == 0) { // && err.String() == "EOF" }
4073             fmt.Printf("--E-- getline error (%s)\n",err)
4074     }
4075 }else{
4076     //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
4077     tpy.Close()
4078     gline := string(buff[0:count])
4079     return gline
4080 }
4081 else
4082 */
4083 {
4084     // if isatty {
4085     fmt.Printf("!%d",hix)
4086     fmt.Print(PROMPT)
4087     //}
4088     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4089     line,_ := reader.ReadLine()
4090     return string(line)
4091 }
4092 }
4093 //== begin ===== getline ===== getline
4094 /*
4095 * getline.c
4096 * 2020-0819 extracted from dog.c
4097 * getline.go
4098 * 2020-0822 ported to Go
4099 */
4100
4101 package main // getline main
4102 import (
4103     "fmt"          // <a href="https://golang.org/pkg/fmt/">fmt</a>
4104     "strings"       // <a href="https://golang.org/pkg/strings/">strings</a>
4105     "os"           // <a href="https://golang.org/pkg/os/">os</a>
4106     "syscall"      // <a href="https://golang.org/pkg/syscall/">syscall</a>
4107     "/bytes"        // <a href="https://golang.org/pkg/bytes/">bytes</a>
4108     "/os/exec"     // <a href="https://golang.org/pkg/os/exec/">os/exec</a>
4109     "/os"           // <a href="https://golang.org/pkg/os/">os</a>
4110 )
4111 */
4112 // C language compatibility functions
4113 var errno = 0
4114 var stdin *os.File = os.Stdin
4115 var stdout *os.File = os.Stdout
4116 var stderr *os.File = os.Stderr
4117 var NULL = nil
4118 var NULLP = 0
4119 //var LINESIZE = 1024
4120
4121 func system(cmdstr string)(int){
4122     /PA := syscall.ProcAttr{
4123     PA := os.ProcAttr {
4124         Path: cmdstr,
4125         Dir:  os.Getenv("HOME"),
4126         FdInherit: true,
4127         Env:  []os.File{os.Stdin,os.Stdout,os.Stderr},
4128         EnvSize: 0,
4129     },
4130     argv := strings.Split(cmdstr," ")
4131     //pid,err := syscall.ForkExec(argv[0],argv,&PA)
4132     proc,err := os.StartProcess(argv[0],argv,&PA);
4133     if( err != nil ){
4134         //fmt.Printf("--E-- system(%v)\n",cmdstr,err);
4135         return -1;
4136     }
4137     pstat, _ := proc.Wait();
4138     pid := pstat.Pid();
4139     if( err != nil ){
4140         fmt.Printf("--E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
4141     }
4142     //syscall.Wait4(pid,nil,0,nil)
4143     //fmt.Printf("====E== pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
4144 }
4145 /*
4146 argv := strings.Split(cmdstr," ")
4147 if,err := os.Stderr.--I-- system(%v)\n",argv)
4148 //cmd := exec.Command(argv[0],...)
4149 cmd := exec.Command(argv[0],argv[1],argv[2])
4150 cmd.Stdin = strings.NewReader("output of system")
4151 var out bytes.Buffer
4152 cmd.Stdout = &out
4153 cmd.Stderr = &err
4154 err := cmd.Run()
4155 if err != nil {
4156     fmt.Printf(os.Stderr.--E-- system(%v)err(%v)\n",argv,err)
4157     fmt.Println("ERR:%s",err.String())
4158 }else{
4159     fmt.Printf("%s",out.String())
4160 }
4161 */
4162 return 0
4163 }
4164 func atoi(str string)(ret int){
4165     ret,err := fmt.Sscanf(str,"%d",ret)
4166     if err == nil {
4167         return ret
4168     }
4169     // should set errno
4170     return 0
4171 }
4172 func getenv(name string)(string){
4173     val,ok := os.LookupEnv(name)
4174     if ok {
4175         return val
4176     }else{
4177         return "?"
4178     }
4179 }
4180 func strcpy(dst StrBuff, src string){
4181     var i int
4182     srcb := []byte(src)
4183     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4184         dst[i] = srcb[i]
4185     }
4186     dst[i] = 0
4187 }
4188 func xstrcpy(dst StrBuff, src StrBuff){
4189     dst = src
4190 }
4191 func strcat(dst StrBuff, src StrBuff){
4192     dst = append(dst,src...)
4193 }
4194 func strupr(str StrBuff)(string){
4195     return string(str[0:strlen(str)])
4196 }
4197 func strlen(str StrBuff)(int){
4198     return len(str)
4199 }
4200 func strlen(str StrBuff)(int){
4201     var i int
4202     for i = 0; i < len(str) && str[i] != 0; i++ {
4203     }
4204     return i
4205 }
4206 func sizeof(data StrBuff)(int){
4207     return len(data)
4208 }
4209 func isatty(fd int)(ret int){
4210     return 1
4211 }
4212
4213 func fopen(file string,mode string)(fp*os.File){
4214     if mode == "r"{
4215         fp,err := os.Open(file)
4216         if( err != nil ){
4217             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
4218             return NULL_FPP;
4219         }
4220         return fp;
4221     }
4222     else{
4223         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4224         if( err != nil ){
4225             return NULL_FPP;
4226         }
4227         return fp;
4228     }
4229 }
4230 func fclose(fp*os.File){
4231     fp.Close()
4232 }
4233 func fflush(fp *os.File)(int){
4234     return 0
4235 }
4236 func fgetc(fp*os.File)(int){
4237     var buf [1]byte
4238     ,err := fp.Read(buf[0:1])
4239     if( err != nil ){
4240         return EOF;
4241     }
4242 }

```

```

4248     return int(buf[0])
4249   }
4250 }
4251 func sfgets(str*string, size int, fp*os.File)(int){
4252   buf := make(StrBuff,size)
4253   var ch int
4254   var i int
4255   for i = 0; i < len(buf)-1; i++ {
4256     ch = fgetc(fp)
4257     //fprint(stdecr,"--fgets %d/%d %X\n",i,len(buf),ch)
4258     if( ch == EOF ) {
4259       break;
4260     }
4261     buf[i] = byte(ch);
4262     if( ch == '\n' ){
4263       break;
4264     }
4265   }
4266   buf[i] = 0
4267   //fprint(stdecr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4268   return i
4269 }
4270 func fgets(buf StrBuff, size int, fp*os.File)(int){
4271   var ch int
4272   var i int
4273   for i = 0; i < len(buf)-1; i++ {
4274     ch = fgetc(fp)
4275     //fprint(stdecr,"--fgets %d/%d %X\n",i,len(buf),ch)
4276     if( ch == EOF ) {
4277       break;
4278     }
4279     buf[i] = byte(ch);
4280     if( ch == '\n' ){
4281       break;
4282     }
4283   }
4284   buf[i] = 0
4285   //fprint(stdecr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4286   return i
4287 }
4288 func fpwrite(ch int , fp*os.File)(int){
4289   var buf [1]byte
4290   buf[0] = byte(ch)
4291   fp.WriteString(buf[0:1])
4292   return 0
4293 }
4294 func fputs(buf StrBuff, fp*os.File)(int){
4295   fp.WriteString(buf)
4296   return 0
4297 }
4298 func xfputs(str string, fp*os.File)(int){
4299   return fputs([]byte(str),fp)
4300 }
4301 func sscanf(str StrBuff,fmts string, params ...interface{})(int{
4302   fmt.Sscanf(string(str[0:strlen(str)]),fmts,params...)
4303   return 0
4304 }
4305 func fprintf(fp*os.File,fmts string, params ...interface{})(int{
4306   fmt.Fprintf(fp,fmts,params...)
4307   return 0
4308 }
4309 // <a name="IME">Command Line IME</a>
4310 // -----
4311 // -----
4312 var MyIMERVER = "MyIMB/0.0.2";
4313 type Romkana struct {
4314   dic string // dictionary ID
4315   pat string // input pattern
4316   out string // output pattern
4317   hit int64 // count of hit and used
4318 }
4319 var dicents = 0
4320 var romkana [1024]Romkana
4321 var Romkana []Romkana
4322
4323 func isinDic(str string)(int){
4324   for i,v := range Romkan {
4325     if v.pat == str {
4326       return i
4327     }
4328   }
4329   return -1
4330 }
4331 const {
4332   DIC_COM_LOAD = "im"
4333   DIC_COM_DUMP = "s"
4334   DIC_COM_LIST = "ls"
4335   DIC_COM_EMA = "en"
4336   DIC_COM_DIS = "di"
4337 }
4338 func helpDic(argv []string{
4339   out := stdecr
4340   cmd := argv[0]
4341   if 0 < len(argv) { cmd = argv[0] }
4342   fprintf(out,... `v Usage\n`,cmd)
4343   fprintf(out,... `Commands\n`)
4344   fprintf(out,... `  v -v [dicName] [dicURL] -- Import dictionary\n`,cmd,DIC_COM_LOAD)
4345   fprintf(out,... `  v -v -v [dicName] -- Search in dictionary\n`,cmd,DIC_COM_DUMP)
4346   fprintf(out,... `  v -v -v [dicName] -- List dictionary\n`,cmd,DIC_COM_LIST)
4347   fprintf(out,... `  v -v [dicName] -- Disable dictionaries\n`,cmd,DIC_COM_DIS)
4348   fprintf(out,... `  v -v -v [dicName] -- Enable dictionaries\n`,cmd,DIC_COM_EMA)
4349   fprintf(out,... ` Keys ... %v\n`,"ESC can be used for '\\\'")
4350   fprintf(out,... ` \\\\c -- Reverse the case of the last character\n`,)
4351   fprintf(out,... ` \\\\N -- Normal input to translated text\n`,)
4352   fprintf(out,... ` \\\\j -- On/Off translation mode\n`,)
4353   fprintf(out,... ` \\\\l -- Force Lower Case\n`,)
4354   fprintf(out,... ` \\\\L -- Force Upper Case (software CapsLock)\n`,)
4355   fprintf(out,... ` \\\\v -- Show translation actions\n`,)
4356   fprintf(out,... ` \\\\x -- Replace the last input character with it Hexa-Decimal\n`,)
4357 }
4358 func xbic(argv[]string{
4359   if len(argv) <= 1 {
4360     helpic(argv)
4361     return
4362   }
4363   argv = argv[1:]
4364   var debug = false
4365   var info = false
4366   var silent = false
4367   var dum = false
4368   var builtin = false
4369   com := argv[0]
4370   argv = argv[1:]
4371   opt := ""
4372   arg := ""
4373
4374   if 0 < len(argv) {
4375     arg1 := argv[0]
4376     if argv[0] == '-' {
4377       switch argv[1] {
4378         default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4379         return
4380       case "-b": builtin = true
4381       case "-d": debug = true
4382       case "-s": silent = true
4383       case "-v": info = true
4384     }
4385     opt = arg1
4386     argv = argv[1:]
4387   }
4388 }
4389 dicName := ""
4390 dicURL := ""
4391 if 0 < len(argv) {
4392   arg = argv[0]
4393   dicName = arg
4394   argv = argv[1:]
4395 }
4396 if 0 < len(argv) {
4397   dicURL = argv[0]
4398   argv = argv[1:]
4399 }
4400 if false {
4401   fprintf(stdecr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4402 }
4403 if cmd == DIC_COM_LOAD {
4404   //dictype := ""
4405   dictBody := ""
4406   if !builtin && dicName != "" && dicURL == "" {
4407     f,err := os.Open(dicName)
4408     if err != nil {
4409       if err == nil {
4410         dicURL = dicName
4411       }else{
4412         f,err = os.Open(dicName+".html")
4413         if err != nil {
4414           dicURL = dicName+".html"
4415         }else{
4416           f,err = os.Open("gshdic-"+dicName+".html")
4417           if err == nil {
4418             dicURL = "gshdic-"+dicName+".html"
4419           }
4420         }
4421     }
4422   }
4423   if err == nil {
4424     var buf = make([]byte,128*1024)
4425     count,err := f.Read(buf)

```

```

4425         f.Close()
4426         if info {
4427             fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
4428         }
4429         dicBody = string(buf[0:count])
4430     }
4431     if dicBody == "" {
4432         switch arg {
4433             default:
4434                 dicName = "WorldDic"
4435                 dicURL = Worlddic
4436                 if info {
4437                     fprintf(stderr,"--Id-- default dictionary `\"%v`\\n",
4438                           dicName);
4439                 }
4440             case "wnn":
4441                 dicName = "WnnDic"
4442                 dicURL = WnnDic
4443             case "sumomo":
4444                 dicName = "SumomoDic"
4445                 dicURL = Sumomodic
4446             case "sijimi":
4447                 dicName = "SijimiDic"
4448                 dicURL = Sijimidic
4449             case "jkl":
4450                 dicName = "JKLJDic"
4451                 dicURL = JA_JKLDic
4452         }
4453     if debug {
4454         fprintf(stderr, "--Id-- %v URL=%v\\n\\n",dicName,dicURL);
4455     }
4456     dicv := strings.Split(dicURL,",")
4457     if debug {
4458         fprintf(stderr,"--Id-- %v encoded data...\\n",dicName)
4459         fprintf(stderr,"Type: %v\\n",dicv[0])
4460         fprintf(stderr,"Body: %v\\n",dicv[1])
4461         fprintf(stderr,'\\n')
4462     }
4463     body,_ := base64.StdEncoding.DecodeString(dicv[1])
4464     dicBody = string(body)
4465 }
4466 if info {
4467     fmt.Printf("--Id-- %v %v\\n",dicName,dicURL)
4468     fmt.Println("%s\\n",dicBody)
4469 }
4470 if debug {
4471     fprintf(stderr,"--Id-- dicName %v text...\\n",dicName)
4472     fprintf(stderr,"%v\\n",string(dicBody))
4473 }
4474 envt := strings.Split(dicBody,"\\n");
4475 if info {
4476     fprintf(stderr,"--Id-- %v scan...\\n",dicName);
4477 }
4478 var added int = 0
4479 var dup int = 0
4480 for i,v := range envt {
4481     var pat string
4482     var out string
4483     if !utf8.ValidString(v) {
4484         if len(pat) <= 0 {
4485             continue
4486         } else{
4487             if 0 <= isinDic(pat) {
4488                 dup += 1
4489             }
4490         }
4491     romkanadicents = RomKana{dicName,pat,out,0}
4492     dicents += 1
4493     added += 1
4494     Romkan = append(Romkan,RomKana{dicName,pat,out,0})
4495     if debug {
4496         fmt.Printf("[32v]{32v}+8v {32v}{8v}\\n",
4497           i,len(pat),pat,len(out),out)
4498     }
4499 }
4500 if !silent {
4501     url := dicURL
4502     if strBegins(url,"data:") {
4503         url = "builtin"
4504     }
4505     fprintf(stderr,"--Id-- %v scan... %v added, %v dup. / %v total (%v)\\n",
4506             dicName,added,dup,len(Romkan),url);
4507 }
4508 // should sort by pattern length for concrete match, for performance
4509 if debug {
4510     arg = "" // search pattern
4511     dump = true
4512 }
4513 }
4514 }
4515 if cmd == DIC_COM_DUMP || dump {
4516     fprintf(stderr,"--Id-- %v dump... %v entries:\\n",dicName,len(Romkan));
4517     var match int;
4518     for i := 0; i < len(Romkan); i++ {
4519         dic := Romkan[i].dic
4520         pat := Romkan[i].pat
4521         out := Romkan[i].out
4522         if arg == "" || 0 <= strings.Index(pat,arg) || 0 <= strings.Index(out,arg) {
4523             fmt.Printf("\\\\(%v\\t\\v {32v}{8v} {32v}{8v}\\n",
4524               i,dic,len(pat),pat,len(out),out)
4525             match += 1
4526         }
4527     }
4528     fprintf(stderr,"--Id-- %v matched %v / %v entries:\\n",arg,match,len(Romkan));
4529 }
4530 }
4531 func loadDefaultDic(dic int){
4532     if 0 < len(Romkan) ){
4533         return
4534     }
4535     //fprintf(stderr,"%r\\n")
4536     xDic([]string{"dic",DIC_COM_LOAD});
4537     var info = false
4538     if info {
4539         fprintf(stderr,"--Id-- Conguratuations!! WorldDic is now activated.\\r\\n")
4540         fprintf(stderr,"--Id-- enter `\"dic\" command for help.\\r\\n")
4541     }
4542 }
4543 func readDic()(int){
4544     var rk *os.File;
4545     var dic = "MyIME-dic.txt";
4546     //rk = fopen("romkana.txt","r");
4547     //rk = fopen("JK-JA-morse-dic.txt","r");
4548     rk = fopen(dic,"r");
4549     if rk == nil {
4550         if true {
4551             if true {
4552                 fprintf(stderr,"--%s-- Could not load %s\\n",MyIMEVER,dic);
4553             }
4554         }
4555         return -1;
4556     }
4557     if( true ){
4558         var di int;
4559         var line = make(StrBuff,1024);
4560         var pat string
4561         var out string
4562         for di = 0; di < 1024; di++ {
4563             if fgets(line,sizeof(line),rk) == NULLSP ){
4564                 break;
4565             }
4566             fmt.Sscanf(string(line[0:strlen(line)]),"`$` $",&pat,&out);
4567             //sscanf(line,"`$` $`\\r\\n`",&pat,&out);
4568             romkanadic[di].pat = pat;
4569             romkanadic[di].out = out;
4570             //fprintf(stderr,"--%d- %10s %s\\n",pat,out)
4571         }
4572         dicents += di
4573         if false {
4574             fprintf(stderr,"--%s-- loaded romkana.txt [%d]\\n",MyIMEVER,di);
4575             for di = 0; di < dicents; di++ {
4576                 fprintf(stderr,
4577                   "`$` $`\\n`,romkana[di].pat,romkana[di].out);
4578             }
4579         }
4580     }
4581     fclose(rk);
4582     //romkana[dicents].pat = "//ddump"
4583     //romkana[dicents].out = "//ddump" // dump the dic. and clean the command input
4584     /*
4585     return 0;
4586     */
4587 }
4588 func matchlen(str1 string, pat1 string)(int){
4589     if strBegins(str1,pat1) {
4590         return len(pat1)
4591     }else{
4592         return 0
4593     }
4594 }
4595 func convs(src string)(string){
4596     var si int;
4597     var ax = len(src);
4598     var di int;
4599     var mi int;
4600     var dstb []byte
4601 }
```

```

4602     for si = 0; si < sx; { // search max. match from the position
4603         if strBegins(src[si:], "%x") {
4604             var integer // s/a/b
4605             ix := strings.Index(src[si+3:], "/")
4606             if 0 < ix {
4607                 var iv int = 0
4608                 fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4609                 if Sscanf(src[si+3:si+3+ix], "%v", &iv) {
4610                     eval := fmt.Sprintf("%x", iv)
4611                     bval := []byte(eval)
4612                     dstb = append(dstb, bval...)
4613                     si = si+3+ix+1
4614                     continue
4615                 }
4616             } if strBegins(src[si:], "%d") {
4617                 // %d/integer // s/a/b/
4618                 ix := strings.Index(src[si+3:], "/")
4619                 if 0 < ix {
4620                     var int int = 0
4621                     fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4622                     eval := fmt.Sprintf("%d", iv)
4623                     bval := []byte(eval)
4624                     dstb = append(dstb, bval...)
4625                     si = si+3+ix+1
4626                     continue
4627                 }
4628             } if strBegins(src[si:], "%t") {
4629                 now := time.Now()
4630                 if now != nil {
4631                     date := now.Format(time.Stamp)
4632                     dstb = append(dstb, []byte(date)...)
4633                     si = si+3
4634                 }
4635             } continue
4636         }
4637         var maxlen int = 0;
4638         var len int;
4639         mi = -1;
4640         for di = 0; di < dicents; di++ {
4641             len = romkana[src[si:]:romkana[di].pat];
4642             if maxlen < len {
4643                 maxlen = len;
4644             }
4645             mi = di;
4646         }
4647     }
4648     if 0 < maxlen {
4649         out := romkana{mi}.out;
4650         dstb = append(dstb, []byte(out)...);
4651         si += maxlen;
4652     } else{
4653         dstb = append(dstb, src[si])
4654         si += 1;
4655     }
4656 }
4657 }
4658 return string(dstb)
4659 }
4660 func trans(src string)(int){
4661     dst := convs(src);
4662     xfpusss(dst, stderr);
4663     return 0;
4664 }
4665 // ----- LINEEDIT
4666 // "?" at the top of the line means searching history
4667 // should be compatible with Telnet
4668 const (
4669     EV_MODE      = 255
4670     EV_IDLE     = 254
4671     EV_TIMEOUT   = 253
4672
4673     GO_UP       = 252 // k
4674     GO_DOWN     = 251 // j
4675     GO_RIGHT    = 250 // l
4676     GO_LEFT     = 249 // h
4677     DEL_RIGHT   = 248 // x
4678     GO_TOPL    = 'A'-0x40 // 0
4679     GO_ENDL    = 'E'-0x40 // $
4680
4681     GO_TOPW    = 239 // b
4682     GO_ENDW    = 238 // e
4683     GO_NEXTW   = 237 // w
4684
4685     GO_FORWCH  = 229 // f
4686     GO_PAIRCH  = 228 // s
4687
4688     GO_DEL     = 219 // d
4689
4690     HI_SRCH_FW = 209 // /
4691     HI_SRCH_BK = 208 // ?
4692     HI_SRCH_RFW = 207 // n
4693     HI_SRCH_RBK = 206 // N
4694 )
4695
4696 // should return number of octets ready to be read immediately
4697 //&printf(stderr,"n-Select(%v %v)n",err,r.Bits[0])
4698
4699
4700 var EventRecvFd = -1 // file descriptor
4701 var EventSendFd = -1
4702 const EventFdOffset = 1000000
4703 const NormalFdOffset = 100
4704
4705 /* 2020-10-21 replaced poll() with channel/select
4706 func putKeyInEvent(event int, evarg int){
4707     if true {
4708         if EventRecvFd <= 0 {
4709             var pvc [1]int=[-1,-1]
4710             var pdv [1]uint=[-1,-1]
4711             syscall.Pipe(pv)
4712             EventRecvFd = pv[0]
4713             EventSendFd = pv[1]
4714             //fmt.Printf("--De-- EventPipe created[%v,%v]\n",EventRecvFd,EventSendFd)
4715         }
4716     } else{
4717         if EventRecvFd < 0 {
4718             // the document differs from this spec
4719             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
4720             sv,err := syscall.Socketpair(syscall.AF_UNIX,syscall.SOCK_STREAM,0)
4721             EventRecvFd = sv[0]
4722             EventSendFd = sv[1]
4723             if err != nil {
4724                 if err != nil {
4725                     fmt.Printf("--De-- EventSock created[%v,%v]\n",
4726                         EventRecvFd,EventSendFd,err)
4727                 }
4728             }
4729             var buf = []byte{ byte(event)}
4730             n,err := syscall.Write(EventSendFd,buf)
4731             if err != nil {
4732                 fmt.Printf("--De-- putEvent[%v]($3v)($v % v)\n",
4733                     EventSendFd,event,n,err)
4734             }
4735         }
4736     }
4737     func ungets(str string){
4738         for _,ch := range str {
4739             putkeyInEvent(int(ch),0)
4740         }
4741     }
4742     func (gsh*GshContext)xReplay(argv[]string){
4743         hix := 0
4744         tempo := 1.0
4745         xtempo := 1.0
4746         repeat := 1
4747
4748         for _,a := range argv { // tempo
4749             if strBegins(a,"x") { // tempo
4750                 fmt.Sscanf(a[1],"%f",&xtempo)
4751                 tempo = 1 / xtempo
4752                 //fmt.Printf(stderr,"--Dr-- tempo=%v\n",a[2:],tempo);
4753             } else if strBegins(a,"r") { // repeat
4754                 fmt.Sscanf(a[1],"%v",&repeat)
4755             } else if strBegins(a,"l") { // loop
4756                 fmt.Sscanf(a[1],"%d",&hix)
4757             } else{
4758                 fmt.Sscanf(a,"$d",&hix)
4759             }
4760         }
4761         if hix == 0 || len(argv) <= 1 {
4762             hix = len(gsh.CommandHistory)-1
4763         }
4764         fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n",hix,xtempo,repeat)
4765         //&printf(stderr,"/gsh.xScanReplay(hix,fals,repeat,tempo,argv)
4766         go gsh.xScanReplay(hix,true,repeat,tempo,argv)
4767
4768         runtime.Gosched(); // wait xScanReplay is launched
4769         //fmt.Printf("--Ir-- Replay setin");
4770     }
4771
4772
4773 // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4774 // 2020-0827 GShell-0.2.3
4775 */
4776 func Fpoinln(fp *os.File,usec int)(uintptr){
4777 }
```

```

4779  nfd := 1
4780
4781  rdv := syscall.FdSet {}
4782  fdl := fp.Fd()
4783  bank1 := fdl/32
4784  mask1 := int32(1 << fdl)
4785  rdv.Bits[bank1] = mask1
4786
4787  fd2 := -1
4788  bank2 := -1
4789  var mask2 int32 = 0
4790
4791  if 0 <= EventRecvFd {
4792    fd2 = EventRecvFd
4793    nfd = fd2 + 1
4794    bank2 = fd2/32
4795    mask2 = int32(1 << fd2)
4796    rdv.Bits[bank2] |= mask2
4797    //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
4798  }
4799
4800  tout := syscall.NsecToTimeval(int64(usec*1000))
4801  //n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
4802  err := syscall.Select(nfd,&rdv,nil,nil,&tout)
4803  if err != nil {
4804    //fmt.Printf("--De-- select() err(%v)\n",err)
4805  }
4806  if err == nil {
4807    if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4808      if false {
4809        fmt.Printf("--De-- got Event\n")
4810      }
4811      return uintptr(EventFdOffset + fd2)
4812    }else{
4813      if (rdv.Bits[bank1] & mask1) != 0 {
4814        return uintptr(NormalFdOffset + fdl)
4815      }else{
4816        return 1
4817      }
4818    }else{
4819      return 0
4820    }
4821  }
4822  /*
4823  */
4824  func fgetcTimeout(fp *os.File,usec int)(int){
4825  READD:
4826  //readyFd := FpollInl(fp,usec)
4827  readyFd := CPollInl(fp,usec)
4828  if readyFd < 100 {
4829    return EV_TIMEOUT
4830  }
4831
4832  var buf [1]byte
4833
4834  if EventFdOffset <= readyFd {
4835    fd := int(readyFd-EventFdOffset)
4836    ,err := syscall.Read(fd,buf[0:1])
4837    If err != nil {
4838      return EOF,
4839    }else{
4840      if buf[0] == EV_MODE {
4841        recvKeyEvent(fd)
4842        goto READ1
4843      }
4844      return int(buf[0])
4845    }
4846  }
4847
4848  ,err = fp.Read(buf[0:1])
4849  if err != nil {
4850    return EOF;
4851  }else{
4852    return int(buf[0])
4853  }
4854  /*
4855  */
4856  func visibleChar(ch int)(string){
4857  switch {
4858    case '!': return string(ch)
4859  }
4860  switch ch {
4861    case '\s': return "\\s"
4862    case '\n': return "\\n"
4863    case '\r': return "\\r"
4864    case '\t': return "\\t"
4865  }
4866  switch ch {
4867    case 0x00: return "NULL"
4868    case 0x07: return "BEL"
4869    case 0x08: return "BS"
4870    case 0x0B: return "SO"
4871    case 0x0F: return "SI"
4872    case 0x1B: return "ESC"
4873    case 0x7F: return "DEL"
4874  }
4875  switch ch {
4876    case EV_IDLE: return fmt.Sprintf("IDLE")
4877    case EV_MODE: return fmt.Sprintf("MODE")
4878  }
4879  return fmt.Sprintf("%X",ch)
4880  /*
4881  */
4882  func recvKeyEvent(fd int){
4883  var buf = make([]byte,1)
4884  _,err = syscall.Read(fd,buf[0:1])
4885  if buf[0] != 0 {
4886    romkanmode = true
4887  }else{
4888    romkanmode = false
4889  }
4890  }
4891  /*
4892  */
4893  func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv []string){
4894  var Start time.Time
4895  var events = []Event{}
4896  for ,e := range Events {
4897    if hix == 0 || e.CmdIndex == hix {
4898      events = append(events,e)
4899    }
4900  }
4901  elen := len(events)
4902  if 0 < elen {
4903    if events[elen-1].event == EV_IDLE {
4904      events = events[0:elen-1]
4905    }
4906  }
4907  for r := 0; r < repeat; r++ {
4908    for ,e := range events {
4909      name := e.when.Nanosecond()
4910      micro := name / 1000
4911      if Start.Second() == 0 {
4912        Start = time.Now()
4913      }
4914      diff := time.Now().Sub(Start)
4915      if replay {
4916        if e.event != EV_IDLE {
4917          //fmt.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
4918          putKeyEvent(e.event,0)
4919          if e.event == EV_MODE { // event with arg
4920            putKeyEvent(int(e.evarg),0)
4921          }
4922        }else{
4923          //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
4924        }
4925      }else{
4926        fmt.Printf("$7.3fms #%-3v !%-3v [%v.%06d] %3v %02X %-4v $10.3fms\n",
4927          float64(diff)/1000000.0,
4928          i,
4929          e.CmdIndex,
4930          e.when.Format(time.Timestamp),
4931          e.event,e.event.visibleChar(e.event),
4932          float64(e.evarg)/1000000.0)
4933      }
4934      if e.event == EV_IDLE {
4935        //fmt.Printf("--replay %v / %v delay\n",i,len(events));
4936        d := time.Duration(float64(time.Duration(e.evarg)) * tempo
4937        //nsleep(time.Duration(e.evarg))
4938        nsleep(d)
4939      }
4940    }
4941  }
4942  }
4943  /*
4944  */
4945  func dumpEvents(argv []string){
4946  hix := 0
4947  if 1 < len(argv) {
4948    fmt.Sscanf(argv[1],"%d",&hix)
4949  }
4950  for ,e := range Events {
4951    name := e.when.Nanosecond()
4952    micro := name / 1000
4953    //if e.event != EV_TIMEOUT {
4954    if hix == 0 || e.CmdIndex == hix {
4955      fmt.Printf("#%-3v !%-3v [%v.%06d] %3v %02X %-4v $10.3fms\n",i,
4956          e.CmdIndex,

```

```

4956         e.when.Format(time.Stamp),micro,
4957     }
4958   //}
4959 }
4960 }
4961 */
4962 func fgetcTimeout(fp *os.File,usec int)(int{
4963   ch := fgetcTimeout(fp,usec)
4964   if ch != EV_TIMEOUT {
4965     now := time.Now()
4966     if 0 < len(Events) {
4967       last := Events[len(Events)-1]
4968       dura := int64(now.Sub(last.when))
4969       Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
4970     }
4971     Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4972   }
4973   Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4974 }
4975 }
4976 // 2020-10-21 replaced poll() with channel/select
4977 var Kbd = make(chan int);
4978 var Kbinit = false;
4979 var evQ = make(chan int);
4980 /*
4981 func keyInput(kbd chan int, fp *os.File){
4982   for {
4983     ch := C.getc(C.stdin);
4984     if(ch == C.EOF){break;}
4985     kbd <- int(ch);
4986   }
4987 }
4988 // https://godoc.org/golang.org/x/crypto/ssh/terminal
4989 // https://stackoverflow.com/questions/14094190/function-similar-to-getchar
4990 func keyInput(kbd chan int, tty *os.File){
4991   tmode := C.setTermRaw();
4992   defer func(){C.setTermMode(tmode); }()
4993   if(!OnWindows()){
4994     system("bin/stty -echo -icanon");
4995     defer func(){ system("bin/stty sane"); }()
4996   }
4997   for {
4998     var rbuf []byte = make([]byte,1);
4999     if( OnWindows ){
5000       C.setTermRaw();
5001     }
5002     _,rerr := tty.Read(rbuf);
5003     if(rerr != nil){
5004       break;
5005     }
5006     //fmt.Printf("%+v\n",rbuf[0]);
5007     kbd <- int(rbuf[0]);
5008   }
5009   if( !OnWindows ) { system("bin/stty echo sane"); }
5010 }
5011 func fgetcTimeout(fp *os.File,usec int)(int{
5012   if( !Kbinit ){
5013     Kbinit = true;
5014     go keyInput(Kbd,fp);
5015   }
5016   for {
5017     select {
5018       case ch := <- time.After(time.Duration(usec*1000)):
5019         //fmt.Println("-Timeout "+ usc"\n",usec);
5020         return EV_TIMEOUT;
5021       case ch := <- Kbd:
5022         // record a Keyin(ch) Event
5023         now := time.Now()
5024         if 0 < len(Events) {
5025           last := Events[len(Events)-1]
5026           dura := int64(now.Sub(last.when))
5027           Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5028         }
5029         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5030       }
5031     }
5032     return ch;
5033   case ch := <- evQ:
5034     if(ch == EV_MODE ){
5035       recvKeyEvent();
5036     }else{
5037       return ch;
5038     }
5039   }
5040 }
5041 func putKeyEvent(event int, evarg int){
5042   evQ <- event;
5043 }
5044 func recvKeyEvent(){
5045   ch := <- evQ;
5046   if( ch != 0 ){
5047     romkanemode = true
5048   }else{
5049     romkanemode = false
5050   }
5051 }
5052 var AtConsoleLineTop = true
5053 var TtyMaxCol = 72 // to be obtained by ioctl?
5054 var EscTimeout = (100*1000)
5055 var {
5056   MODE_ViMode bool // vi compatible command mode
5057   MODE_ViMode2 bool // sh compatible command mode
5058   romkanemode bool // shown translation mode, the mode to be retained
5059   MODE_Recursive bool // recursive translation
5060   MODE_CapsLock bool // software CapsLock
5061   MODE_LowerLock bool // force lower-case character lock
5062   MODE_ViInsert int // visible insert mode, should be like "I" icon in X Window
5063   MODE_ViTrace bool // output newline before translation
5064 }
5065 type IInput struct {
5066   lno int
5067   lastlno int
5068   pch []byte // input queue
5069   prompt string
5070   line string
5071   right string
5072   inIMode bool
5073   pinJMode bool
5074   waitingMeta string // waiting meta character
5075   Lastcmd string
5076 }
5077 func (iin*IInput)Getc(timeoutUs int)(int{
5078   ch1 := EOF
5079   ch2 := EOF
5080   ch3 := EOF
5081   if( 0 < len(iin.pch) ){ // deQ
5082     ch1 = iin.pch[0]
5083     iin.pch = iin.pch[1];
5084   }else{
5085     ch1 = fgetcTimeout(stdin,timeoutUs);
5086   }
5087   if( ch1 == 033 ){ // escape sequence
5088     ch2 = fgetcTimeout(stdin,EscTimeout);
5089     if( ch2 == EV_TIMEOUT ){
5090       ch3 = fgetcTimeout(stdin,EscTimeout);
5091       if( ch3 == EV_TIMEOUT ){
5092         iin.pch = append(iin.pch,ch2) // enQ
5093       }else{
5094         switch( ch2 ){
5095           default:
5096             iin.pch = append(iin.pch,ch2) // enQ
5097             iin.pch = append(iin.pch,ch3) // enQ
5098           case '[':
5099             switch( ch3 ){
5100               case 'A': ch1 = GO_UP; // ^
5101               case 'B': ch1 = GO_DOWN; // v
5102               case 'C': ch1 = GO_RIGHT; // >
5103               case 'D': ch1 = GO_LEFT; // <
5104               case '3':
5105                 ch4 := fgetcTimeout(stdin,EscTimeout);
5106                 if( ch4 == '-' ){
5107                   //fprintf(stderr,"%02X %02X %02X\n",ch1,ch2,ch3,ch4);
5108                   ch1 = DEL_RIGHT
5109                 }
5110             }
5111           case '\\':
5112             ch4 := fgetcTimeout(stdin,EscTimeout);
5113             //fprintf(stderr,"%02X %02X %02X\n",ch1,ch2,ch3,ch4);
5114             switch( ch3 ){
5115               case '-': ch1 = DEL_RIGHT
5116             }
5117           }
5118         }
5119       }
5120     }
5121   }
5122   //ch4 := fgetcTimeout(stdin,EscTimeout);
5123   //fprintf(stderr,"%02X %02X %02X\n",ch1,ch2,ch3,ch4);
5124   switch( ch3 ){
5125     case '-': ch1 = DEL_RIGHT
5126   }
5127 }
5128 }
5129 }
5130 return ch1
5131 }
5132 func (inn*IInput)clearline(){

```

```

5133 var i int
5134 fprintf(stderr,"%c");
5135 // show be ANSI ESC sequence
5136 for i = 0,1,< TtyMaxCol; i++ { // to the max. position in this input action
5137     fputc(i,os.Stderr);
5138 }
5139 fprintf(stderr,"%r");
5140
5141 func (iin*IInput)Redraw(){
5142     redraw(iin,iin.lno,int,line string,right string){
5143         iinMeta := false
5144         showMeta := "" // visible Meta mode on the cursor position
5145         showLino := fmt.Sprintf("%d",iin.lno)
5146         InsertMark := "" // in visible insert mode
5147
5148     if MODE_VicMode {
5149         if 0 < len(iin.right) {
5150             InsertMark = ""
5151         }
5152
5153         if len(iin.waitingMeta) {
5154             inMeta = true
5155             if iin.waitingMeta[0] != 033 {
5156                 showMeta = iin.waitingMeta
5157             }
5158         }
5159     } else if romkanmode {
5160         romkan := ""
5161         inMeta := "-"
5162         inveri := ""
5163         if MODE_CapsLock {
5164             inmeta = "A"
5165         } else if MODE_LowerLock {
5166             inmeta = "a"
5167         } else if MODE_Vitrace {
5168             inveri = "v"
5169         } else if MODE_VicMode {
5170             inveri = ";"
5171         }
5172     } else if romkannode {
5173         romkan = "\u0343\u2011\u202d"
5174         if MODE_Capslock {
5175             inmeta = "R"
5176         } else {
5177             inmeta = "r"
5178         }
5179     }
5180     if inMeta {
5181         inmeta = "\\"
5182     }
5183     showMode = "["+romkan+inmeta+inveri+"]";
5184
5185 Pre := "%r" + showMode + showLino
5186 Output := ""
5187 Left := ""
5188 Right := ""
5189 if romkanmode {
5190     Left = convs(line)
5191     Right = InsertMark+convs(right)
5192 } else {
5193     Left = line
5194     Right = InsertMark+right
5195 }
5196 Output = Pre+Left
5197 if MODE_Vitrace {
5198     Output += iin.LastCmd
5199 }
5200 Output += showMeta+Right
5201 for iin(Output) < TtyMaxCol { // to the max. position that may be dirty
5202     Output += "\n"
5203     // should be ANSI ESC sequence
5204     // not necessary just after newline
5205 }
5206 Output += Pre+Left+showMeta // to set the cursor to the current input position
5207 fprintf(stderr,"%s",Output)
5208
5209 if MODE_Vittrace {
5210     if 0 < len(iin.LastCmd) {
5211         iin.LastCmd = ""
5212         fprintf(stderr,"%r\n")
5213     }
5214 }
5215 AtConsoleLineTop = false
5216 //fmt.Println("Redraw(%v)(%v)\n",len(line),len(right));
5217 //<a href="https://colan.org/pka/unicode/utf8">utf8</a>
5218 func DelHeadChar(str string)(rline string,head string){
5219     clen := utf8.DecodeRune([]byte(str))
5220     head = string(0:clen)
5221     return str[clen],head
5222 }
5223 func delTailChar(str string)(rline string,last string){
5224     var i = 0
5225     var clen = 0
5226     for {
5227         i,siz := utf8.DecodeRune([]byte(str)[i:])
5228         if i <= 0 { break }
5229         clen = siz
5230         i += siz
5231     }
5232     last = str[:len(str)-clen:]
5233     return str[0:len(str)-clen],last
5234 }
5235
5236 // 3 for output and history
5237 // 4 for keylog?
5238 // <a name="getline">Command Line Editor</a>
5239 func xgetline(cmdch int,prevline string,gsh*GshContext)(string){
5240     var iin Input
5241     iin.lastlno = lno
5242     iin.lno = lno
5243
5244     CmdIndex = len(gsh.CommandHistory)
5245     if(isatty(0) == 0) {
5246         if( sgetchar(&iin.line,LINESIZE,stdin) == NULL ) {
5247             iin.line = "exit\n";
5248         } else{
5249             return iin.line
5250         }
5251     }
5252     if( true ) {
5253         //var pts string;
5254         //pts = ptsname(0);
5255         //pts = ttypname(0);
5256         //fprint(stderr,"-pts[0] = %s\n",pts?pts:"");
5257     }
5258     if( false ) {
5259         fprintf(stderr,"! ");
5260         fflush(stderr);
5261         sgetchar(&iin.line,LINESIZE,stdin);
5262         return iin.line
5263     }
5264     if( !onWindows() || system("/bin/stty -echo -icanon"); )
5265     xline := iin.xgetline(prevline,gsh)
5266     if( !onWindows() )(system("/bin/stty echo sane"); )
5267     return xline
5268 }
5269
5270 func (iin*IInput)Translate(cmdch int){
5271     romkanmode = !romkanmode;
5272     if MODE_Vitrace {
5273         fprintf(stderr,"%s\r\n",string(cmdch));
5274     } else{
5275         if( cmdch == 'J' ){
5276             fprint(stderr,J\r\n);
5277             iin.inJMode = true
5278         }
5279         iin.Redraw();
5280         loadDefaultBdc(cmdch);
5281         iin.Redraw();
5282     }
5283 }
5284 func (iin*IInput)Replace(cmdch int){
5285     iin.Redraw();
5286     dst := fmt.Sprintf("\\\\v",string(cmdch))
5287     iin.Redraw();
5288     loadDefaultBdc(cmdch);
5289     dst := convs(iin.line[iin.right]);
5290     iin.line = dst
5291     iin.right = ""
5292     if( iin.inJMode ) {
5293         fprintf(stderr,I\r\n);
5294         iin.inJMode = true
5295     }
5296     iin.Redraw();
5297 }
5298 //<a href="https://colan.org/pka/unicode/utf8">utf8</a>
5299 func isAlpha(ch rune)(bool){
5300     return ch >= 'A' && ch <= 'Z' ||
5301            ch >= 'a' && ch <= 'z';
5302 }
5303
5304 //<a href="https://colan.org/pka/unicode/utf8">utf8</a>
5305 func isAlphaCh(rune)(bool){
5306     return r >= 'A' && r <= 'Z' ||
5307            r >= 'a' && r <= 'z';
5308 }

```

```

5310     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5311         return true
5312     }
5313     return false
5314 }
5315 func isAlnum(ch rune)(bool){
5316     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5317         return true
5318     }
5319     if '0' <= ch && ch <= '9' {
5320         return true
5321     }
5322     return false
5323 }
5324 // 0.2.8 2020-0901 created
5325 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5326 func (iin*Input)GotoTOPW(){
5327     str := iin.line
5328     i := len(str)
5329     if i <= 0 {
5330         return
5331     }
5332     //10 := i
5333     i = 10
5334     lastSize := 0
5335     var lastRune rune
5336     var found = -1
5337     for 0 < i { // skip preamble spaces
5338         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5339         if !isAlnum(lastRune) { // character, type, or string to be searched
5340             i = lastSize
5341             continue
5342         }
5343         break
5344     }
5345     for 0 < i {
5346         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5347         if lastSize <= 0 { continue } // not the character top
5348         if !isAlnum(lastRune) { // character, type, or string to be searched
5349             found = i
5350             break
5351         }
5352         i -= lastSize
5353     }
5354     if found < 0 && i == 0 {
5355         found = 0
5356     }
5357     if 0 <= found {
5358         if isAlnum(lastRune) { // or non-kana character
5359             else{ // when positioning to the top o the word
5360                 i += lastSize
5361             iin.right = str[i:] + iin.right
5362             if 0 < i {
5363                 iin.line = str[0:i]
5364             }else{
5365                 iin.line = ""
5366             }
5367         }
5368     }
5369 //fmt.Println("(n%d){$d}{$s}{$s}\n",io,i,found,iin.line,iin.right)
5370 //fmt.Println("")) // set debug messae at the end of line
5371 }
5372 // 0.2.8 2020-0901 created
5373 func (iin*Input)GotoENDW(){
5374     str := iin.right
5375     if len(str) <= 0 {
5376         return
5377     }
5378     lastSize := 0
5379     var lastRune rune
5380     i := 0
5381     inWord := false
5382
5383     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
5384     if isAlnum(lastRune) {
5385         r,z := utf8.DecodeRuneInString(str[lastSize:])
5386         if 0 < z && isAlnum(r) {
5387             inWord = true
5388         }
5389     }
5390     for i < len(str) {
5391         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5392         if lastSize <= 0 { break } // broken data?
5393         if !isAlnum(lastRune) { // character, type, or string to be searched
5394             break
5395         }
5396         lastW = i // the last alnum if in alnum word
5397         i += lastSize
5398     }
5399     if inWord {
5400         goto DISP
5401     }
5402     for i < len(str) {
5403         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5404         if lastSize <= 0 { break } // broken data?
5405         if !isAlnum(lastRune) { // character, type, or string to be searched
5406             break
5407         }
5408         i += lastSize
5409     }
5410     for i < len(str) {
5411         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5412         if lastSize <= 0 { break } // broken data?
5413         if !isAlnum(lastRune) { // character, type, or string to be searched
5414             break
5415         }
5416         lastW = i
5417         i += lastSize
5418     }
5419     lastW = i // the last alnum if in alnum word
5420     i += lastSize
5421     DISP:
5422     if 0 < lastW {
5423         iin.line = iin.line + str[0:lastW]
5424         iin.right = str[lastW:]
5425     }
5426 //fmt.Println("(n%d){$s}{$s}\n",i,iin.line,iin.right)
5427 //fmt.Println("")) // set debug messae at the end of line
5428 }
5429 // 0.2.8 2020-0901 created
5430 func (iin*Input)GotoNEXTW(){
5431     str := iin.right
5432     if len(str) <= 0 {
5433         return
5434     }
5435     lastSize := 0
5436     var lastRune rune
5437     var found = -1
5438     i := 1
5439     for i < len(str) {
5440         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5441         if lastSize <= 0 { break } // broken data?
5442         if !isAlnum(lastRune) { // character, type, or string to be searched
5443             found = i
5444             found += lastSize
5445         }
5446         i += lastSize
5447     }
5448     if 0 < found {
5449         if isAlnum(lastRune) { // or non-kana character
5450             else{ // when positioning to the top o the word
5451                 found += lastSize
5452             }
5453             iin.line = iin.line + str[0:found]
5454             if 0 < found {
5455                 iin.right = str[found:]
5456             }else{
5457                 iin.right = ""
5458             }
5459         }
5460     }
5461 //fmt.Println("\n($d){$s}{$s}\n",i,iin.line,iin.right)
5462 //fmt.Println("")) // set debug messae at the end of line
5463 }
5464 // 0.2.8 2020-0902 created
5465 func (iin*Input)GotoPAIRCH(){
5466     str := iin.right
5467     if len(str) <= 0 {
5468         return
5469     }
5470     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5471     if lastSize <= 0 {
5472         return
5473     }
5474     forw := false
5475     back := false
5476     pair := ''
5477     switch string(lastRune){
5478     case ")": pair = "("; forw = true
5479     case ")": pair = "("; back = true
5480     case ")": pair = "("; forw = true
5481     case ")": pair = "("; back = true
5482     case ")": pair = "["; forw = true
5483     case "<": pair = ">"; forw = true
5484     case ">": pair = "<"; back = true
5485     case "<": pair = ">"; back = true
5486     case ">": pair = "<"; context depednet, can be f" or back-double quote
5487     case "<": pair = ">"; // context depednet, can be f' or back-quote
5488     }

```

```

5487     // case Japanese Kakkos
5488     if forw {
5489         iin.SearchForward(pair)
5490     }
5491     if back {
5492         iin.SearchBackward(pair)
5493     }
5494 }
5495 // 0.2.8 2020-0902 created
5496 func (iin*IInput)SearchForward(pat string)(bool){
5497     right := iin.right
5498     found := -1
5499     i := 0
5500     if strbegins(right,pat) {
5501         z := utf8.DecodeRuneInString(right[i:])
5502         if 0 <= z {
5503             i += z
5504         }
5505     }
5506     for i < len(right) {
5507         if strbegins(right[i:],pat) {
5508             found = i
5509             break
5510         }
5511         z := utf8.DecodeRuneInString(right[i:])
5512         if z <= 0 { break }
5513         i += z
5514     }
5515     if 0 <= found {
5516         iin.line = iin.line + right[0:found]
5517         iin.right = iin.right[found:]
5518         return true
5519     }else{
5520         return false
5521     }
5522 }
5523 // 0.2.8 2020-0902 created
5524 func (iin*IInput)SearchBackward(pat string)(bool){
5525     limit := iin.line
5526     found := -1
5527     i := len(line)-1
5528     for i >= 0 < i; i-- {
5529         z := utf8.DecodeRuneInString(line[i:])
5530         if z <= 0 {
5531             continue
5532         }
5533         //fprintf(stderr,"-- %v\n",pat,line[i:])
5534         if strbegins(line[i:],pat) {
5535             found = i
5536             break
5537         }
5538     }
5539     //fprintf(stderr,"--%d\n",found)
5540     if 0 <= found {
5541         iin.right = line[found:] + iin.right
5542         iin.line = line[0:found]
5543         return true
5544     }else{
5545         return false
5546     }
5547 }
5548 // 0.2.8 2020-0902 created
5549 // search from top, end, or current position
5550 func (gsh*GshContext)SearchHistory(pat string, forw bool,string){
5551     if forw {
5552         for ,v := range gsh.CommandHistory {
5553             if 0 <= strings.Index(v.CmdLine,pat) {
5554                 //fprintf(stderr,"%v\n",v.CmdLine)
5555                 return true,v.CmdLine
5556             }
5557         }
5558     }else{
5559         hlen := len(gsh.CommandHistory)
5560         for i := hlen-1; i>=0; {
5561             v := gsh.CommandHistory[i]
5562             if 0 <= strings.Index(v.CmdLine,pat) {
5563                 //fprintf(stderr,"%v\n",v.CmdLine)
5564                 return true,v.CmdLine
5565             }
5566         }
5567     }
5568     //fprintf(stderr,"n-- not-found(%v)\n",pat)
5569     return false,"(Not Found in History)"
5570 }
5571 // 0.2.8 2020-0902 created
5572 func (iin*IInput)GotoFORSTR(pat string,gsh*GshContext){
5573     if !forw {
5574         if 0 < len(iin.right) {
5575             if 0 < len(iin.right) {
5576                 found := iin.SearchForward(pat)
5577             }
5578             if !found {
5579                 found, line := gsh.SearchHistory(pat,true)
5580                 if found {
5581                     iin.line = line
5582                     iin.right = ""
5583                 }
5584             }
5585         }
5586         func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5587             found := false
5588             if 0 < len(iin.line) {
5589                 found = iin.SearchBackward(pat)
5590             }
5591             if !found {
5592                 found, line := gsh.SearchHistory(pat,false)
5593                 if found {
5594                     iin.line = line
5595                     iin.right = ""
5596                 }
5597             }
5598         }
5599         func (iin*IInput)getstringl(prompt string)(string){ // should be editable
5600             iin.clearline();
5601             fprintf(stderr,"%r\n",prompt)
5602             str := ""
5603             for {
5604                 ch := iin.Getc(10*1000*1000)
5605                 if ch == '\n' || ch == '\r' {
5606                     break
5607                 }
5608                 sch := string(ch)
5609                 str += sch
5610                 fprintf(stderr,"%s",sch)
5611             }
5612             return str
5613         }
5614     }
5615 // search pattern must be an array and selectable with ^N/P
5616 var SearchPat = ""
5617 var SearchForw = true
5618
5619 func (iin*IInput)xgetlinel(prevline string, gsh*GshContext)(string){
5620     var ch int;
5621     MODE_ShowMode = false
5622     MODE_VicMode = false
5623     iin.Redraw();
5624     first := true
5625
5626     for cix := 0; cix++ {
5627         iin.pinJmode = iin.inJmode
5628         iin.inJmode = false
5629         ch = iin.Getc(1000*1000)
5630
5631         if ch != EV_TIMEOUT && first {
5632             first = false
5633             mode := 0
5634             if romkanmode {
5635                 mode = 1
5636             }
5637             now := time.Now()
5638             Events = append(Events,Event{now,EV_MODE,int64(mode),CmdIndex})
5639         }
5640         if ch == 033 {
5641             MODE_ShowMode = true
5642             MODE_VicMode = IMODE_VicMode
5643             iin.Redraw();
5644             continue
5645         }
5646         if MODE_VicMode {
5647             switch ch {
5648                 case '0': ch = GO_TOPL
5649                 case '$': ch = GO_ENDL
5650                 case 'b': ch = GO_TOPW
5651                 case 'd': ch = GO_DOWNW
5652                 case 'w': ch = GO_NEXTW
5653                 case '%': ch = GO_PAIRCH
5654
5655                 case 'i': ch = GO_DOWN
5656                 case 'k': ch = GO_UP
5657                 case 'h': ch = GO_LEFT
5658                 case 'l': ch = GO_RIGHT
5659                 case 'x': ch = DEL_RIGHT
5660                 case 'a': MODE_VicMode = !MODE_VicMode
5661                 ch = GO_RIGHT
5662             }
5663         }
5664     }
5665 }
```

```

5664     case 'i': MODE_VicMode = IMODE_VicMode
5665     iin.Redraw();
5666     continue
5667   case '-':
5668     right.head := delHeadChar(iin.right)
5669     if len([!byte(head)]) == 1 {
5670       ch = int(head[0])
5671       if ('a' <= ch && ch <= 'z') {
5672         ch = ch + 'A'-'a'
5673       } else {
5674         if ('A' <= ch && ch <= 'Z') {
5675           ch = ch + 'a'-'A'
5676         }
5677       iin.right = string(ch) + right
5678     }
5679     iin.Redraw();
5680     continue
5681   case 'f': // GO_FORWCH
5682     iin.Redraw();
5683     ch = iin.GetString(3*1000*1000)
5684     if ch == EV_TIMEOUT {
5685       iin.Redraw();
5686       continue
5687     }
5688     SearchPat = string(ch)
5689     SearchForm = true
5690     iin.GotoFORWSTR(SearchPat,gsh)
5691     iin.Redraw();
5692     continue
5693   case '/':
5694     SearchPat = iin.GetString("/") // should be editable
5695     SearchForm = true
5696     iin.GotoFORWSTR(SearchPat,gsh)
5697     iin.Redraw();
5698     continue
5699   case '?':
5700     SearchPat = iin.GetString("?") // should be editable
5701     SearchForm = false
5702     iin.GotoBACKSTR(SearchPat,gsh)
5703     iin.Redraw();
5704     continue
5705   case 'n':
5706     if !SearchForm {
5707       iin.GotoFORWSTR(SearchPat,gsh)
5708     } else {
5709       iin.GotoBACKSTR(SearchPat,gsh)
5710     }
5711     iin.Redraw();
5712     continue
5713   case 'N':
5714     if !SearchForm {
5715       iin.GotoFORWSTR(SearchPat,gsh)
5716     } else {
5717       iin.GotoBACKSTR(SearchPat,gsh)
5718     }
5719     iin.Redraw();
5720     continue
5721   }
5722   switch ch {
5723     case GO_TOP:
5724       iin.GotoTOPW()
5725       iin.Redraw();
5726       continue
5727     case GO_BOT:
5728       iin.GotoBNDW()
5729       iin.Redraw();
5730       continue
5731     case GO_NEXTW:
5732       if iin.right == ' ' then
5733         iin.GotoNEXTW()
5734       iin.Redraw();
5735       continue
5736     case GO_PAIRCH:
5737       iin.GotoPAIRCH()
5738       iin.Redraw();
5739       continue
5740     case GO_PREVW:
5741       if iin.left == ' ' then
5742         iin.GotoPREVW()
5743       iin.Redraw();
5744       continue
5745     case '0': if ch == 033 ){
5746       MODE_ShowMode = true
5747       metach := ch
5748       iin.waitingMeta = string(ch)
5749       iin.Redraw();
5750       /* set cursor //fprintf(stderr,"???\\b\\b\\b")
5751       ch = fgetchar(stdin,2000*1000)
5752       /* reset cursor
5753       iin.waitingMeta = ""
5754
5755       cmdch := ch
5756       if( ch == EV_TIMEOUT ){
5757         if metach == 033 {
5758           continue
5759         }
5760         ch = metach
5761       } else
5762       if( ch == 'm' || ch == 'M'){
5763         mch := fgetcharTimeout(stdin,1000*1000)
5764         if mch == 'r' {
5765           romkanmode = true
5766         } else{
5767           romkanmode = false
5768         }
5769         continue
5770       } else
5771       if( ch == 'k' || ch == 'K'){
5772         MODE_Recursive = IMODE_Recursive
5773         iin.Translate(cmdch);
5774         continue
5775       } else
5776       if( ch == 'j' || ch == 'J'){
5777         iin.Translate(cmdch);
5778         continue
5779       } else
5780       if( ch == 'i' || ch == 'I'){
5781         iin.Replace(cmdch);
5782         continue
5783       } else
5784       if( ch == 'l' || ch == 'L'){
5785         MODE_LowerLock = IMODE_LowerLock
5786         MODE_CapsLock = false
5787         if MODE_ViTrace {
5788           fprintf(stderr,"%v\r\n",string(cmdch));
5789         }
5790         iin.Redraw();
5791         continue
5792       } else
5793       if( ch == 'u' || ch == 'U'){
5794         MODE_CapLock = IMODE_CapLock
5795         MODE_LowerLock = false
5796         if MODE_ViTrace {
5797           fprintf(stderr,"%v\r\n",string(cmdch));
5798         }
5799         iin.Redraw();
5800         continue
5801       } else
5802       if( ch == 'v' || ch == 'V'){
5803         MODE_ViTrace = IMODE_ViTrace
5804         if MODE_ViTrace {
5805           fprintf(stderr,"%v\r\n",string(cmdch));
5806         }
5807         iin.Redraw();
5808         continue
5809       } else
5810       if( ch == 'c' || ch == 'C'){
5811         if 0 < len(iin.line) {
5812           xline.tail = delTailChar(iin.line)
5813           if len([!byte(tail)]) == 1 {
5814             ch = int(tail[0])
5815             if ('a' <= ch && ch <= 'z') {
5816               ch = ch + 'A'-'a'
5817             } else {
5818               if ('A' <= ch && ch <= 'Z') {
5819                 ch = ch + 'a'-'A'
5820               }
5821             iin.line = xline + string(ch)
5822           }
5823         }
5824       } else
5825       if MODE_ViTrace {
5826         fprintf(stderr,"%v\r\n",string(cmdch));
5827       }
5828       iin.Redraw();
5829       continue
5830     } else
5831       iin.pch = append(iin.pch,ch) // push
5832       ch = '\\'
5833     }
5834   switch( ch ){
5835     case 'P'-0x40: ch = GO_UP
5836     case 'N'-0x40: ch = GO_DOWN
5837     case 'B'-0x40: ch = GO_LEFT
5838     case 'F'-0x40: ch = GO_RIGHT
5839   }

```

```

5841 //fprintf(stderr,"B[%02X]\n",ch);
5842 switch(ch){
5843     case 0:
5844         continue;
5845     case 't':
5846         iin.Replace('j');
5847         continue;
5848     case 'X'-0x40:
5849         iin.Replace('j');
5850         continue;
5851     case EV_TIMEOUT:
5852         iin.Redraw();
5853         if iin.pinJmode {
5854             fprintf(stderr,"\\J\r\n");
5855             iin.inJmode = true
5856         }
5857         continue;
5858     case GO_UP:
5859         if iin.lno == 1 {
5860             continue;
5861         }
5862         cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
5863         if ok {
5864             iin.line = cmd
5865             iin.right = ""
5866             iin.lno = iin.lno - 1
5867         }
5868         iin.Redraw();
5869         continue;
5870     case GO_DOWN:
5871         cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
5872         if ok {
5873             iin.line = cmd
5874             iin.right = ""
5875             iin.lno = iin.lno + 1
5876         }else{
5877             iin.line = ""
5878             iin.right = ""
5879             if iin.lno == iin.lastlno-1 {
5880                 iin.lno = iin.lno + 1
5881             }
5882         }
5883         iin.Redraw();
5884         continue;
5885     case GO_LEFT:
5886         if 0 < len(iin.line) {
5887             xline,tail := delTailChar(iin.line)
5888             iin.line = xline
5889             iin.right = tail + iin.right
5890         }
5891         iin.Redraw();
5892         continue;
5893     case GO_RIGHT:
5894         if( 0 < len(iin.right) && iin.right[0] != 0 ){
5895             xright,head := delHeadChar(iin.right)
5896             iin.right = xright
5897             iin.line += head
5898         }
5899         iin.Redraw();
5900         continue;
5901     case EOF:
5902         goto EXIT;
5903     case 'R'-0x40: // replace
5904         dst := conv(iin.line+iin.right);
5905         iin.line = dst
5906         iin.right = "";
5907         iin.Redraw();
5908         continue;
5909     case 'T'-0x40: // just show the result
5910         readDic();
5911         romkanmode = !romkanmode;
5912         iin.Redraw();
5913         continue;
5914     case 'U'-0x40:
5915         iin.Redraw();
5916         continue;
5917     case 'K'-0x40:
5918         iin.right = "";
5919         iin.Redraw();
5920         continue;
5921     case 'C'-0x40:
5922         iin.Redraw();
5923         continue;
5924     case 'E'-0x40:
5925         iin.line += iin.right
5926         iin.right = "";
5927         iin.Redraw();
5928         continue;
5929     case 'A'-0x40:
5930         iin.right = iin.line + iin.right
5931         iin.line = "";
5932         iin.Redraw();
5933         continue;
5934     case 'D'-0x40:
5935         iin.line = "";
5936         iin.right = "";
5937         iin.clearline();
5938         iin.Redraw();
5939         continue;
5940     case 'P'-0x40:
5941         if( 0 < len(iin.right) ){
5942             iin.right_ = delHeadChar(iin.right)
5943             iin.Redraw();
5944         }
5945         continue;
5946     case 0x7F: // BS? not DEL
5947         if( 0 < len(iin.line) ){
5948             iin.line_ = delTailChar(iin.line)
5949             iin.Redraw();
5950         }
5951         /*
5952         else
5953             if( 0 < len(iin.right) ){
5954                 iin.right_ = delHeadChar(iin.right)
5955                 iin.Redraw();
5956             }
5957         */
5958         continue;
5959     case 'H'-0x40:
5960         if( 0 < len(iin.line) ){
5961             iin.line_ = delTailChar(iin.line)
5962             iin.Redraw();
5963         }
5964         continue;
5965     if( OnWindows && ch == '\n' ){
5966         continue;
5967     }
5968     if( ch == '\n' || ch == '\r' ){
5969         iin.line += iin.right;
5970         iin.right = "";
5971         iin.Redraw();
5972         //fputc(ch,stderr);
5973         fprintf(stderr,"%c\n"); // NL on Unix, CR on Windows
5974         AtConsoleLineTop = true
5975         break;
5976     }
5977     if MODE_CapsLock {
5978         if 'A' <= ch && ch <= 'Z' {
5979             ch = ch+'A'-'a';
5980         }
5981     }
5982     if MODE_LowerLock {
5983         if 'A' <= ch && ch <= 'Z' {
5984             ch = ch+'a'-'A';
5985         }
5986     }
5987     iin.line += string(ch);
5988     iin.Redraw();
5989 }
5990 EXIT:
5991     return iin.line + iin.right;
5992 }
5993 func getline_main(){
5994     line := xgetline(0,"",nil);
5995     fprintf(stderr,"%s\n",line);
5996     /* dp = strpbrk(line,"`\r\n`");
5997     if( dp != NULL ){
5998         *dp = 0;
5999     }
5999     if( 0 ){
6000         fprintf(stderr,"%n(%d)\n",int(strlen(line)));
6001     }
6002     if( lseek(3,0,0) == 0 ){
6003         if( mode == modeRead ) {
6004             var buf {8*1024}byte;
6005             convs(line,buf);
6006             strcpy(line,buf);
6007             write(3,line,strlen(line));
6008             ftruncate(3,lseek(3,0,SEEK_CUR));
6009             //fprintf(stderr,"outsize=%d\n", (int)lseek(3,0,SEEK_END));
6010             lseek(3,0,SEEK_SET);
6011             close(3);
6012         }else{
6013             write(3,line,strlen(line));
6014             ftruncate(3,lseek(3,0,SEEK_CUR));
6015             //fprintf(stderr,"outsize=%d\n", (int)lseek(3,0,SEEK_END));
6016             lseek(3,0,SEEK_SET);
6017             close(3);
6018     }
6019 }
```

```

6018     fprintf(stderr, "\r\ngetline: ");
6019     trans(line);
6020     //printf("%s\n",line);
6021     printf("\n");
6022   }
6023 */
6024 }
6025 //== end ===== getline
6026
6027 //
6028 // $USERHOME/.gsh/
6029 //   gsh-rc.txt, or gsh-configure.txt
6030 //   gsh-history.txt
6031 //   gsh-aliases.txt // should be conditional?
6032 //
6033 func (gshCtx *GshContext)gshSetupHomeDir(bool) {
6034   homedir,found := userHomeDir()
6035   if !found {
6036     fmt.Printf("--E-- You have no UserHomeDir\n");
6037     return true
6038   }
6039   gshhome := homedir + "/" + GSH_HOME
6040   err2 := os.Stat(gshhome)
6041   if err2 != nil {
6042     err := os.Mkdir(gshhome,0700)
6043     if err != nil {
6044       fmt.Printf("--E-- Could not Create %s (%s)\n",
6045             gshhome,err)
6046     }
6047   }
6048   fmt.Printf("--I-- Created %s\n",gshhome)
6049 }
6050 gshCtx.GshHomeDir = gshhome
6051 return false
6052 }
6053 func setupGshContext(GshContext,bool){
6054   //gshPA := syscall.ProcAttr {
6055   gshPA := os.ProcAttr {
6056     "" // the starting directory
6057     os.Environ(), // environ[]
6058     //|uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
6059     []os.File{os.Stdin,os.Stdout,os.Stderr},
6060     nil, // OS specific
6061   }
6062   cwd, _ := os.Getwd()
6063   gshCtx := GshContext {
6064     cwd, // StartDir
6065     "", // Getline
6066     []GshDirHistory { { cwd,time.Now(),0 } }, // ChdirHistory
6067     gshPA,
6068     []GCommandHistory{}, // something for invocation?
6069     GCommandHistory{}, // CmdCurrent
6070     false,
6071     []GProcessState{}, // []int{},
6072     aUsage{},
6073     "" // GshHomeDir
6074     Ttyid(),
6075     false,
6076     false,
6077     []PluginInfo{},
6078     []string{},
6079     " ",
6080     "v",
6081     ValueStack{},
6082     GServer{}, // LastServer
6083     "" // RSERV
6084     cwd, // RWD
6085     CheckSum{},
6086   }
6087   err := gshCtx.gshSetupHomeDir()
6088   return gshCtx, err
6089 }
6090 func (gsh*GshContext)gshellh(gline string)(bool){
6091   ghist := gsh.CmdCurrent
6092   ghist.WordCount = os.Getwd()
6093   ghist.WchDirIdx = len(gsh.ChdirHistory)-1
6094   //fmt.Printf("--D--ChdirHistory(%d)\n",len(gsh.ChdirHistory))
6095   ghist.StartAt = time.Now()
6096   rusagev1 := Getrusagev()
6097   gsh.CmdCurrent.FoundFile = []string{}
6098   fin := gsh.tgshellh(gline)
6099   name := Getname()
6100   ghist.RusageSub = RusageSubv(rusagev2,rusagev1)
6101   ghist.EndAt = time.Now()
6102   ghist.CmdLine = gline
6103   ghist.FoundFile = gsh.CmdCurrent.FoundFile
6104
6105   /* record it but not show in list by default
6106   if len(gline) == 0 {
6107     continue
6108   }
6109   if gline == "hi" || gline == "history" { // don't record it
6110     continue
6111   }
6112   */
6113   gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6114   return fin
6115 }
6116 // as name="main">Main loop</a>
6117 func script(gshCtxGiven *GshContext) (_ GshContext) {
6118   gshCtxtBuf,err0 := setupGshContext()
6119   if err0 {
6120     return gshCtxtBuf;
6121   }
6122   gshCtx := &gshCtxtBuf
6123
6124 //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
6125 //resmap()
6126
6127 /*
6128 if false {
6129   gsh_getlinev,with_exgetline :=
6130   which("PATH",[]string{"which","gsh-getline","-s"})
6131   if with_exgetline {
6132     gsh_getlinev[0] = toFullPath(gsh_getlinev[0])
6133     gshCtx.GetLine = toFullPath(gsh_getlinev[0])
6134   }else{
6135     fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6136   }
6137 */
6138
6139
6140 ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
6141 gshCtx.CommandHistory = append(gshCtx.CommandHistory,ghist0)
6142
6143 prevline := ""
6144 skipping := false
6145 for hix := len(gshCtx.CommandHistory); ; {
6146   gline := gshCtx.getline(hix,skipping,prevline)
6147   if skipping {
6148     if strings.Index(gline,"if") == 0 {
6149       fmt.Println("fin\n");
6150       skipping = false;
6151     }else{
6152       //fmt.Printf("%s\n",gline);
6153     }
6154   }
6155   continue
6156   if strings.Index(gline,"if") == 0 {
6157     //fmt.Printf("--D-- if start: %s\n",gline);
6158     skipping = true;
6159   }
6160   if false {
6161     os.Stdout.Write([]byte("gotline:"))
6162     os.Stdout.Write([]byte(gline))
6163     os.Stdout.Write([]byte("\n"))
6164   }
6165   gline = strsubst(gshCtx,gline,true)
6166   if false {
6167     fmt.Printf("fmt.Println %%v\n");
6168     fmt.Println("fmt.Println %%s - %s\n",gline)
6169     fmt.Println("fmt.Println %%U - %s\n",gline)
6170     fmt.Println("fmt.Println \"%s\" - %s\n",gline)
6171     os.Stdout.Write("-")
6172     os.Stdout.Write([]byte(gline))
6173     fmt.Println("\n")
6174   }
6175   /*
6176   // should be cared in substitution ?
6177   if < len(gline) && gline[0] == '!' {
6178     xline, set, err := searchHistory(gshCtx,gline)
6179     if err {
6180       continue
6181     }
6182     if set {
6183       // set the line in command line editor
6184     }
6185     gline = xline
6186   */
6187   fin := gshCtx.gshellh(gline)
6188   if fin {
6189     break;
6190   }
6191   prevline = gline;
6192   hix++;
}

```



```

6372 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/background-repeat">repeat</a></span>
6373 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
6374 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
6375 <a href="https://mdn-web-docs-e3-as-west-2.amazonaws.com/MDN-Browser-Compatibility-Report-2020.pdf">MDN Browser Compatibility Report (2020)</a>
6376
6377 Go language (August 2020 / Go 1.15)
6378 <a href="https://golang.org">The Go Programming Language</a>
6379 <a href="https://golang.org/pkg/">Packages</a>
6380 <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>
6381
6382 <a href="https://stackoverflow.com/">Stackoverflow</a>
6383 <!--
6384 <iframe loading="lazy" src="https://golang.org" width="100%" height="300"></iframe>
6385 -->
6386 </div></details>
6387 /*
6388 <details id="html-src" onclick="frame_open()"><summary>Raw Source</summary><div>
6389 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
6390 <details id="gsh-whole-view"><summary>Whole file</summary>
6391 <a name="whole"></a>
6392 <span id="src-frame"></span><!-- a window to show source code -->
6393 </details>
6394 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
6395 <a name="style-src-view"></a>
6396 <span id="gsh-style-view"></span>
6397 </details>
6398 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
6399 <a name="script-src-view"></a>
6400 <span id="gsh-script-view"></span>
6401 </details>
6402 <div>
6403 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
6404 <a name="gsh-data-frame"></a>
6405 <span id="gsh-data-view"></span>
6410 </details>
6411 </div></details>
6412 /*
6413 */
6414 <div id="GshFooter0"></div>
6415 <!-- 2020-09-17 SatoxITS, visible script { -- -
6416 <details><summary>GJScript</summary>
6417 <script>document.write(<!-- font-family:Georgia; -->)
6418 <pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJScript!'); }
6419   gjtest1();
6420 </pre>
6421 <script>
6422   gjs=document.getElementById('gjscript_1');
6423   //eval(gjs.innerHTML);
6424   //gjs.outerHTML = '';
6425 </script>
6426 </details><!-- ----- END-OF-VISIBLE-PART ----- > -->
6427
6428 <!--
6429 // 2020-0906 added,
6430 https://developer.mozilla.org/en-US/docs/Web/CSS/-index
6431 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6432 -->
6433 <span id="GshGrid">(&^_)</span><small>(Hit j k l h)</small></span>
6434
6435 <span id="GStat"><br>
6436 </span>
6437 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6438 <span id="GTop"></span>
6439 <div id="GshellPlane" onclick="showGShellPlane()";></div>
6440 <div id="RawTextViewer"></div>
6441 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6442
6443 <style id="GshStyleDef">
6444   #lineheight{table,tr,td {
6445     line-height:12px;
6446     padding:4px;
6447     spacing:0;
6448     border:12px;
6449   }
6450   textareas{lineNumber {
6451     font-size:12px;
6452     font-family:monospace,Courier New;
6453     padding:4px;
6454     wrap:off;
6455   }
6456   #RawTextViewer{
6457     z-index:0;
6458     position:fixed; top:0px; left:0px;
6459     width:100%; xxxheight:50px; xheight:0px;
6460     overflow:auto;
6461     color:#fff; background-color:rgba(128,128,256,0.2);
6462     font-size:12px;
6463     spellcheck:false;
6464   }
6465   #RawTextViewerClose{
6466     z-index:0;
6467     position:fixed; top:-100px; left:-100px;
6468     color:#fff; background-color:rgba(128,128,256,0.2);
6469     font-size:20px; font-family:Georgia;
6470     white-space:pre;
6471   }
6472   #xxxGshellPlane{
6473     z-index:0;
6474     position:fixed; top:0px; left:0px;
6475     width:100%; height:50px;
6476     overflow:auto;
6477     color:#fff; background-color:rgba(128,128,256,0.3);
6478     font-size:12px;
6479   }
6480   #xxxGTop{
6481     z-index:9;
6482     opacity:1.0;
6483     position:fixed; top:0px; left:0px;
6484     width:320px; height:20px;
6485     color:#fff; background-color:rgba(32,32,160,0.15);
6486     color:#fff; font-size:12px;
6487   }
6488   #xxxGPos{
6489     z-index:12;
6490     position:fixed; top:0px; left:0px;
6491     opacity:1.0;
6492     width:640px; height:30px;
6493     color:#fff; background-color:rgba(0,0,0,0.2);
6494     color:#fff; font-size:12px;
6495   }
6496   #GMmenu{
6497     z-index:100000000;
6498     position:fixed; top:250px; left:0px;
6499     opacity:1.0;
6500     width:100px; height:100px;
6501     color:#fff; background-color:rgba(0,0,0,0);
6502     color:#fff; font-size:16px; font-family:Georgia;
6503     background-repeat:no-repeat;
6504   }
6505   #xxxGStat{
6506     z-index:8;
6507     opacity:0.0;
6508     position:fixed; top:20px; left:0px;
6509     xwidth:640px;
6510     width:100%; height:90px;
6511     color:#fff; background-color:rgba(0,0,128,0.04);
6512     color:#fff; font-size:16px; font-family:Georgia;
6513     font-size:20px; font-family:Georgia;
6514   }
6515   #GLog{
6516     z-index:10;
6517     position:fixed; top:50px; left:0px;
6518     opacity:1.0;
6519     width:640px; height:60px;
6520     color:#fff; background-color:rgba(0,0,128,0.10);
6521     font-size:12px;
6522   }
6523   #GshGrid {
6524     z-index:11;
6525     opacity:0.0;
6526     position:fixed; top:0px; left:0px;
6527     width:320px; height:30px;
6528     color:#fff; font-size:16px;
6529     color:#ff9; font-size:16px;
6530   }
6531   xbody {display:none;}
6532   .gsh-link{color:green;}
6533   #gsh {border-width:1px; margin:0;padding:0;}
6534   #gsh {font-family:monospace,Courier New;color:#ddff;font-size:8px;}
6535   #gsh {border-color:#100000;}
6536   #gsh header{height:100px;background-image:url(GShell-Logo00.png);}
6537   #GshMenu{font-size:14pt;color:#c44; }
6538   #GshMenu{font-size:14pt;color:#c44; }
6539   #GshMenu1{font-size:14pt;color:#2a2;padding:4px; text-align:right;

```



```

7080 width:166px; height:20px;
7081 font-family:monospace;
7082 font-size:1.0em;
7083 line-height:1.0;
7084 color:#fff; background-color:rgba(0,0,64,0.2);
7085 text-align:center;
7086 vertical-align:middle;
7087 }
7088 .GJIcon{
7089 display:inline;
7090 position:relative;
7091 top:0px; left:1px;
7092 border:2px solid #000;
7093 margin:0px; padding:0px;
7094 width:13.2px; height:13.2px;
7095 border-radius:2px;
7096 font-family:Georgia;
7097 font-size:13.2px;
7098 line-height:1.0;
7099 white-space:nowrap;
7100 color:#fff; background-color:rgba(32,32,160,0.8);
7101 text-align:center;
7102 vertical-align:middle;
7103 text-shadow:0px 0px;
7104 }
7105 .GTText:focus{
7106 color:#fff !important;
7107 background-color:rgba(32,32,160,0.8) !important;
7108 line-height:1.0;
7109 }
7110 .GTText{
7111 display:inline;
7112 position:relative;
7113 top:0px; left:0px;
7114 border:0px solid #000; margin:0px; padding:0px;
7115 width:160px; height:160px;
7116 border:0px;
7117 font-family:courier New,monospace !important;
7118 font-size:8pt;
7119 line-height:1.0;
7120 white-space:pre;
7121 color:#fff; xbackground-color:rgba(0,0,64,0.5);
7122 background-color:rgba(32,32,128,0.8) !important;
7123 }
7124 .GMMode{
7125 display:inline;
7126 position:relative;
7127 top:0px; left:0px;
7128 border:0px solid #000; border-radius:0px;
7129 margin:0px; padding:0px;
7130 width:280px; height:20px;
7131 font-size:9pt;
7132 line-height:1.0;
7133 white-space:nowrap;
7134 color:#fff; background-color:rgba(0,0,64,0.7);
7135 text-align:left;
7136 vertical-align:middle;
7137 }
7138 
```

```

7139 <script id="gsh-script">
7140 // 2020-0909 added, permanet local storage
7141 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7142 var MyHistory =
7143   window.localStorage;
7144 MyHistory = Permanent.getitem('MyHistory')
7145 if( MyHistory == null ){ MyHistory = "" }
7146 d = new Date()
7147 MyHistory = d.getTime()/1000+ "+document.URL+\n" + MyHistory
7148 Permanent.setitem('MyHistory',MyHistory)
7149 //Permanent.setitem('MyWindow',window)
7150
7151 var GJLog_Win = null
7152 var GJLog_Tab = null
7153 var GJLog_Stat = null
7154 var GJLog_Text = null
7155 var GJWin_Mode = null
7156 var PProductInterval = 0
7157
7158 var GJ_FactoryID = -1
7159 var GJFactory = null
7160 if( document.getElementById('GJFactory_0') ){
7161   GJFactory_0.height = 0
7162   GJFactory = e
7163   e.setAttribute('class','GJFactory')
7164   var GJ_FactoryID = 0
7165 }else{
7166   GJFactory = GJFactory_1
7167   var GJ_FactoryID = 1
7168 }
7169
7170 function GJFactory_Destroy(){
7171   gjf = GJFactory
7172   //gjf = document.getElementById('GJFactory')
7173   //alert('gjf='+gjf)
7174   if( gjf != null ){
7175     if( gjf.childNodes != null ){
7176       for( i = 0; i < gjf.childNodes.length; i++ ){
7177         gjf.removeChild(gjf.childNodes[i])
7178       }
7179     }
7180     gjf.innerHTML = ''
7181     gjf.style.width = 0
7182     gjf.style.height = 0
7183     gjf.removeAttribute('style')
7184   }
7185   GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
7186   window.clearInterval(PProductInterval)
7187   return '- Destroy: work product destroyed'
7188 }else{
7189   return '-- Destroy: work product not exist'
7190 }
7191
7192 var TransMode = false
7193 var OnkeyControl = false
7194 var OnkeyShift = false
7195 var OnkeyAlt = false
7196 var OnkeyJ = false
7197 var OnkeyK = false
7198 var OnkeyL = false
7199
7200 function GJWin_OnKeyUp(ev){
7201   keycode = ev.code;
7202   if( keycode == 'ShiftLeft' ){
7203     OnkeyShift = false
7204   }else{
7205     if( keycode == 'ControlLeft' ){
7206       OnkeyControl = false
7207     }else{
7208       if( keycode == 'AltLeft' ){
7209         OnkeyAlt = false
7210       }else{
7211         if( keycode == 'KeyJ' ){ OnkeyJ = false }else
7212           if( keycode == 'KeyK' ){ OnkeyK = false }else
7213             if( keycode == 'KeyL' ){ OnkeyL = false }else
7214               {
7215               }
7216         }
7217       ev.preventDefault()
7218     }
7219   }
7220   function and(a,b){ if(a){ if(b){ return true; } return false; } }
7221   function GJWin_OnKeyDown(ev){
7222     keycode = ev.code;
7223     mode = '';
7224     key = '';
7225     if( keycode == 'ControlLeft' ){
7226       OnkeyControl = true
7227       ev.preventDefault()
7228     }else{
7229       if( keycode == 'ShiftLeft' ){
7230         OnkeyShift = true
7231         ev.preventDefault()
7232       }else{
7233         if( keycode == 'AltLeft' ){
7234           ev.preventDefault()
7235           OnkeyAlt = true
7236           return;
7237         }else{
7238           if( keycode == 'Backquote' ){
7239             TransMode = !TransMode
7240             ev.preventDefault()
7241           }else{
7242             if( and(keycode == 'Space', OnKeyShift) ){
7243               TransMode = !TransMode
7244               ev.preventDefault()
7245             }else{
7246               if( keycode == 'ShiftRight' ){
7247                 TransMode = !TransMode
7248               }else{
7249                 if( keycode == 'Escape' ){
7250                   TransMode = true
7251                   ev.preventDefault()
7252                 }else{
7253                   if( keycode == 'Enter' ){
7254                     TransMode = false
7255                     /ev.preventDefault()
7256                   }
7257                 }
7258               }
7259             }
7260           }
7261         }
7262       }
7263     }
7264   }
7265 }
```

```

7257 }
7258 if( keycode == 'KeyJ' ){ OnKeyJ = true }else
7259 if( keycode == 'KeyK' ){ OnKeyK = true }else
7260 if( keycode == 'KeyL' ){ OnKeyL = true }else
7261 {
7262 }
7263
7264 if( ev.altKey ){ key += 'Alt+' }
7265 if( OnKeyControl ){ key += 'Ctrl+' }
7266 if( OnKeyShift ){ key += 'Shift+' }
7267 if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
7268 if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
7269 if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
7270 key += keycode
7271
7272 if( TransMode ){
7273 //mod = "(343\201\202r"
7274 Jakutf8 = new Uint8Array([0343,0201,0202]);
7275 ut8dec = new TextDecoder();
7276 Jak = ut8dec.decode(Jakutf8);
7277 mode = "[ " + JA + " ]";
7278
7279 }else{
7280   mode = '[---]'
7281 }
7282 //--- /gjmode.innerHTML = "[---]"
7283 GJWin_Mode.innerHTML = mode + ' ' + key
7284 //alert('Key:' +keycode)
7285 ev.stopPropagation()
7286 //ev.preventDefault()
7287
7288 function GJWin_OnScroll(ev){
7289   x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
7290   y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
7291   GJLog_append('OnScroll: x=' + x + ',y=' + y)
7292   document.addEventListener('scroll',GJWin_OnScroll)
7293   function GJWin_OnResize(ev){
7294     w = window.innerWidth
7295     h = window.innerHeight
7296     GJLog_append('OnResize: w=' + w + ',h=' + h)
7297   }
7298   window.addEventListener('resize',GJWin_OnResize)
7299
7300 var DragStartX = 0
7301 var DragStartY = 0
7302 function GJWin_DragStart(ev){
7303   ev.preventDefault();
7304   //why this is the grabbing position
7305   this.style.position = 'fixed'
7306   x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
7307   y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
7308   GJLog_Stat.value = 'DragStart: x=' + x + ',y=' + y
7309 }
7310
7311 function GJWin_Drag(ev){
7312   x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
7313   this.style.left = x - DragStartX
7314   this.style.top = y - DragStartY
7315   this.style.zIndex = '30000'
7316   this.style.position = 'fixed'
7317   x = this.getBoundingClientRect().left.toFixed(0)
7318   y = this.getBoundingClientRect().top.toFixed(0)
7319   GJLog_Stat.value = 'x=' + x + ',y=' + y
7320   ev.preventDefault()
7321   ev.stopPropagation()
7322 }
7323
7324 function GJWin_DragEnd(ev){
7325   x = ev.clientX; y = ev.clientY
7326   //x = ev.pageX; y = ev.pageY
7327   this.style.left = x - DragStartX
7328   this.style.top = y - DragStartY
7329   this.style.zIndex = '30000'
7330   this.style.position = 'fixed'
7331   if( true ){
7332     console.log("Dropped: "+this.nodeName+'#'+this.id+' x=' + x + ' y=' + y
7333     + ' parent=' + this.parentNode.id)
7334   }
7335   x = this.getBoundingClientRect().left.toFixed(0)
7336   y = this.getBoundingClientRect().top.toFixed(0)
7337   GJLog_Stat.value = 'x=' + x + ',y=' + y
7338   ev.preventDefault()
7339   ev.stopPropagation()
7340 }
7341
7342 // 2020-09-15 let every object have console view!
7343 var GJ_ConsoleID = 0
7344 var PrevReport = new Date()
7345 function GJLog_StatUpdate(){
7346   txa = GJLog_Stat;
7347   if( txa == null ){
7348     return;
7349   }
7350   tmLap0 = new Date();
7351   p = txa.parentNode;
7352   pw = txa.getBoundingClientRect().width;
7353   ph = txa.getBoundingClientRect().height;
7354   tx1 = "#"+p.id+" pw=" + pw + ",ph=" + ph + '\n';
7355   tx1 += "#"+p.id+" pw*px+, ph*ph+\n";
7356   tx1 += "#"+p.id+" pw*px+, ph*ph+\n";
7357
7358   w = txa.getBoundingClientRect().width;
7359   h = txa.getBoundingClientRect().height;
7360   tx1 += "w=" + w + ", h=" + h + '\n';
7361   tx1 += 'w=' + w + ', h=' + h + '\n';
7362
7363   //txa.value += '\n';
7364   //txa.value += DateShort() + '\n';
7365   tx1 += '\n';
7366   tx1 += DateShort() + '\n';
7367   tmLap1 = new Date();
7368
7369   tx1.value += tx1;
7370   tmLap2 = new Date();
7371
7372   // vertical centering of the last line
7373   sHeight = txa.scrollHeight - 30; // depends on the font-size
7374   tmLap3 = new Date();
7375
7376   txa.scrollTop = sHeight; // depends on the font-size
7377   tmLap4 = new Date();
7378
7379   now = tmLap0.getTime();
7380   if( PrevReport == 0 || 10000 <= now - PrevReport ){
7381     PrevReport = now;
7382     console.log('StatBarUpdate:');
7383     + ' txw=' + txw.value.length + ' byte,' +
7384     + ' tisize=' + (tmLap4 - tmLap0) + ' ms (' +
7385     + ' tadd=' + (tmLap2 - tmLap1) + ', ' +
7386     + ' hcal=' + (tmLap3 - tmLap2) + ', ' +
7387     + ' scrl=' + (tmLap4 - tmLap3) + ')';
7388   };
7389 }
7390
7391 GJWin_StatUpdate = GJLog_StatUpdate;
7392 function GJ_showFimel(wid){
7393   //e = document.getElementById(wid);
7394   //console.log(wid.id+' .value.length=' + wid.value.length)
7395   if( e != null ){
7396     if( e.value = DateShort());
7397   }else{
7398     // should remove the Listener
7399   }
7400
7401 function GJ_NewConsole(wname){
7402   this.value += 'resized:' + '\n'
7403 }
7404
7405 function GJ_Console(wname){
7406   wid = wname + '-' + GJ_ConsoleID
7407   GJ_ConsoleID += 1
7408
7409   GJFactory.style.setProperty('width',360+'px'); //GJFsize
7410   GJFactory.style.setProperty('height',320+'px')
7411   e = GJFactory;
7412   console.log('GJFa #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7413
7414 if( GJFactory.innerHTML == "" ){
7415   GJFactory.innerHTML = '<' + 'H3>' + GJ_Factory_ + GJ_FactoryID + '<' + '/H3><' + 'hr>\n'
7416 }else{
7417   GJFactory.innerHTML += '<' + 'hr>\n'
7418 }
7419 gjwin = GJLog_Win = document.createElement('span')
7420 gjwin.id = wid
7421 gjwin.setAttribute('class', 'GJWin')
7422 gjwin.setAttribute('data-type', 'tr')
7423 gjwin.addEventListener('dragstart',GJWin_DragStart)
7424 gjwin.addEventListener('drag',GJWin_Drag)
7425 gjwin.addEventListener('dragend',GJWin_Drag)
7426 gjwin.addEventListener('dragover',GJWin_DragIgnore)
7427 gjwin.addEventListener('dragenter',GJWin_DragIgnore)
7428 gjwin.addEventListener('dragleave',GJWin_DragIgnore)
7429 gjwin.addEventListener('dragexit',GJWin_DragIgnore)
7430 gjwin.addEventListener('drop',GJWin_DragIgnore)
7431 gjwin.addEventListener('keydown',GJWin_OnKeyDown)
7432
7433 gjtab = GJLog_Tab = document.createElement('textarea')

```

```

7434 gjtab.addEventListener('keydown',GJWin_OnKeyDown)
7435 gjtab.style.readonly = true
7436 gjtab.setAttribute('disabled', false
7437 gjtab.value = wid
7438 gjtab.id = wid + 'Tab'
7439 gjtab.setAttribute('class','GJTab')
7440 gjtab.setAttribute('spellcheck','false')
7441 gjwin.appendChild(gjtab)
7442
7443 gjstat = GJLog_Stat = document.createElement('textarea')
7444 gjstat.addEventListener('keydown',GJWin_OnKeyDown)
7445 gjstat.id = wid + '_Stat'
7446 gjstat.value = DateShort()
7447 gjstat.setAttribute('class','GJStat')
7448 gjstat.setAttribute('spellcheck','false')
7449 gjwin.appendChild(gjstat)
7450
7451 gjicon = document.createElement('span')
7452 gjicon.addEventListener('keydown',GJWin_OnKeyDown)
7453 gjicon.id = wid + '_Icon'
7454 gjicon.innerHTML = '<G>font color="#F44">J</font>'
7455 gjicon.setAttribute('class','GJIcon')
7456 gjicon.setAttribute('spellcheck','false')
7457 gjwin.appendChildChild(gjicon)
7458
7459 gjtext = GJLog_Text = document.createElement('textarea')
7460 gjtext.addEventListener('keydown',GJWin_OnKeyDown)
7461 gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7462 gjtext.addEventListener('resize',GJwin_OnResizeTextarea)
7463 gjtext.id = wid + '_Text'
7464 gjtext.setAttribute('class','GJText')
7465 gjtext.setAttribute('spellcheck','false')
7466 gjwin.appendChild(gjtext)
7467
7468
7469 // user's mode as of TME
7470 gjmode = GJWin_Mode = document.createElement('textarea')
7471 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7472 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7473 gjmode.id = wid + '_Mode'
7474 gjmode.setAttribute('class','GJMode')
7475 gjmode.setAttribute('spellcheck','false')
7476 gjmode.innerHTML = '---'
7477 gjwin.appendChild(gjmode)
7478
7479 gjwin.zIndex = 30000
7480 GJFactory.appendChild(gjwin)
7481
7482 gjtab.scrollTop = 0
7483 gjstat.scrollTop = 0
7484
7485 //x = gjwin.getBoundingClientRect().left.toFixed(0)
7486 //y = gjwin.getBoundingClientRect().top.toFixed(0)
7487 //gjwin.style.position = 'static'
7488 //gjwin.style.left = 0
7489 //gjwin.style.top = 0
7490
7491 //update = {'wid':value:DateShort()}}
7492 //update = {'GJ_showTime1':wid});
7493 // 2020-09-13 this causes memory leak
7494 //ProductInterval = window.setInterval(update,200)
7495 //ProductInterval = window.setInterval(GJWin_StatUpdate,200)
7496 //ProductInterval = window.setInterval(GJ_showTime1,200,wid);
7497 PProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
7498 return update
7499 }
7500 function xxxGJF_StripClass(){
7501 GJLog_Win.style.removeProperty('width')
7502 GJLog_Tab.style.removeProperty('width')
7503 GJLog_Stat.style.removeProperty('width')
7504 GJLog_Text.style.removeProperty('width')
7505 return "Stripped classes"
7506 }
7507 function isElem(id){
7508 return document.getElementById(id) != null
7509 }
7510 function GJLog_append(...args){
7511 txt = GJLog_Text;
7512 if( txt == null ){
7513 return; // maybe GJLog element is removed
7514 }
7515 logs = args.join(',')
7516 txt.value += logs + '\n'
7517 txt.scrollTop = txt.scrollHeight
7518 //GJLog_Stat.value = DateShort()
7519 }
7520 //window.addEventListener('time',GJLog_StatUpdate)
7521 function test_GJ_Console(){
7522 window.setInterval(GJLog_StatUpdate,1000);
7523 GJ_NewConsole('GJ_Console');
7524 e = GJfactory;
7525 console.log('GJF0 #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7526 e.style.width = 360; //GJFsize
7527 e.style.height = 320;
7528 console.log('GJF0 #' + e.id + ' to w=' + e.style.width + ', h=' + e.style.height)
7529 }
7530 /// test_GJ_Console();
7531
7532 var StopConsoleLog = true
7533 // 2020-09-15 added,
7534 // log should be saved to permanent memory
7535 // const px = new Proxy(console.log,{ alert() })
7536 console.log = console.log || px.log;
7537 __console_log = console.log;
7538 __console_warn = console.warn;
7539 __console_error = console.error;
7540 __console_exception = console.exception;
7541 // should pop callstack info.
7542 console.exception = function(...args){
7543 __console_exception(...args);
7544 alert('-- got console.exception("'+args+'")')
7545 };
7546 console.error = function(...args){
7547 __console_error(...args);
7548 alert('-- got console.error("'+args+'")')
7549 };
7550 console.warn = function(...args){
7551 __console_warn(...args);
7552 alert('-- got console.warn("'+args+'")')
7553 };
7554 console.info = function(...args){
7555 alert('-- got console.info("'+args+'")')
7556 __console_info(...args)
7557 };
7558 console.xxxlog = function(...args){ // rewrite xxxlog to log to enable it
7559 __console_log(...args);
7600 If(StopConsoleLog) {
7601 return;
7602 }
7603 if( 0 <= args[0].indexOf('!') ){
7604 //alert('-- got console.log("'+args+'")')
7605 }
7606 }
7607 GJLog_append(...args)
7608
7609 //document.getElementById('GshFaviconURL').href = GShellFavicon
7610 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
7611 //document.getElementById('GshfaviconURL').href = ITSMOREQR
7612 //document.getElementById('GshfaviconURL').href = GSHELLogo
7613
7614 // id of GShell HTML elements
7615 var E_BANNER = "GshBanner" // banner element in HTML
7616 var E_FOOTER = "GshFooter" // footer element in HTML
7617 var E_MENU = "gsh-menu-index" // Colang code of GShell
7618 var E_GOCODE = "gsh-gocode" // Colang code of GShell
7619 var E_TODO = "gsh-todo" // TODO of GShell
7620 var E_DICT = "gsh-dict" // Dictionary of GShell
7621
7622 function bannerElem(){ return document.getElementById(E_BANNER); }
7623 function bannerStyleFunc(){ return bannerElem().style; }
7624 var bannerStyle = bannerStyleFunc()
7625 function GshGetImages(){
7626 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7627 bannerStyle.backgroundColor = "url("+GShellLogo+)";
7628 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+"";
7629 //bannerStyle.backgroundImage = "url("+GShellFavicon+"";
7630 //GMenu.style.BackgroundImage = "url("+GShellInsideIcon+"";
7631 //showFooter();
7632 }
7633 function GshInsideIconSetup(){
7634 GMenu.style.BackgroundImage = "url("+GShellInsideIcon+"";
7635 GMenu.style.zIndex = 1000000;
7636 //GMenu.style.left = window.innerWidth - 100
7637 GMenu.style.left = 0;
7638 GMenu.style.top = window.innerHeight - 90; // - 200
7639 window.addEventListener('resize',GshInsideIconSetup);
7640 }
7641
7642 function footerElem(){ return document.getElementById(E_FOOTER); }
7643 function footerStyle(){ return footerElem().style; }
7644 //footerElem().style.backgroundImage="url("+ITSMOREQR+"";
7645 //footerStyle().backgroundImage = "url("+ITSMOREQR+"";
7646
7647 function html_fold(e){
7648 if( e.innerHTML == "Fold" ){
7649 e.innerHTML = "Unfold"
7650 document.getElementById('gsh-menu-exit').innerHTML=""
7651 }
7652 }

```

```

7611     document.getElementById('GshStatement').open=false
7612     GshFeatures.open = true
7613     document.getElementById('html-src').open=false
7614     document.getElementById(E_GINDEX).open=false
7615     document.getElementById(E_GOCODE).open=false
7616     document.getElementById(E_TODO).open=false
7617     document.getElementById('references').open=false
7618   }
7619   e.innerHTML = "Fold"
7620   document.getElementById('gsh').style.display=true
7621   GshFeatures.open = true
7622   document.getElementById(E_GINDEX).open=true
7623   document.getElementById(E_GOCODE).open=true
7624   document.getElementById(E_TODO).open=true
7625   document.getElementById('references').open=true
7626 }
7627 }
7628 function html_pure(e){
7629   if( e.innerHTML == "Pure" ){
7630     document.getElementById('gsh').style.display = false
7631     e.innerHTML = "Unpure"
7632   }else{
7633     document.getElementById('gsh').style.display=false
7634     //document.style.display = "none"
7635     e.innerHTML = "Pure"
7636   }
7637 }
7638 }
7639 var bannerIsStopping = false
7640 //https://www.w3schools.com/jscript/met_win_setinterval.asp
7641 function shiftBG(){
7642   bannerIsStopping = !bannerIsStopping
7643   bannerStyle.backgroundPosition = "0 0";
7644 }
7645 // status should be inherited on Window Fork(), so use the status in DOM
7646 function himl_stop(e,toggle){
7647   if( toggle ){
7648     if( e.innerHTML == "Stop" ){
7649       bannerIsStopping = true
7650     }else{
7651       bannerIsStopping = false
7652     }
7653   }else{
7654     // update Javascript variable from DOM status
7655     if( e.innerHTML == "Stop" ){ // shown if it's running
7656       bannerIsStopping = false
7657     }else{
7658       bannerIsStopping = true
7659     }
7660   }
7661 }
7662 }
7663 }
7664 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7665 //html_stop(bannerBlem(),false) // onInit.
7666
7667 //https://www.w3schools.com/jscript/met_win_setinterval.asp
7668 var banShift=0;
7669 function consoleLog(str){
7670   //console.log(str);
7671 }
7672 function shiftBanner(){
7673   var now = new Date().getTime();
7674   bpos = ((now/10)*10000).toFixed(0)+"px" + " 0px";
7675   if( !bannerIsStopping ){
7676     bannerStyle.backgroundPosition = bpos;
7677     //GshBanner.style.setProperty('background-position',bpos,'!important');
7678     banShift += 1;
7679     console.log("banShift <"+GshBanner.nodeName+">> "+banShift
7680     +' now='+(now%10)
7681     +' /'+ ' stops'+bannerIsStopping
7682     +' pos='+bpos
7683     +' -> '+bannerStyle.backgroundPosition);
7684   }
7685 }
7686 }
7687 function Banner_init(){
7688   console.log("-- Banner Shift init.");
7689   window.setInterval(shiftBanner,10); // onInit.
7690 }
7691
7692 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open(</a>
7693 // from embedded html to standalone page
7694 var MyChildren = 0
7695 function html_fork(){
7696   ResetPerMon();
7697   ResetAffiliation();
7698   RemoveSharingCanvas();
7699   GIPfactory_Destroy();
7700   MyChildren += 1
7701   Wind = document.getElementById('gsh-WinId').innerHTML + ". " + MyChildren;
7702   newWin = window.open(" "+Wind,"");
7703   src = document.getElementById("gsh");
7704   src.innerHTML = src.innerHTML;
7705   newWin.document.write("/<"+"html>\n");
7706   newWin.document.write(srchtml);
7707   newWin.document.write("<"+"html>\n");
7708   newWin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7709   newWin.document.getElementById(' gsh-WinId').innerHTML = Wind;
7710   newWin.document.close();
7711   newWin.focus();
7712 }
7713 function html_close(){
7714   window.close()
7715 }
7716 function win_jump(win){
7717   //win = window.top;
7718   win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
7719   if( win == null ){
7720     //console.log("jump to window.opener("+win+")\n")
7721   }else{
7722     console.log("jump to window.opener("+win+)\n")
7723     win.focus();
7724   }
7725 }
7726
7727 // 0.2.9 2020-0902 created chekcsum of HTML
7728 CRC32UNIX = 0x04c11db7 // Unix cksum
7729 function byteCRC32add(bigrcc,octstr,octlen){
7730   var crc = new Int32Array(1)
7731   crc[0] = bigrc
7732
7733   let oi = 0
7734   for( ; oi < octlen; oi++ ){
7735     var oct = new Int8Array(1)
7736     oct[0] = octstr[oi]
7737     for( let bi = 0; bi < 8; bi++ ){
7738       //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+":"+"bi+"]\n")
7739       ovf1 = crc[0] < 0 ? 1 : 0
7740       ovf2 = oct[0] < 0 ? 1 : 0
7741       ovf = ovf1 ^ ovf2
7742       oct[0] <<= 1
7743       crc[0] <<= 1
7744       if( ovf ){ crc[0] ^= CRC32UNIX }
7745     }
7746   }
7747   //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oi+"."+octlen+"\n")
7748   return crc[0];
7749 }
7750 function strCRC32add(bigrcc,str1,strlen){
7751   var crc = new Uint32Array(1)
7752   crc[0] = bigrc
7753   var code = new Uint8Array(strlen);
7754   for( let i = 1 < strlen; i++ ){
7755     code[i] = str1.charCodeAt(i) // not charAt() !!!!
7756     //console.log("== "+code[i].toString(16)+" <== "+str1[i]+\n")
7757   }
7758   crc[0] = byteCRC32add(crc,code,strlen)
7759   //console.log("--CRC32 stdAdd return crc="+crc[0]+\n")
7760   return crc[0];
7761 }
7762 function byteCRC32end(bigrcc,len){
7763   var crc = new Uint32Array(1)
7764   crc[0] = bigrc
7765   var sllen = new Uint8Array(4)
7766   let li = 0
7767   for( ; li < 4; ){
7768     sllen[li] = len
7769     li += 1
7770     len >= 8
7771     if( len == 0 ){
7772       break
7773     }
7774   }
7775   crc[0] = byteCRC32add(crc[0],sllen,li)
7776   crc[0] ^= 0xFFFFFFFF
7777   return crc[0];
7778 }
7779 function strCRC32(str1,len){
7780   var crc = new Uint32Array(1)
7781   crc[0] = 0
7782   crc[0] = strCRC32add(0,str1,len)
7783   crc[0] = byteCRC32end(crc[0],len)
7784   //console.log("--CRC32 "+crc[0]+" "+len+"\n")
7785   return crc[0]
7786 }

```

```

7788 DestroyGJLink = null; // to be replaced
7789 DestroyFooter = null; // to be defined
7790 DestroyEventSharingDevice = function dummy(){}
7791 Destroy_WirtualDesktop = function(){}
7792 DestroyNavButtons = function(){}
7793
7794 function getSourceText(){
7795   if( DestroyFooter != null ) DestroyFooter();
7796   version = document.getElementById('GshVersion').innerHTML
7797   sfavicon = document.getElementById('GshFaviconURL').href;
7798   sbanner = document.getElementById('GshBanner').style.backgroundImage;
7799   spositi = document.getElementById('GshBanner').style.backgroundPosition;
7800
7801   if( document.getElementById('GJC_1') != null ) GJC_1.remove()
7802   if( DestroyGJLink != null ) DestroyGJLink();
7803   DestroyEventSharingCodeview();
7804   Destroy_WirtualDesktop();
7805   GshTopbar.innerHTML = "";
7806   DestroyNavButtons();
7807   DestroyNavButtons();
7808   ResetPerfMon();
7809   ResetAfvView();
7810   Reset_ShadingCanvas();
7811
7812 // these should be removed by CSS selector or class, after seava to non-printed attribute
7813 GshBanner.removeAttribute('style');
7814 document.getElementById('GshMenuSign').removeAttribute('style');
7815 styleGMen = GMenu.getAttribute('style')
7816 GMENU.removeAttribute('style');
7817 styleGStat = GStat.getAttribute('style')
7818 styleGTop = GTop.getAttribute('style')
7819 GTop.removeAttribute('style');
7820 styleGshGrid = GshGrid.getAttribute('style')
7821 GshGrid.removeAttribute('style');
7822 //styleGLog = GLog.getAttribute('style');
7823 //GLog.removeAttribute('style');
7824 //styleGLog = GLog.getAttribute('style');
7825 //GLog.removeAttribute('style');
7826 //GLog.innerHTML = '';
7827 styleRawPlane = GshellPlane.getAttribute('style')
7828 GshellPlane.removeAttribute('style')
7829 styleRawTextviewer = RawTextViewer.getAttribute('style')
7830 RawTextViewer.removeAttribute('style')
7831 styleRawTextviewerClose = RawTextViewerClose.getAttribute('style')
7832 RawTextViewerClose.removeAttribute('style')
7833
7834 GshFaviconURL.href = "";
7835 if( iselem('ConfigIcon') ) ConfigIcon.srC = "";
7836
7837 //it seems that innerHTML and outerHTML generate style="" for these (?)
7838 //GshBanner.removeAttribute('style');
7839 //GshMenuSign.removeAttribute('style');
7840 //GshFooter.removeAttribute('style');
7841 //GshMenuSign.removeAttribute('style');
7842 //GshBanner.style="";
7843 GshMenuSign.style="";
7844
7845 textareas = document.createElement("textareas")
7846 srohtml = document.getElementById("gsh").outerHTML;
7847 //textareas = document.createElement("textareas")
7848 // 2020-091 ?? ... this causes inserting style="" to Banner and Footer,
7849 // with Chromium/? after reloading from file//!
7850 textareas.innerHTML = srohtml
7851
7852 <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7853 var rawtext = textareas.value
7854 //textarea.destroy()
7855 //rawtext = gsh.textContent // this removes #include <FILENAME> too
7856 var orgtext = ""
7857 + "<"+html">\n" // lost preamble text
7858 + rawtext
7859 + "<"+html">\n" // lost trail text
7860 ;
7861 tlen = orgtext.length
7862 //console.log("getSourceText: length='"+tlen+"\n")
7863 document.getElementById('GshFaviconURL').href = sfavicon;
7864
7865 document.getElementById('GshBanner').style.backgroundImage = sbanner;
7866 document.getElementById('GshBanner').style.backgroundPosition = spositi;
7867
7868 GStat.setAttribute('style',styleGStat)
7869 GMENU.setAttribute('style',styleGMen)
7870 GTop.setAttribute('style',styleGTop)
7871 //GLog.setAttribute('style',styleGLog)
7872 //GPos.setAttribute('style',styleGPos)
7873 GshGrid.setAttribute('style',styleGshGrid)
7874 GshGrid.setAttribute('style',styleGshPlane)
7875 RawTextViewer.setAttribute('style',styleRawTextviewer)
7876 RawTextViewerClose.setAttribute('style',styleRawTextviewerClose)
7877 canontext = orgtext.replace(' style=""','');
7878 // open="" too
7879 return canontext
7880
7881 }
7882 function getDigest(){
7883   var text = ""
7884   text = getSourceText()
7885   var digest = strCRC32(text,tlen) + " " + tlen
7886   return { text, digest }
7887 }
7888
7889 function html_digest(){
7890   version = document.getElementById('GshVersion').innerHTML
7891   ln = (text, digest) = getDigest()
7892   alert('cksum: ' + digest + " " + version)
7893 }
7894 function charsin(str,char){
7895   ln = 0;
7896   for( i = 0; i < str.length; i++ ){
7897     if( str.charCodeAt(i) == char.charCodeAt(0) )
7898       ln++;
7899   }
7900   return ln;
7901 }
7902
7903 //class digestElement extends HTMLElement { }
7904 //<script>customElements.define('digest',digestElement)</script>
7905 function showDigest(e){
7906   result = "version=" + GshVersion.innerHTML + '\n'
7907   result += "lines=" + e.dataset.lines + '\n'
7908   result += "length=" + e.dataset.length + '\n'
7909   result += "crc12u=" + e.dataset.crc32u + '\n'
7910   result += "time=" + e.dataset.time + '\n';
7911
7912   alert(result)
7913 }
7914
7915 function html_sign(e){
7916   if( RawTextViewer.style.zIndex == 1000 ){
7917     hideRawTextViewer()
7918     return
7919   }
7920   GshTopbar.innerHTML = "";
7921   ResetPerfMon();
7922   ResetAfvView();
7923   Reset_ShadingCanvas();
7924   DestroyNavButtons();
7925   DestroyNavButtons();
7926   DestroyEventSharingCodeview();
7927   Destroy_WirtualDesktop();
7928   GJFactory_Destroy();
7929   if( DestroyGJLink != null ) DestroyGJLink();
7930   //gsh_digest.innerHTML = '';
7931   text = getSourceText() // the original text
7932   tlen = text.length
7933   digest = strCRC32(text,tlen)
7934   //gsh_digest.innerHTML = digest + " " + tlen
7935   //text = getSourceText() // the text with its digest
7936   Lines = charsin(text, '\n')
7937
7938   name = "gsh"
7939   sid = name + "-digest"
7940   d = new Date()
7941   signedat = d.getTime()
7942
7943   sign = '/'+sid+'<'+span+'\n'
7944   + ' id="'+ sid + '\n'
7945   + ' class="digest"\n'
7946   + ' data-target-id="'+name+'\n'
7947   + ' data-length="'+ tlen + digest + '\n'
7948   + ' data-lines="'+ Lines + '\n'
7949   + ' data-times="'+ signedat + '\n'
7950   + '><' + '/span>\n'+ '\n'
7951
7952   text = sign + text
7953
7954   txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>' +
7955   + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">' +
7956   for( i = 1; i <= Lines; i++ ){
7957     txthml += i.toString() + '\n'
7958   }
7959   txthml += ''
7960   + '<' + '/textarea>' +
7961   + '<' + '/td>' + 'td>' +
7962   + '<' + 'textarea cols=150 rows=' + Lines + ' spellcheck="false"' +
7963
7964

```

```

7965     + ' class="LineNumbered">'  
7966     + 'text + '<+'>textareas'  
7967     + '<+' + '</td><+' + '</tr><' + '</table>'  
7968  
7969 for( i = 1; i <= 30; i++ ){  
7970     txthtml += '<br>\n'  
7971 }  
7972 RawTextViewer.innerHTML = txthtml  
7973 RawTextViewer.spellcheck = false // (spelcheck above seems ineffective)  
7974  
7975     btn = e  
7976     e.style.color = "rgba(128,128,255,0.9)";  
7977     y = e.getBoundingClientRect().top.toFixed(0)  
7978     /<+>eBoundingClientRect().height.toFixed(0)  
7979     RawTextViewer.style.top = Number(y) + 30  
7980     RawTextViewer.style.left = 100;  
7981     RawTextViewer.style.height = window.innerHeight - 20;  
7982     //RawTextViewer.style.opacity = 1.0;  
7983     //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";  
7984     RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";  
7985     RawTextViewer.style.zIndex = 1000;  
7986     RawTextViewer.style.display = true;  
7987  
7988 if( RawTextViewerClose.style == null ){  
7989     RawTextViewerClose.style = "";  
7990 }  
7991 RawTextViewerClose.style.top = Number(y) + 10  
7992 RawTextViewerClose.style.left = 100;  
7993 RawTextViewerClose.style.zIndex = 1001;  
7994  
7995 ScrollToElement(CurElement,RawTextViewerClose)  
7996 }  
7997 function hideRawTextViewer(){  
7998     RawTextViewer.style.left = 10000;  
7999     RawTextViewer.style.zIndex = -100;  
8000     RawTextViewer.style.opacity = 0.0;  
8001     RawTextViewer.style = null;  
8002     RawTextViewer.innerHTML = "";  
8003  
8004     GshMenuSign.style.color = "rgba(255,128,128,1.0)";  
8005     RawTextViewerClose.style.top = 0;  
8006     RawTextViewerClose.style = null  
8007 }  
8008  
8009 // source code view  
8010 function frame_close(){  
8011     srcframe = document.getElementById("src-frame");  
8012     srcframe.innerHTML = "";  
8013     //srcframe.style.cols = 1;  
8014     srcframe.style.rows = 1;  
8015     srcframe.style.height = 0;  
8016     srcframe.style.display = false;  
8017     src = document.getElementById("SrcTextarea");  
8018     src.innerHTML = "";  
8019     //src.col = 0  
8020     src.row = 0  
8021     src.display = false  
8022     //alert("--closed--")  
8023 }  
8024 //<-- | <span onclick="html_view();">Source</span> -->  
8025 //<!-- | <span onclick="frame_close();">Sourceclose</span> -->  
8026 //<!-- | <span>Download</span> -->  
8027 function frame_open(){  
8028     GshToolBar.innerHTML = "";  
8029     ResetPencilIcon();  
8030     ResetAfview();  
8031     Reset_ShadingCanvas();  
8032     DestroyIndexBar();  
8033     DestroyNaviButtons();  
8034     if( DestroyFooter != null ) DestroyFooter();  
8035     document.getElementById('GshFaviconURL').href = "";  
8036     oldsrc = document.getElementById("GENSRC");  
8037     if( oldsrc != null ){  
8038         //alert("--I--(erasing old text)")  
8039         oldsrc.innerHTML = "";  
8040         return  
8041     }  
8042     //alert("--I--(no old text)")  
8043     styleBanner = GshBanner.getAttribute("style")  
8044     GshBanner.removeAttribute("style")  
8045     if( document.getElementById('GJC_1') ){ GJC_1.remove() }  
8046  
8047     GshFaviconURL.href = "";  
8048     if( iselem('ConfigIcon') ) ConfigIcon.src = "";  
8049     GStat.removeAttribute('style')  
8050     GStatId.removeAttribute('style')  
8051     GStatSign.removeAttribute('style')  
8052     GStatSign.removeAttribute('style')  
8053     //GPos.removeAttribute('style')  
8054     //GPos.innerHTML = "";  
8055     //GLog.removeAttribute('style')  
8056     //GLog.innerHTML = "";  
8057     GLog.removeAttribute('style')  
8058     GLog.removeAttribute('style')  
8059     GShellPlane.removeAttribute('style')  
8060     RawTextViewer.removeAttribute('style')  
8061     RawTextViewerClose.removeAttribute('style')  
8062  
8063     if( DestroyGJLink != null ) DestroyGJLink();  
8064     GJFactory.Destroy()  
8065     Destroy_VirtualDesktop();  
8066     DestroyEventSharingCodeview();  
8067  
8068     src = document.getElementById("gsh");  
8069     archtml = src.outerHTML  
8070     srcframe = document.getElementById("src-frame");  
8071     srcframe.innerHTML = ""  
8072     + '<+' + cite_id+'\GENSRC`>\n"  
8073     + '<+' + style + '\n"  
8074     + '#GENSRC textarea{tab-size:4;}\n"  
8075     + "#GENSRC textarea{-o-tab-size:4;}\n"  
8076     + "#GENSRC textarea{-moz-tab-size:4;}\n"  
8077     + "#GENSRC textarea{spellcheck:false;}\n"  
8078     + '</' + style + '\n"  
8079     + '<+' + textarea_id="SrcTextarea" cols=100 rows=20 class="gsh-code" spellcheck="false">>'  
8080     + '</' + html + '\n' // lost preamble text  
8081     + srchtml  
8082     + '<+' + html + '\n' // lost trail text  
8083     + '</' + textarea + '\n"  
8084     + '</' + cite + '-> \n';  
8085  
8086     //srcframe.style.cols = 80;  
8087     //srcframe.style.rows = 80;  
8088  
8089     GshBanner.setAttribute('style',styleBanner)  
8090 }  
8091 function fill_CSSView(){  
8092     part = document.getElementById('GshStyleDef')  
8093     view = document.getElementById('gsh-style-view')  
8094     view.innerHTML = ""  
8095     + '<+' + textarea cols=100 rows=20 class="gsh-code">>'  
8096     + part.innerHTML  
8097     + '</' + textarea + "  
8098 }  
8099 function fill_JavaScriptView(){  
8100     jspart = document.getElementById('gsh-script')  
8101     view = document.getElementById('gsh-script-view')  
8102     view.innerHTML = ""  
8103     + '<+' + textarea cols=100 rows=20 class="gsh-code">>  
8104     + jspart.innerHTML  
8105     + '</' + textarea + "  
8106 }  
8107 function fill_DataView(){  
8108     part = document.getElementById('gsh-data')  
8109     view = document.getElementById('gsh-data-view')  
8110     view.innerHTML = ""  
8111     + '<+' + textarea cols=100 rows=20 class="gsh-code">>  
8112     + part.innerHTML  
8113     + '</' + textarea + "  
8114 }  
8115 function jump_to_StyleView(){  
8116     jsview = document.getElementById('html-src')  
8117     jsview.open = true  
8118     jsview = document.getElementById('gsh-style-frame')  
8119     jsview.open = true  
8120     fill_CSSView()  
8121 }  
8122 function jump_to_JavaScriptView(){  
8123     jsview = document.getElementById('html-src')  
8124     jsview.open = true  
8125     jsview = document.getElementById('gsh-script-frame')  
8126     jsview.open = true  
8127     fill_JavaScriptView()  
8128 }  
8129 function jump_to_DataView(){  
8130     jsview = document.getElementById('html-src')  
8131     jsview.open = true  
8132     jsview = document.getElementById('gsh-data-frame')  
8133     jsview.open = true  
8134     fill_DataView()  
8135 }  
8136 function jump_to_WholeView(){  
8137     jsview = document.getElementById('html-src')  
8138     jsview.open = true  
8139     jsview = document.getElementById('gsh-whole-view')  
8140     jsview.open = true  
8141     frame_open()

```

```

8142 }
8143 function html_view(){
8144     html_stop();
8145
8146     banner = document.getElementById('GshBanner').style.backgroundImage;
8147     footer = document.getElementById('GshFooter').style.backgroundImage;
8148     document.getElementById('GshBanner').style.backgroundImage = "";
8149     document.getElementById('GshBanner').style.backgroundPosition = "";
8150     document.getElementById('GshFooter').style.backgroundImage = "";
8151
8152 //srcwin = window.open("", "CodeView2");
8153 //srcwin = window.open("", "", "");
8154 srcwin.document.write("<span id=\"gsh\"\n");
8155
8156 src = document.getElementById("gsh");
8157 srcwin.document.write("<"+style>\n");
8158 srcwin.document.write("textarea(tab-size:4);\n");
8159 srcwin.document.write("textarea(~o-tab-size:4;}\n");
8160 srcwin.document.write("textarea(-moz-tab-size:4;}\n");
8161 srcwin.document.write("</span>\n");
8162 srcwin.document.write("<h2>\n");
8163 srcwin.document.write("<"+span onclick="window.close();>Close</span> | \n");
8164 //srcwin.document.write("<"+span onclick="html_stop();>Run</span>\n");
8165 srcwin.document.write("</h2>\n");
8166 srcwin.document.write("<"+div id="gsh-src-src" cols=100 rows=60>");
8167 srcwin.document.write("<"+html>\n");
8168 srcwin.document.write("gsh\\>");
8169 srcwin.document.write(src.innerHTML);
8170 srcwin.document.write("<"+span>"+html>\n");
8171 srcwin.document.write("</"+textarea>\n");
8172
8173 document.getElementById('GshBanner').style.backgroundImage = banner;
8174 document.getElementById('GshFooter').style.backgroundImage = footer
8175
8176 sty = document.getElementById("GshStyleDef");
8177 srcwin.document.write("<"+style>\n");
8178 srcwin.document.write(sty.innerHTML);
8179 srcwin.document.write("<"+style>\n");
8180
8181 run = document.getElementById("gsh-script");
8182 srcwin.document.write("<"+script>\n");
8183 srcwin.document.write(run.innerHTML);
8184 srcwin.document.write("<"+script>\n");
8185
8186 srcwin.document.write("<"+span><"+html>\n"); // gsh span
8187 srcwin.document.close();
8188 srcwin.focus();
8189 }
8190 GSH = document.getElementById("gsh")
8191 //GSH.onclick = "alert('Ouch!')";
8192 //GSH.css = {"background-color:#eef;"}
8193 //GSH.style = "background-color:#eef;";
8194 //GSH.style.display = false;
8195 //alert('Ouch0!');
8196 //GSH.style.display = true;
8197
8198 // 2020-08-04 created, tentative
8199 //document.addEventListener('keydown', jgshCommand);
8200 CurElement = GshStatement
8201 CurElement = GshMenu
8202 MemElement = GshMenu
8203
8204 function nextSib(e){
8205     n = e.nextSibling;
8206     for( i = 0; i < 100; i++ ){
8207         if( n == null ){
8208             break;
8209         }
8210         if( n.nodeName == "DETAILS" ){
8211             return n;
8212         }
8213         n = n.nextSibling;
8214     }
8215     return null;
8216 }
8217 function prevSib(e){
8218     n = e.previousSibling;
8219     for( i = 0; i < 100; i++ ){
8220         if( n == null ){
8221             break;
8222         }
8223         if( n.nodeName == "DETAILS" ){
8224             return n;
8225         }
8226         n = n.previousSibling;
8227     }
8228     return null;
8229 }
8230 function setColor(e,eName,eColor){
8231     if( e.hasChildNodes() ){
8232         s = e.childNodes;
8233         if( s != null ){
8234             for( ci = 0; ci < s.length; ci++ ){
8235                 if( s[ci].nodeName == eName ){
8236                     s[ci].style.color = eColor;
8237                     s[ci].style.backgroundColor = eColor;
8238                     //s[ci].style.backgroundColor = eColor;
8239                     break;
8240                 }
8241             }
8242         }
8243     }
8244 }
8245 // https://docs.microsoft.com/en-us/previous-versions/hh781509(v=vs.85)
8246 function showCurElementPosition(ev){
8247 // if( document.getElementById("GPos") == null ){
8248 //     return;
8249 //
8250 // }
8251 // if( GPos == null ){
8252 //     return;
8253 //
8254     e = CurElement
8255     y = e.getBoundingClientRect().top.toFixed(0)
8256     x = e.getBoundingClientRect().left.toFixed(0)
8257
8258     h = ev + " "
8259     h += "y=" + y + ", " + 'x=' + x + " -- "
8260     h += "w=" + window.innerWidth + ", h=" + window.innerHeight + " -- "
8261 //GPos.test = h
8262 //GPos.innerHTML = h
8263 // GPos.innerHTML = h
8264 }
8265
8266 function zero2(n){
8267     if( n < 10 ){
8268         return '0' + n;
8269     }else{
8270         return n;
8271     }
8272 }
8273 function DateHourMin(){
8274     d = new Date();
8275     //return '02d:@2d.sprintf(d.getHours(),d.getMinutes());
8276     return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
8277 }
8278 function DateShort0(d){
8279     return d.getDate();
8280     + '/' + zero2(d.getMonth())
8281     + ' ' + zero2(d.getDate());
8282     + ':' + zero2(d.getHours());
8283     + ':' + zero2(d.getMinutes());
8284     + ':' + zero2(d.getSeconds());
8285 }
8286 function DateShort(){
8287     return DateShort0(new Date());
8288 }
8289 function DateLong0(ms){
8290     d = new Date();
8291     d.setTime(ms);
8292     return Dateshort0(d)
8293     + '.' + d.getMilliseconds()
8294     + '.' + d.getTimezoneOffset()/60
8295     + '.' + d.getTime() + '.' + d.getMilliseconds()
8296 }
8297 function DateLong(){
8298     return DateLong0(new Date());
8299 }
8300 function GshellMenu(e){
8301 //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8302 //showShellplane()
8303 ConfigClick();
8304 }
8305 // placements of planes
8306 function GshellResizeX(ev){
8307 //if( document.getElementById("GMENU") != null ){
8308 //    GMENU.style.left = '0';
8309 //    GMENU.style.left = window.innerWidth - 100
8310 //    GMENU.style.top = window.innerHeight - 90 - 200
8311 //    console.log("place GMENU "+GMENU.style.left+" "+GMENU.style.top)
8312 //}
8313 GMENU.style.width = window.innerWidth
8314 //if( document.getElementById("GPos") != null ){
8315 //    GPos.style.width = window.innerWidth
8316 //    GPos.style.height = window.innerHeight - 30; //GPos.style.height
8317 //}
8318 }

```

```

8319 //if( document.getElementById("GLog") != null ){
8320 //  GLog.style.width = window.innerWidth
8321 //  GLog.innerHTML = ""
8322 //}if( document.getElementById("GLog") != null ){
8323 //GLog.innerHTML = "Resize: w=" + window.innerWidth +
8324 //", h=" + window.innerHeight
8325 //}
8326 //showCurElementPosition(ev)
8327 }
8328 function GShellResizeX("[RESIZE]"){
8329 }
8330 window.onresize = GShellResize
8331 var prevNode = null
8332 var LogMouseMoveOverElement = false;
8333 function GJSH_OnMouseMove(ev){
8334   if( LogMouseMoveOverElement == false ){
8335     return;
8336   }
8337   x = ev.clientX
8338   y = ev.clientY
8339   d = new Date()
8340   t = d.getTime() / 1000
8341   if( !document.elementFromPoint ){
8342     e = document.elementFromPoint(x,y)
8343     if( e != null ){
8344       if( e == prevNode ){
8345         console.log('Mo:'+'at+'+'('+x+','+y+')' +
8346           '+e.nodeType+' '+'e.tagName+' #' + e.id)
8347       }else{
8348         console.log('Mo:'+'at+'+'('+x+','+y+')' +
8349           '+e.nodeType+' '+'e.tagName+' #' + e.id)
8350       prevNode = e
8351     }else{
8352       console.log(t+'('+x+','+y+') no element')
8353     }
8354   }else{
8355     console.log(t+'('+x+','+y+') no elementFromPoint')
8356   }
8357 }
8358 }
8359 window.addEventListener('mousemove',GJSH_OnMouseMove);
8360
8361 function GJSH_OnMouseMoveScreen(ev){
8362   x = ev.screenX
8363   y = ev.screenY
8364   d = new Date()
8365   t = d.getTime() / 1000
8366   console.log(t+'('+x+','+y+') no elementFromPoint')
8367 }
8368 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
8369
8370 function ScrollToElement(oe,ne){
8371   ne.scrollTop+=10
8372   nx = ne.getBoundingClientRect().left.toFixed(0)
8373   ny = ne.getBoundingClientRect().top.toFixed(0)
8374   //GLog.innerHTML = "["+ny+","+nx+"]"
8375   //window.scrollTo(0,0)
8376
8377   GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
8378   GshGrid.style.left = "-250px";
8379   GshGrid.style.zindex = 0
8380   if( false ){
8381     oy = oe.getBoundingClientRect().top.toFixed(0)
8382     ox = oe.getBoundingClientRect().left.toFixed(0)
8383     y = e.getBoundingClientRect().top.toFixed(0)
8384     x = e.getBoundingClientRect().left.toFixed(0)
8385     window.scrollTo(x,y)
8386     ny = e.getBoundingClientRect().top.toFixed(0)
8387     nx = e.getBoundingClientRect().left.toFixed(0)
8388     //GLog.innerHTML = "["+ny+","+ox+"]->["+ny+","+x+"]->["+ny+","+nx+"]"
8389   }
8390 }
8391 function ShowGShellPlane(){
8392   if( GShellPlane.style.zIndex == 0 ){
8393     GShellPlane.style.zIndex = 1000;
8394     GShellPlane.style.left = 30;
8395     GShellPlane.style.height = 320;
8396     GShellPlane.innerHTML = DateLong() + "<br>" +
8397     "-- History --<br>" + MyHistory;
8398   }else{
8399     GShellPlane.style.zIndex = 0;
8400     GShellPlane.style.left = 0;
8401     GShellPlane.style.height = 50;
8402     GShellPlane.innerHTML = "";
8403   }
8404 }
8405 var SuppressGShell = false
8406 function jshCommand(keyevent){
8407   if( SuppressGShell ){
8408     return
8409   }
8410   key = keyevent
8411   keycode = key.code
8412   //GStat.style.width = window.innerWidth
8413   GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8414
8415   console.log("JSGsh-Key:" +keycode+"(^-)/")
8416   if( keycode == "Slash" ){
8417     GLog.append("Folding all elements")
8418     e = document.elementFromPoint(x,y)
8419     console.log('('+x+','+y+')' +'e.nodeType+' +'e.tagName+' #' + e.id)
8420   }else{
8421     if( keycode == "Digit0" ) // fold side-bar
8422     // zero page
8423     showShellPlane();
8424   }else{
8425     if( keycode == "Digit1" ) // fold side-bar
8426     primary.style.width = "94%"
8427     secondary.style.width = "6%"
8428     secondary.style.opacity = 0
8429     GStat.innerHTML = "[Single Column View]"
8430   }else{
8431     if( keycode == "Digit2" ) // unfold side-bar
8432     primary.style.width = "58%"
8433     secondary.style.width = "36%"
8434     secondary.style.opacity = 1
8435     GStat.innerHTML = "[Double Column View]"
8436   }else{
8437     if( keycode == "KeyY" ){ // fold/unfold all
8438       html_fold(GshMenuFold);
8439       location.href = "#"+CurElement.id;
8440     }else{
8441       if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8442         CurElement.open = !CurElement.open;
8443       }else{
8444         if( CurElement.open == true )
8445           CurElement.open = false
8446         else
8447           if( CurElement.open == false )
8448             CurElement.open = true
8449       }
8450     }
8451     if( CurElement.open == false )
8452       e = CurElement
8453       GJLog.append("Current Element: " + e + "<br>" +
8454         "+name=" + e.nodeName + ", "
8455         "+id=" + e.id + ", "
8456         "+children=" + e.childNodes.length + ", "
8457         "+parent=" + e.parentNode.id + "<br>" +
8458         "+text=" + e.textContent)
8459     GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8460   }
8461 }
8462 if( keycode == "KeyM" ){ // memory the position
8463   CurElement = CurElement
8464 }
8465 if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8466   e = nextSib(CurElement)
8467   if( e != null )
8468     setColor(CurElement,"SUMMARY","#ffff")
8469     setColor(e,"SUMMARY","#8f8") // should be complement ?
8470   oe = CurElement
8471   CurElement = e
8472   //location.href = "#"+e.id;
8473   ScrollToElement(oe,e)
8474 }
8475 else
8476 if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8477   oe = CurElement
8478   e = prevSib(CurElement)
8479   if( e != null )
8480     setColor(CurElement,"SUMMARY","#ffff")
8481     setColor(e,"SUMMARY","#8f8") // should be complement ?
8482   CurElement = e
8483   //location.href = "#"+e.id;
8484   ScrollToElement(oe,e)
8485 }
8486 else
8487   e = document.getElementById("GshBanner")
8488   if( e != null ){
8489     setColor(CurElement,"SUMMARY","#ffff")
8490     CurElement = e
8491     ScrollToElement(oe,e)
8492   }
8493   e = document.getElementById("primary")
8494   if( e != null ){
8495     setColor(CurElement,"SUMMARY","#ffff")
8496     CurElement = e
8497   }

```

```

8496     ScrollToElement(oe,e)
8497   }
8498 }
8499 }
8500 if( keycode == "KeyR" ){
8501   location.reload()
8502 }
8503 if( keycode == "KeyJ" ){
8504   GshGrid.style.top = '120px';
8505   GshGrid.innerHTML = '>_<{down}';
8506 }
8507 if( keycode == "KeyK" ){
8508   GshGrid.style.top = '0px';
8509   GshGrid.innerHTML = '<_>{Up}';
8510 }
8511 if( keycode == "KeyH" ){
8512   GshGrid.style.left = '0px';
8513   GshGrid.innerHTML = '(_){Left}';
8514 }
8515 if( keycode == "KeyL" ){
8516   //GLog.innerHTML+=
8517   GJLog_append(
8518     'screen='+screen.width+'px'+<br>+
8519     'window='+window.innerWidth+'px'+<br>
8520   )
8521   GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8522   GshGrid.innerHTML = '(_){Right}';
8523 }
8524 if( keycode == "KeyS" ){
8525   html_stop(GshMenuStop,true)
8526 }
8527 if( keycode == "KeyF" ){
8528   html_fork()
8529 }
8530 if( keycode == "KeyC" ){
8531   window.close()
8532 }
8533 if( keycode == "KeyD" ){
8534   html_digest()
8535 }
8536 if( keycode == "KeyV" ){
8537   e = document.getElementById('gsh-digest')
8538   if( e != null ){
8539     showDigest(e)
8540   }
8541 }
8542 }
8543 showCurElementPosition("[+key.code+]"--);
8544 //If document.getElementById("GPos") != null {
8545 //  /GPos.innerHTML += "[+key.code+]"--;
8546 //}
8547 //GShellResizeX("[+key.code+]"--);
8548 }
8549 var initGSKC = false;
8550 function Gshell_initKeyCommands(){
8551   if( initGSKC ) return; initGSKC = true;
8552   GShellResizeX("[INIT]");
8553   DisplaySize = '-- Display: '
8554   + 'screen='+screen.width+'px, '+ 'window='+window.innerWidth+'px';
8555   let {text, digest} = getDigest()
8556   //GLog.innerHTML +=
8557   GJLog_append();
8558   GShell: ' + GshVersion.innerHTML + '\n' +
8559   '-- Digest: ' + digest + '\n' +
8560   DisplaySize
8561   // "<br>" + "-- LastVisit:<br>" + MyHistory
8562   GShellResizeX(null);
8563 }
8564 //GShell initKeyCommands();
8565
8566 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8567 //Convert a string into an ArrayBuffer
8568 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8569 function str2ab(str) {
8570   const buf = new ArrayBuffer(str.length);
8571   const bufView = new Uint8Array(buf);
8572   for (let i = 0, strLen = str.length; i < strLen; i++) {
8573     bufView[i] = str.charCodeAt(i);
8574   }
8575   return buf;
8576 }
8577 function importPrivateKey(pem) {
8578   const binaryDerString = window.atob(pemContents);
8579   const binaryDer = str2ab(binaryDerString);
8580   return window.crypto.subtle.importKey(
8581     "pkcs8",
8582     binaryDer,
8583     {
8584       name: "RSA-PSS",
8585       modulusLength: 2048,
8586       publicExponent: new Uint8Array([1, 0, 1]),
8587       hash: "SHA-256",
8588     },
8589     true,
8590     ["sign"]
8591   );
8592 }
8593 //importPrivateKey(ppem)
8594
8595 //Key = {}
8596 //buf = 'abc'
8597 //enc = "xxxxxxxx"; //crypto.publicEncrypt(key,buf)
8598 //b64 = btoa(enc)
8599 //dec = atob(b64)
8600 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8601 </script>
8602 </div>
8603 </div>
8604 </div>
8605 </div>
8606 </div>
8607 </div>
8608 </div>
8609 </div>
8610 </div>
8611 </div>
8612 </div>
8613 </div>
8614 </div>
8615 </div>
8616 </div>
8617 </div>
8618 </div>
8619 </div>
8620 </div>
8621 </div>
8622 </div>
8623 </div>
8624 </div>
8625 </div>
8626 </div>
8627 </div>
8628 </div>
8629 </div>
8630 </div>
8631 </div>
8632 </div>
8633 </div>
8634 </div>
8635 </div>
8636 </div>
8637 </div>
8638 </div>
8639 </div>
8640 </div>
8641 </div>
8642 </div>
8643 </div>
8644 </div>
8645 </div>
8646 </div>
8647 </div>
8648 </div>
8649 </div>
8650 </div>
8651 </div>
8652 </div>
8653 </div>
8654 </div>
8655 </div>
8656 </div>
8657 </div>
8658 </div>
8659 </div>
8660 </div>
8661 </div>
8662 </div>
8663 </div>
8664 </div>
8665 </div>
8666 </div>
8667 </div>
8668 </div>
8669 </div>
8670 </div>
8671 </div>
8672 </div>

```

```

8673     base.xinc = 4;
8674     base.yinc = 4;
8675     base.scrx = 0;
8676     base.scry = 0;
8677     base.pgw = 100;
8678     base.phg = 100;
8679     moveGwindow(PackmonGo_2);
8680
8681     ctx = PackmonGo_3_Canvas.getContext('2d');
8682     pgo = 0.3;
8683     ctx.fillStyle = 'rgba(64,64,255,'+pgo+')';
8684     ctx.fillRect(0,0,pgw,phg);
8685     base = PackmonGo_3;
8686     gsh.appendChild(base);
8687     base.style.zindex = 1002;
8688     base.style.position = "fixed";
8689     base.style.left = 200 + 'px';
8690     base.style.top = 0 + 'px';
8691     base.xinc = 4;
8692     base.yinc = 4;
8693     base.scrx = 0;
8694     base.scry = 0;
8695     base.soft = 0;
8696     base.pgw = 100;
8697     base.phg = 100;
8698     moveGscreen(PackmonGo_3);
8699
8700     ctx = PackmonGo_4_Canvas.getContext('2d');
8701     pgw = phg = 400;
8702     ctx.beginPath();
8703     ctx.fillStyle = 'rgba(0,0,0,0)';
8704     ctx.arc(200,200,200,Math.PI/2,true);
8705     ctx.stroke();
8706     pgo = 0.4;
8707     ctx.fillStyle = 'rgba(255,31,32,'+pgo+')';
8708     ctx.fillRect(0,0,pgw,phg);
8709     base = PackmonGo_4;
8710     gsh.appendChild(base);
8711     base.style.zindex = 1002;
8712     base.style.position = "fixed";
8713     base.style.left = 200 + 'px';
8714     base.style.top = 0 + 'px';
8715     base.xinc = 4;
8716     base.yinc = 4;
8717     base.scrx = 0;
8718     base.scry = 0;
8719     base.soft = 50000;
8720     base.pgw = phg;
8721     base.phg = phg;
8722     moveGscreen(PackmonGo_4);
8723 }
8724 function movePGwindow(base){
8725   if(!stopPackmonFlag ){
8726     return;
8727   }
8728   x = parseInt(base.style.left);
8729   y = parseInt(base.style.top);
8730   w = window.innerWidth;
8731   h = window.innerHeight;
8732   if(x < 0 || w-base.pgw < x ){
8733     base.xinc = -base.xinc;
8734   }
8735   x += base.xinc;
8736   if(y < 0 || h-base.phg < y ){
8737     base.yinc = -base.yinc;
8738   }
8739   y += base.yinc;
8740   base.style.left = x + 'px';
8741   base.style.top = y + 'px';
8742   //console.log('PG x=' +x +',y=' +y);
8743   //window.setTimeout(movePG,10);
8744 }
8745 function movePGscreen(base){
8746   if(!stopPackmonFlag ){
8747     return;
8748   }
8749   sw = screen.width;
8750   sh = screen.height;
8751   d = new Date();
8752   s = d.getTime(); // milli-seconds
8753   sx = ((base.soft) % 10000) * (screen.width / 10000);
8754   sy = (((s-base.soft) % 5000) * (screen.height / 5000));
8755   x = sx - window.screenX;
8756   y = sy - window.screenY;
8757   base.style.left = x + 'px';
8758   base.style.top = y + 'px';
8759 }
8760 function movePGscreenCircle(base){
8761   if(!stopPackmonFlag ){
8762     return;
8763   }
8764   sw = screen.width;
8765   sh = screen.height;
8766   pgw = base.pgw;
8767   phg = base.phg;
8768   d = new Date();
8769   s = d.getTime(); // milli-seconds
8770   ds = ((base.soft) % 10000) * (screen.width / 10000);
8771   vs = ((base.soft) % 5000) * (screen.height / 5000);
8772   vsw = pgw + sw + pgw; // delay
8773   vsh = phg + sh + phg;
8774   sx = -pgw + vsw * ((ds % 10000)/10000);
8775   sy = -phg + vsh * ((ds % 10000)/10000);
8776   x = sx - window.screenX;
8777   y = sy - window.screenY;
8778   base.style.left = x + 'px';
8779   base.style.top = y + 'px';
8780 }
8781 if(PackmonInit == false){
8782   //window.setInterval(movePG,300);
8783   window.setInterval(movePGwindow,10,PackmonGo_1);
8784   window.setInterval(movePGwindow,10,PackmonGo_2);
8785   window.setInterval(movePGscreen,10,PackmonGo_3);
8786   window.setInterval(movePGscreen,10,PackmonGo_4);
8787   window.setInterval(movePGscreenCircle,10,PackmonGo_4);
8788   window.addEventListener('click',stopPackMon);
8789   PackmonInit = true;
8790   stopPackmonFlag = false;
8791 }
8792 function PackmonGo_Setup(e){
8793   stopPackmon();
8794   spawnPackmonGo();
8795   if( e != null ){
8796     e.stopPropagation()
8797   }
8798 }
8799 }
880 PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
881 if( PackmonGo_Section.open == true ){
882   PackmonGo_Setup();
883 }
884 </script>
885 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
886 <input id="PackmonGo_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
887 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
888 <span id="PackmonGo_WorkCodeView"></span>
889 <script id="PackmonGo_WorkCodeView">
890   function PackmonGo_openWorkCodeView(){
891     showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
892   }
893   PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
894 </script>
895 PackmonGo_openWorkCodeView(); // should be invoked by an event
896 </details>
897 <!-- Template_WorkCodeSpan } -->
898 <!-- //<span> -->
899 //<!-- ===== Work } ===== -->
900
901 <!-- ===== Work { ===== -->
902 <span id="SightGlass_WorkCodeSpan">
903   <!-- =====-->
904   <details id="SightGlass"><summary>SightGlass</summary>
905   <h2>SightGlass</h2>
906   <details><summary>Setter</summary>
907   <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
908   <div id="SightGlass_l_BGScrofOffset" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
909   <div id="SightGlass_l_BGPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
910   <div id="SightGlass_l_Wheel" class="SightGlass_TextData">Wheel</div>
911   </details>
912   <div id="SightGlass_l_Window" class="SightGlass_Window" draggable="true"></div>
913   <style>
914     .SightGlass_Textdata {
915       color:#000;
916       font-size:10pt;
917     }
918     .SightGlass_Window {
919       zoom:2;
920       resize:both;
921       width:600px;
922       height:320px;
923     }
924   </style>
925 
```

```

8850 background-color:rgba(200,200,200,0.5);
8851 background-size:200px;
8852 background-size:1280px 720px;
8853 background-image:url("WD-WallPaper03.png");
8854 }
8855 <body>
8856 scroll-behavior:smooth;
8857 }
8858 </style>
8859 <script>
8860 //
8861 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
8862 var SGScrInitX = 0;
8863 var SGScrInitY = 0;
8864 var SG_initX = 0;
8865 var SG_initY = 0;
8866 function SG_Adjust(){
8867     xy = window.screenX + ' ' + window.screenY;
8868     wh = window.innerWidth + ' ' + window.innerHeight;
8869     xywh = 'xy' + ' ' + xy + ' ' + wh + ' ' + wh + ' ';
8870     if( SightGlass.open ) SightGlass_1_BPosition.innerHTML = 'Browser: ' + xywh;
8871
8872     sg = SightGlass_1_Window;
8873     sgr = sg.getBoundingClientRect();
8874     xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
8875     wh = sgr.width + ' ' + sgr.height;
8876     xywh = 'xy' + ' ' + xy + ' ' + wh + ' ' + wh + ' ';
8877     if( SightGlass.open ) SightGlass_1_GPosition.innerHTML = 'SiGlass: ' + xywh;
8878
8879 //SightGlass_1_Window.style.backgroundPosition = sgr.left+'px ' + sgr.top+'px';
8880 if( SG_initX == 0 ){
8881     SGScrInitX = window.screenX;
8882     SGScrInitY = window.screenY;
8883     //SightGlass_1_Window.style.backgroundSize = '200%';
8884     //SightGlass_1_Window.style.backgroundImage = 'url("WD-WallPaper03.png")';
8885     SG_initX = sgr.left;
8886     SG_initY = sgr.top;
8887 }
8888 dx = SG_initX - sgr.left;
8889 dy = SG_initY - sgr.top;
8890
8891 dx = SGScrInitX - window.screenX;
8892 dy = SGScrInitY - window.screenY;
8893 scroff = 'Screen: '+sgr.left+'px '+sgr.top+'px';
8894 if( SightGlass.open ) SightGlass_1_BGScrlOffset.innerHTML = scroff;
8895 dx += dx;
8896 dy += dy;
8897
8898 bgpos = dx+'px ' + dy+'px';
8899 SightGlass_1_Window.style.backgroundPosition = bgpos;
8900 if( SightGlass.open ) SightGlass_1_BGPosition.innerHTML = 'BGround: ' +
8901     + SightGlass_1_Window.style.backgroundPosition;
8902
8903 var wheels = 0;
8904 function SG_Wheel(e){
8905     wheels += 1;
8906     SightGlass_1_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
8907     if( e.target.id == "SightGlass_1_Window" ){
8908         e.preventDefault();
8909     }
8910 }
8911 function SG_Adjust1(){
8912     SG_Adjust();
8913     //sampling = 0;
8914     sampling = 20;
8915     //sampling = 200;
8916     window.setTimeout(SG_Adjust1,sampling);
8917 }
8918 function SightGlass_Setup(){
8919     SG_Adjust1();
8920     window.addEventListener('resize',SG_Adjust);
8921     window.addEventListener('mousemove',SG_Adjust);
8922     window.addEventListener('scroll',SG_Adjust);
8923     window.addEventListener('wheel',SG_Wheel,{passive:false});
8924     //window.moveTo(0,0);
8925
8926     // if focused
8927     //window.setInterval(SG_Adjust,1);
8928     window.setTimeout(SG_Adjust,1);
8929 }
8930 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
8931 function Electron_Setup(){
8932     const { BrowserWindow } = require('electron');
8933     const currentWindow = BrowserWindow.getFocusedWindow();
8934     //currentWindow.on('move',function(){ SG_Adjust(); });
8935     currentWindow.addEventListener('move',SG_Adjust);
8936 }
8937 </script>
8938
8939 <input id="SightGlass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8940 <input id="SightGlass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8941 <input id="SightGlass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8942 <span id="SightGlass_WorkCodeSpan">
8943 <script id="SightGlass_WorkScript">
8944 function SightGlass_showWorkCodeView(){
8945     function SightGlass_showWorkCodeView(){
8946         showHtmlCode(SightGlass_WorkCodeView,SightGlass_WorkCodeSpan);
8947     }
8948     SightGlass_WorkCodeViewOpen.addEventListener('click',SightGlass_showWorkCode);
8949 }
8950 SightGlass_openWorkCodeView(); // should be invoked by an event
8951 </script>
8952 </details>
8953 <!-- SightGlass_WorkCodeSpan -->
8954 </-->
8955 //<!-- ===== Work j ===== -->
8956
8957
8958 /*
8959 <!-- ----- GJConsole BEGIN ( ----- -->
8960 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
8961 <details><summary>GJ Console</summary>
8962 <p>
8963 <span id="GJC_RootNode0"></span>
8964 <span id="GJC.Container"></span>
8965 </p>
8966 <style id="GJConsoleStyle">
8967 .GJConsole {
8968     z-index:1000;
8969     width:1000px;
8970     height:200px;
8971     margin:2px;
8972     color:#ffff; background-color:#66a;
8973     font-size:12px; font-family:monospace,Courier New;
8974 }
8975 </style>
8976 <script id="GJConsoleScript" class="GJConsole">
8977 var PS1 = "% "
8978 function GJC_KeyDown(keyevent){
8979     key = keyevent.code
8980     if( key == "Enter" ){
8981         this.Command(this);
8982         this.value += "\n" + PS1 // prompt
8983     }else{
8984         if( key == "Escape" ){
8985             SuppressJSShell = false
8986             GshMenu.focus() // should be previous focus
8987         }
8988     }
8989 var GJC_SessionId
8990 function GJC_SetSessionId(){
8991     var xd = new Date()
8992     GJC_SessionId = xd.getTime() / 1000
8993 }
8994 GJC_SetSessionId()
8995 function GJC_Memory(mem,args,text){
8996     argv = args.split(' ')
8997     cmd = argv[0]
8998     argv.shift()
8999     args = argv.join(' ')
9000     ret = ""
9001
9002     if( cmd == 'clear' ){
9003         Permanent.setItem(mem,'')
9004     }else{
9005         if( cmd == 'read' ){
9006             ret = Permanent.getItem(mem)
9007         }else{
9008             if( cmd == 'save' ){
9009                 val = Permanent.getItem(mem)
9010                 if( val == null ){ val = "" }
9011                 d = new Date()
9012                 val += d.getTime()/1000+ " " +GJC_SessionId+ " "+document.URL+ " "+args+"\n"
9013                 val += text.value
9014                 Permanent.setItem(mem,val)
9015             }else{
9016                 if( cmd == 'write' ){
9017                     val = Permanent.getItem(mem)
9018                     if( val == null ){ val = "" }
9019                     d = new Date()
9020                     val += d.getTime()/1000+ " " +GJC_SessionId+ " "+document.URL+ " "+args+"\n"
9021                     Permanent.setItem(mem,val)
9022             }else{
9023                 ret = "Commands: write | read | save | clear"
9024             }
9025         }
9026     }
9027     return ret

```

```

9027 }
9028 // -- 2020-09-14 added TableEditor
9029 var GJE_CurElement = null; //GJE_RootNode
9030 GJE_NodeSaved = null
9031 GJE_TableNo = 1
9032 function GJE_StyleKeyCommand(key){
9033   keycode = key.code
9034   console.log(GJE_Key: '+keycode)
9035   if( keycode == 'Escape' ){
9036     GJE_SetStyle(this);
9037   }
9038   key.stopPropagation()
9039   // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9040 }
9041 var GJE_CommandMode = false
9042 function GJE_TableKeyCommand(key,tab){
9043   wasCmdMode = GJE_CommandMode
9044   key = key.code
9045   if( key == 'Space' ){
9046     console.log('To command mode: "'+tab.nodeName+"'#"+tab.id)
9047     //tab.setAttribute('contenteditable','false')
9048     tab.style.caretColor = "blue"
9049     GJE_CommandMode = true
9050   }else{
9051     if( key == "KeyA" ){
9052       tab.style.caretColor = "red"
9053       GJE_CommandMode = false
9054     }else{
9055       if( key == "KeyI" ){
9056         tab.style.caretColor = "red"
9057         GJE_CommandMode = false
9058       }else{
9059         if( key == "KeyO" ){
9060           tab.style.caretColor = "red"
9061           GJE_CommandMode = false
9062         }else{
9063           if( key == "KeyJ" ){
9064             console.log("ROW-DOWN")
9065           }else{
9066             if( key == "KeyK" ){
9067               console.log("ROW-UP")
9068             }else{
9069               if( key == "KeyW" ){
9070                 console.log("COL-FORW")
9071               }else{
9072                 if( key == "KeyB" ){
9073                   console.log("COL-BACK")
9074                 }
9075               }
9076             }
9077           }
9078         }
9079       }
9080     }
9081   }
9082   x = ev.clientX
9083   y = ev.clientY
9084   console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x+' y='+y)
9085   // https://www.w3.org/TR/uievents/#events-mouseevents
9086   // https://www.w3.org/TR/uievents/#events-dragdrop
9087   function GJE_DropEvent(ev,elem){
9088     x = ev.clientX
9089     y = ev.clientY
9090     this.style.x = x
9091     this.style.y = y
9092     this.style.position = 'absolute' // 'fixed'
9093     this.parentNode = gsh // just for test
9094     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
9095     + " parent="+this.parentNode.id)
9096   }
9097   function GJE_SetTableStyle(ev){
9098     this.innerHTML = this.value; // sync. for external representation?
9099     if(false){
9100       stdid = this.parentNode.id+this.id
9101       stdid += ' and remove "span" at the end
9102       e = document.getElementById(stdid)
9103       //alert('SetTableStyle #' + e.id + '\n' + this.value)
9104       if( e != null ){
9105         e.innerHTML = this.value
9106       }else{
9107         console.log('Style Not found: ' + stdid)
9108       }
9109     }
9110   }
9111 }
9112 function setSSOfClass(cclass,cstyle){
9113   const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
9114   rlen = ss.cssRules.length;
9115   let tabrule = null;
9116   rulex = -1
9117
9118   // should skip white space at the top of cstyle
9119   sel = cstyle.charAt(0);
9120   selector = sel+cclass;
9121   console.log('-- search style rule for '+selector)
9122
9123   for(let i = 0; i < rlen; i++){
9124     if(ss.cssRules[i].sourceType == 'local'){
9125       console.log('CSS rule ['+i+']/'+rlen+') '+r.selectorText);
9126       if(r.selectorText === selector){ // css class selector
9127         tabrule = ss.cssRules[i];
9128         console.log('CSS rule found for:['+i+']/'+rlen+') '+selector);
9129         ss.deleteRule(i);
9130         //len = ss.cssRules.length;
9131         rulex = i
9132         rulex = i
9133         // should search and replace the property here
9134     }
9135   }
9136   // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
9137   if(tabrule == null){
9138     console.log('CSS rule NOT found for:['+rlen+') '+selector);
9139     ss.insertRule(cstyle,rlen);
9140     ss.insertRule(cstyle,0); // override by 0?
9141     console.log('CSS rule inserted:[ '+(rlen+1)+']\n'+cstyle);
9142   }else{
9143     ss.insertRule(cstyle,rlen);
9144     ss.insertRule(cstyle,0);
9145     console.log('CSS rule replaced:[ '+(rlen+1)+']\n'+cstyle);
9146   }
9147 }
9148 function GJE_SetStyle(te){
9149   console.log('Apply the style to:' + te.id + '\n');
9150   console.log('Apply the style to:' + te.parentNode.id + '\n');
9151   console.log('Apply the style to:' + te.parentNode.className + '\n');
9152   cclass = te.parentNode.className;
9153   setCSSofClass(cclass,te.value); // should get selector part from
9154   // selector { rules }
9155
9156   if(false){
9157     //console.log('Apply the style:')
9158     //stdid = this.parentNode.id+this.id+
9159     //stdid = this.id+'.style'
9160     css = te.value
9161     stdid = te.parentNode.id+'.style'
9162     e = document.getElementById(stdid)
9163     if( e != null ){
9164       //console.log('Apply the style: '+e.id+'\n'+te.value);
9165       //console.log('Apply the style: '+e.id+'\n'+css);
9166       e.innerHTML = css; //te.value;
9167       //ncss = e.sheet;
9168       //ncss.insertRule(te.value,ncss.cssRules.length);
9169     }else{
9170       console.log('No element to Apply the style: '+stdid)
9171     }
9172     tblid = te.parentNode.id+'.table';
9173     e = document.getElementById(tblid);
9174     if( e != null ){
9175       //e.setAttribute('style',css);
9176       e.setProperty('style',css,'!important');
9177     }
9178   }
9179 }
9180 function makeTable(argv){
9181   //tid = GJE_CurElement
9182   //cwe = GJE_Container;
9183   //cwe = GJFactory;
9184   tid = 'table_'+ GJE_TableNo
9185
9186   nt = new Text('\n')
9187   cwe.appendChild(nt)
9188
9189   ne = document.createElement('span'); // the container
9190   ne.appendChild(ne)
9191   ne.id = tid+'-span'
9192   ne.setAttribute('contenteditable',true)
9193
9194   hspan = document.createElement('span'); // html part
9195   //hspan.id = tid + '-html'
9196   //ne.innerHTML = '\n'
9197   nt = new Text('\n')
9198   ne.appendChild(nt)
9199   ne.appendChild(hspan)
9200   ne.appendChild(hspan)
9201
9202   hspan.id = tid
9203   hspan.setAttribute('class',tid)

```

```

9204 ne.setAttribute('draggable','true')
9205 ne.addEventListener('drag',GJE_DragEvent);
9206 ne.addEventListener('dragend',GJE_DropEvent);
9207
9208 var col = 3
9210 var row = 2
9211 if( argv[0] != null ){
9212     col = argv[0]
9213     argv.shift()
9214 }
9215 if( argv[0] == null ){
9216     row = argv[0]
9217     argv.shift()
9218 }
9219
9220 //ne.setAttribute('class','')
9221 ht = "\n"
9222 ht += '<'+table+' id="'+tid+'"' + ' class="'+tid+'"'.
9223 ht += '<'+thead+''
9224 ht += onkeydown="GJE_TableKeyCommand(event,this)"
9225 ht += " ondrag="GJE_DragEvent(event,this)"+'\n'
9226 ht += " ondragend="GJE_DropEvent(event,this)"+'\n'
9227 ht += " draggable="true"+'\n'
9228 ht += " contenteditable="true"+'\n'
9229 ht += ">\n"
9230 ht += "<tbody>\n";
9231 for(r = 0; r < row; r++ ){
9232     ht += "<tr>\n";
9233     for(c = 0; c < col; c++ ){
9234         ht += "<td>";
9235         ht += "ABCD EFGHIJKLMNOPQRSTUVWXYZ."charAt(c) + r
9236         ht += "</td>\n"
9237     }
9238     ht += "</tr>\n"
9239 }
9240 ht += '<'+tbody+'>';
9241 ht += '<'+table+'>';
9242 hspan.innerHTML = ht;
9243 nt = new Text('\n')
9244 ne.appendChild(nt)
9245
9246 st = '#'+tid+' /*\n' // for instance specific
9247 st += 'background-color:#aaa;\n'
9248 st += 'background-color:#eef;\n'
9249 st += 'color:#222;\n'
9250 st += 'font-size:#14pt !important;\n'
9251 st += 'font-family:monospace,Courier New !important;\n'
9252 st += '/* hit ESC to apply */\n'
9253
9254 // wish script to be included
9255 [n] = document.createElement('script')
9256 ne.appendChild(n)
9257 ne.innerHTML = 'function SetStyle(e){'
9258
9259 // selector seems lost in dynamic style appending
9260 if(falso){
9261     ns = document.createElement('style')
9262     ns.appendChild(ns)
9263     ns.id = tid + '.style'
9264     ns.innerHTML = '\n'+st
9265     ns = new Text('\n')
9266     ne.appendChild(ns)
9267 }
9268 setCSSOfClass(tid,st); // should be in JavaScript script?
9269
9270 nx = document.createElement('textarea')
9271 ne.appendChild(nx)
9272 nx.id = tid + '_style_def'
9273 nx.setAttribute('class','GJ_StyleEditor')
9274 nx.spellcheck = false
9275 nx.cols = 60
9276 nx.rows = 10
9277 nx.innerHTML = '\n'+st
9278 nx.addEventListener('change',GJE_SetTableStyle);
9279 nx.addEventListener('keydown',GJE_StyleKeyCommand);
//nx.addEventListener('click',GJE_SetTableStyle);
9280
9281 nt = new Text('\n')
9282 cwe.appendChild(nt)
9283
9284 GJE_TableNo += 1
9285 return 'created TABLE id="'+tid+'"'.
9286
9287 function GJE_NodeEdit(argv){
9288     cwe = GJE_CurElement
9289     cmd = argv[0]
9290     argv.shift()
9291     args = argv.join(' ')
9292     ret = ""
9293
9294     if( cmd == '-u' || cmd == '-un' || cmd == 'undo' ){
9295         if( GJE_NodeSaved != null ){
9296             xn = GJE_RootNode
9297             GJE_RootNode = GJE_NodeSaved
9298             GJE_NodeSaved = xn
9299             ret = "-- did undo"
9300         }else{
9301             ret = "-- could not undo"
9302         }
9303     }
9304     return ret
9305
9306     GJE_NodeSaved = GJE_RootNode.cloneNode()
9307     if( cmd == '-c' || cmd == '-cd' || cmd == 'cd' ){
9308         if( argv[0] == null ){
9309             ne = GJE_RootNode
9310         }else{
9311             if( argv[0] == '..' ){
9312                 ne = cwe.parentNode
9313             }else{
9314                 ne = document.getElementById(argv[0])
9315             }
9316             if( ne != null ){
9317                 GJE_CurElement = ne
9318                 ret = "-- current node: " + ne.id
9319             }else{
9320                 ret = "-- not found: " + argv[0]
9321             }
9322         }
9323     }else if( cmd == 'mkt' || cmd == '.mktable' ){
9324         makeTable(argv)
9325     }
9326     if( cmd == '-m' || cmd == '.mk' || cmd == 'mk' ){
9327         ne = document.createElement(argv[0])
9328         ne.id = argv[0]
9329         ret = "- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9330         cwe.appendChild(ne)
9331         if( cmd == '-m' || cmd == '.mk' ){
9332             GJE_CurElement = ne
9333         }
9334     }else if( cmd == '-n' || cmd == '.nm' || cmd == 'nm' ){
9335         cwe.setId(argv[0])
9336     }
9337     if( cmd == '-r' || cmd == '.rm' || cmd == 'rm' ){
9338     }else if( cmd == '-h' || cmd == '.sh' || cmd == 'sh' ){
9339         s = argv.join(' ')
9340         cwe.innerHTML = s
9341     }else if( cmd == '-a' || cmd == '.sa' || cmd == 'sa' ){
9342         cwe.setAttribute(argv[0],argv[1])
9343     }else if( cmd == '-l' ){
9344         if( cmd == 'i' || cmd == '.ih' || cmd == 'ih' ){
9345             ret = cwe.innerHTML
9346         }else if( cmd == 'p' || cmd == '.pw' || cmd == 'pw' ){
9347             ret = cwe.nodeType + "+" + cwe.tagName + " " + cwe.id
9348             for(wt in cwe.parentNode){ we = wt; if( we != null ){
9349                 ret += "+" + we.nodeType + " " + we.tagName + " " + we.id
9350                 we = we.parentNode
9351             }
9352         }
9353     }else if( cmd == 'Command' || cmd == 'rm \n' ){
9354         line = argv[0].split('\n')
9355         line = line[line.length-1]
9356         argv = line.split(' ')
9357         text.value += '\n'
9358         if( argv[0] == '$' ){ argv.shift()
9359         }args0 = argv.join(' ')
9360         cmd = argv[0]
9361         argv.shift()
9362         args = argv.join(' ')
9363     }
9364 }
9365
9366 //alert(ret);
9367 return ret
9368
9369 }
9370
9371 function GJE_Command(text){
9372     lines = text.value.split('\n')
9373     line = lines[lines.length-1]
9374     argv = line.split(' ')
9375     text.value += '\n'
9376     if( argv[0] == '$' ){ argv.shift()
9377     }args0 = argv.join(' ')
9378     cmd = argv[0]
9379     argv.shift()
9380     args = argv.join(' ')
9381 }

```

```

9381     if( cmd == 'nolog' ){
9382         StopConsoleLog = true
9383     }else{
9384         if( cmd == 'new' ){
9385             if( argv[0] == 'table' ){
9386                 argv.shift()
9387                 console.log('args=' + argv)
9388                 text.value += makeTable(argv)
9389             }else if( argv[0] == 'console' ){
9390                 text.value += GJ_NewConsole('GJ_Console')
9391             }else{
9392                 text.value += '-- new { console | table }'
9393             }
9394         }else if( cmd == 'strip' ){
9395             //text.value += GJF_StripClass()
9396         }else if( cmd == 'ass' ){
9397             if( argv[0] == '#table_1' ){
9398                 if(argv[0]==''0')
9399                     rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
9400                 else
9401                     rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}';
9402                 document.styleSheets[3].deleteRule(0);
9403                 document.styleSheets[3].insertRule(rule1,0);
9404                 text.value += 'CSS rule added: '+rule1
9405             }else if( cmd == 'print' ){
9406                 e = null;
9407                 if( e == null ){
9408                     e = document.getElementById('GJFactory_0')
9409                 }if( e == null ){
9410                     e = document.getElementById('GJFactory_1')
9411                 }if( argv[0] != null ){
9412                     id = argv[0];
9413                     if( id == 'f' ){
9414                         /e = document.getElementById('GJE_RootNode');
9415                     }else{
9416                         e = document.getElementById(id)
9417                     }
9418                     if( e != null ){
9419                         text.value += e.outerHTML
9420                     }else{
9421                         text.value += "Not found: " + id
9422                     }
9423                 }else{
9424                     text.value += GJE_RootNode.outerHTML
9425                 }
9426             }else if( cmd == 'destroy' ){
9427                 text.value += GJFactory_Destroy()
9428             }else if( cmd == 'save' ){
9429                 e = document.getElementById('GJFactory')
9430                 Permanent.setItem('GJFactory-1',e.innerHTML)
9431                 text.value += '-- Saved GJFactory'
9432             }else if( cmd == 'load' ){
9433                 gif = Permanent.getItem('GJFactory-1')
9434                 e = document.getElementById('GJFactory')
9435                 e.innerHTML = gif
9436                 // must restore EventListener
9437                 text.value += '-- EventListener was not restored'
9438             }else if( cmd.charAt(0) == '.' ){
9439                 argv0 = argv0.split('.')
9440                 text.value += GJE_NodeEdit(argv0)
9441             }else if( cmd == 'cont' ){
9442                 bannerIsStopping = false
9443                 GshMenuStop.innerHTML = "Stop"
9444             }else if( cmd == 'date' ){
9445                 text.value += DateLong()
9446             }else if( cmd == 'echo' ){
9447                 text.value += args
9448             }else if( cmd == 'fork' ){
9449                 html_fork()
9450             }else if( cmd == 'last' ){
9451                 text.value += MyHistory
9452                 //document.createElement("span")
9453                 //h.innerHTML = MyHistory
9454                 //text.value += h.innerHTML
9455                 //tx = MyHistory.replace("\n","");
9456                 //text.value += tx.replace("<br>","\n") + "xxxx<"+>yyyy"
9457             }else if( cmd == 'ne' ){
9458                 text.value += GJE_NodeEdit(argv)
9459             }else if( cmd == 'reload' ){
9460                 location.reload()
9461             }else if( cmd == 'mem' ){
9462                 text.value += GJC_Memory('GJC_Storage',args,text)
9463             }else if( cmd == 'stop' ){
9464                 bannerIsStopping = true
9465                 GshMenuStop.innerHTML = "Start"
9466             }else if( cmd == 'who' ){
9467                 text.value += "SessionId=" + GJC_SessionId + " " + document.URL
9468             }else if( cmd == 'wall' ){
9469                 text.value += GJC_Memory('GJC_Wall','write',text)
9470             }else{
9471                 text.value += "Commands: help | echo | date | last \n"
9472                 + '          new | save | load | mem \n'
9473                 + '          who | wall | fork | nife'
9474             }
9475         }
9476         function GJC_Input(){
9477             if( this.value.endsWith("\n") ){ // remove NL added by textarea
9478                 this.value = this.value.slice(0,this.value.length-1)
9479             }
9480         }
9481         var GJC_Id = null
9482         function GJC_Resize(){
9483             GJC_Id.style.left.zindex = 20000
9484             //GJC_Id.style.width = window.innerWidth - 16
9485             GJC_Id.style.width = '100%';
9486             GJC_Id.style.height = 300;
9487             GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9488             GJC_Id.style.color = "rgb(255,255,255,1.0)"
9489         }
9490         function GJC_FocusIn(){
9491             this.spellcheck = false
9492             SuppressJSShell = true
9493             this.onkeydown = GJC_KeyDown
9494             GJC_Resize()
9495         }
9496         function GJC_FocusOut(){
9497             SuppressJSShell = false
9498             this.removeEventlistener('keydown',GJC_KeyDown);
9499         }
9500         window.addEventListener('resize',GJC_Resize);
9501         function GJC_OnStorage(e{
9502             //alert('Got Message')
9503             //GJC.value += "\n((ReceivedMessage))\n"
9504         }
9505         window.addEventListener('storage',GJC_OnStorage);
9506         //window.addEventListener('storage',(),()=>{alert('GotMessage')})
9507         function GJC_Setup(gjcid){
9508             /gjcid.style.width = gsh.getBoundingClientRect().width
9509             gjid.style.width = '100%';
9510             gjcid.value = "GJShell Console // " + GehVersion.innerHTML + "\n"
9511             /gjid.value += "Date: " + DateLong() + "\n"
9512             gjid.value += PS1
9513             gjid.onfocus = GJC_FocusIn
9514             gjid.addEventListener('input',GJC_Input);
9515             gjid.addEventListener('focusout',GJC_FocusOut);
9516             GJC_Id = gjid
9517         }
9518         function GJC_Clear(id){
9519         }
9520         function GJConsole_initConsole(){
9521             if( document.getElementById('GJC_0') != null ){
9522                 GJC_Setup(GJC_0)
9523             }else{
9524                 GJC_0.innerHTML = '<'+textarea id="GJC_1" class="GJConsole"><'+/textarea>';
9525                 GJC_Setup(GJC_1)
9526                 factory = document.createElement('span');
9527                 gsh.appendChild(factory)
9528                 GJE_RootNode = factory;
9529                 GJE_CurElement = GJE_RootNode;
9530             }
9531         }

```

```

9558     }
9559   } var initGJCF = false;
9560   function GJConsole_initFactory(){
9561     if( initGJCF ) { Return; } initGJCF = true;
9562     GShell_initKeyCommands();
9563     GJConsole_initConsole();
9564   } //GJConsole_initFactory();
9565   // TODO: focus handling
9566
9567 </script>
9568 <style>
9569   GJ_StyleEditor {
9570     font-size:9pt !important;
9571     font-family:Courier New, monospace !important;
9572   }
9573 </style>
9574 </details>
9575 </span>
9576 <!-- ----- GJConsole END } ----- -->
9577 */
9578 /*span id="BlinderText">
9579 <style id="BlinderTextStyle">
9580 #GJlinkView {
9581   xposition:absolute; z-index:5000;
9582   position:relative;
9583   display:block;
9584   border:1px solid black;
9585   color:#fff;
9586   width:800px; height:300px; resize:both;
9587   margin:0px; padding:4px;
9588   background-color:rgba(200,200,200,0.5) !important;
9589 }
9590 .MsgText {
9591   width:578px !important;
9592   resize:both !important;
9593   color:#000 !important;
9594 }
9595 .GjNote {
9596   font-family:Georgia !important;
9597   font-size:13pt !important;
9598   color:#2a2 !important;
9599 }
9600 .textfield {
9601   display:inline;
9602   border:0.5px solid #444;
9603   border-radius:3px;
9604   color:#000; background-color:#fff;
9605   width:106pt; height:18pt;
9606   margin:2px;
9607   padding:2px;
9608   resize:none;
9609   vertical-align:middle;
9610   font-size:10pt; font-family:Courier New;
9611 }
9612 .textlabel {
9613   border:0px solid #000 !important;
9614   background-color:rgba(0,0,0,0);
9615 }
9616 .textURL {
9617   width:300pt !important;
9618   border:0px solid #000 !important;
9619   background-color:rgba(0,0,0,0);
9620 }
9621 .VisibleText {
9622 }
9623 .BlinderText {
9624   color:#000; background-color:#eee;
9625 }
9626 .joinButton {
9627   font-family:Georgia !important;
9628   font-size:1pt;
9629   line-height:1.1;
9630   height:18pt;
9631   width:50pt;
9632   padding:3px !important;
9633   text-align:center !important;
9634   border-color:#aaa !important;
9635   border-radius:5px;
9636   color:#fff; background-color:#4a4 !important;
9637   vertical-align:middle !important;
9638 }
9639 .SendButton {
9640   vertical-align:top;
9641 }
9642 .wsLog {
9643   font-size:10pt;
9644   color:#000 !important;
9645   line-height:1.0;
9646   background-color:rgba(255,255,255,0.7) !important;
9647   font-family:Courier New,monospace !important;
9648   width:99.3%;
9649   white-space:pre;
9650 }
9651 </style>
9652 <!-- Form autofill test
9653 Location<input id="xxserv" type="text" value="https://192.168.10.1/boafrm/formLogin" size="80">
9654 <form method="POST" id="xxform" action="https://192.168.10.1/boafrm/formLogin">
9655   <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
9656 <details><summary>Form Auto. Filling</summary>
9657 <style>
9658   .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9659   , display:inline !important; font-size:10pt !important; padding:1px !important;
9660   }
9661 </style>
9662 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive=""
9663 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
9664 Location<input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
9665 Username:<input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
9666 Password:<input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
9667 Sessionid:<input id="xxsid" class="xxinput" name="SESSION_ID" type="text" size="80">
9668 <input id="xxsub" class="xxinput" type="submit" value="Submit"></form>
9669 </span>
9670 <script>
9671   function XXSetFormAction(){
9672     xxform.setAttribute('action',xxserv.value);
9673   }
9674   xxform.setAttribute('action',xxserv.value);
9675   xxserv.addEventListener('change',XXSetFormAction);
9676   //xxserv.value = location.href;
9677 </script>
9678 </details>
9679 </
9680 */
9681 <details id="BlinderTextClass"><summary>BlinderText</summary>
9682 <span class="gsh-src">
9683 <span id="BlinderTextScript">
9684   // https://w3c.github.io/uievents/#event-type-keydown
9685   // 2020-09-21 class BlinderText - textarea element not to be readable
9686   // BlinderText attributes
9687   // bl_plainText - all
9688   // bl_hideChecksum - [false]
9689   // bl_showLength - [false]
9690   // bl_visible - [false]
9691   // data-bl_config - []
9692   // - min. length
9693   // - max. length
9694   // - acceptable charset in generate text
9695   function BlinderChecksum(text){
9696     plain = text.bl_plainText;
9697     return strCRC32(plain,plain.length).toFixed(0);
9698   }
9699   function Blinderkeydown(ev){
9700     pass = ev.target
9701     if( ev.code == 'Enter' ){
9702       ev.preventDefault();
9703     }
9704     ev.stopPropagation()
9705   }
9706   function Blinderkeyup(ev){
9707     blind = ev.target
9708     if( ev.code == 'Backspace'){
9709       blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
9710     }else
9711     if( and(ev.code == 'KeyV', ev.ctrlKey) ){
9712       blind.bl_visible = iblind.bl_visible;
9713     }else
9714     if( and(ev.code == 'KeyL', ev.ctrlKey) ){
9715       blind.bl_showLength = !blind.bl_showLength;
9716     }else
9717     if( and(ev.code == 'KeyU', ev.ctrlKey ) ){
9718       blind.bl_plainText = '';
9719     }else
9720     if( and(ev.code == 'KeyR', ev.ctrlKey) ){
9721       checksum = BlinderChecksum(blind);
9722       blind.bl_plainText = checksum; // .toString(32);
9723     }else
9724     if( ev.code == 'Enter' ){

```

```

9735     ev.stopPropagation();
9736     ev.preventDefault();
9737     return;
9738   }else{
9739     if( ev.key.length != 1 ){
9740       console.log('KeyUp: '+ev.code+'/'+ev.key);
9741       return;
9742     }else{
9743       blind.bl_plainText += ev.key;
9744     }
9745   }
9746   leng = blind.bl_plainText.length;
9747   //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
9748   checksum = BlinderChecksum(blind) % 10; // show last one digit only
9749   visual = '';
9750   if( !blind.bl_hideChecksum || blind.bl_showLength ){
9751     visual += '[';
9752   }
9753   if( !blind.bl_hideChecksum ){
9754     visual += '#'+checksum.toString(10);
9755   }
9756   if( blind.bl_showLength ){
9757     visual += '/' + leng;
9758   }
9759   if( !blind.bl_hideChecksum || blind.bl_showLength ){
9760     visual += ']';
9761   }
9762   if( blind.bl_visible ){
9763     visual += blind.bl_plainText;
9764   }else{
9765     visual += '*'..repeat(leng);
9766   }
9767   blind.value = visual;
9768 }
9769 function BlinderKeyup(ev){
9770   BlinderKeyup(ev);
9771   ev.stopPropagation();
9772 }
9773 // https://w3c.github.io/uievents/#keyboardevent
9774 // https://w3c.github.io/uievents/#uievent
9775 // https://www.whatwg.org/#event
9776 function BlinderTextEvent(){
9777   ev = event;
9778   blind = ev.target;
9779   console.log('Event: '+ev.type+'#'+blind.nodeName+'#'+blind.id)
9780   if( ev.type == 'keyup' ){
9781     BlinderKeyup(ev);
9782   }else{
9783     if( ev.type == 'keydown' ){
9784       BlinderKeydown(ev);
9785     }else{
9786       console.log('thru-event '+ev.type+'#'+blind.nodeName+'#'+blind.id)
9787     }
9788   }
9789   //< textarea hidden id="BlinderTextClassDef" class="textField">
9790   // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9791   // spellcheck="false"></textarea>
9792   //< textarea hidden id="gjxText" class="BlinderText">
9793   //< input type="text" id="BlinderText" placeholder="PassWord" />
9794   // onkeydown="BlinderTextEvent()"
9795   // onkeyup="BlinderTextEvent()"
9796   // spellcheck="false"></textarea>
9797   function SetupBlinderText(parent,txa,phold){
9798     if( txa == null ){
9799       txa = document.createElement('textarea');
9800       //txa.id = id;
9801       txa.setAttribute('class','txaField BlinderText');
9802       txa.setAttribute('placeholder','phold');
9803       txa.setAttribute('onkeydown','BlinderTextEvent()');
9804       txa.setAttribute('onkeyup','BlinderTextEvent()');
9805       txa.setAttribute('spellcheck','false');
9806       //txa.setAttribute('bl_plainText','false');
9807       txa.bl_plainText = '';
9808       //parent.appendChild(txa);
9809     }
9810     function DestroyBlinderText(txa){
9811       txa.removeAttribute('class');
9812       txa.removeAttribute('placeholder');
9813       txa.removeAttribute('onkeydown');
9814       txa.removeAttribute('onkeyup');
9815       txa.removeAttribute('spellcheck');
9816       txa.bl_plainText = '';
9817     }
9818   }
9819   // visible textarea like Username
9820   //< input type="text" id="VisibleTextEvent" class="VisibleText" />
9821   function VisibleTextEvent(){
9822     if( event.code == 'Enter' ){
9823       if( event.target.NoEnter ){
9824         event.preventDefault();
9825       }
9826     }
9827     event.stopPropagation();
9828   }
9829   function SetupVisibleText(parent,txa,phold){
9830     if( false ){
9831       txa.setAttribute('class','textField VisibleText');
9832     }else{
9833       newclass = txa.getAttribute('class');
9834       if( and(newclass != null, newclass != '' ) ){
9835         newclass += ' ';
9836       }
9837       newclass += 'VisibleText';
9838       txa.setAttribute('class',newclass);
9839     }
9840     //console.log('SetupVisibleText#'+txa.id);
9841     txa.setAttribute('cols','');
9842     txa.setAttribute('rows','');
9843     txa.setAttribute('placeholder','');
9844     txa.setAttribute('onkeydown','VisibleTextEvent()');
9845     txa.setAttribute('onkeyup','VisibleTextEvent()');
9846     txa.setAttribute('spellcheck','false');
9847     cols = txa.getAttribute('cols');
9848     if( cols != null ){
9849       txa.style.width = '580px';
9850       //console.log('VisibleText#'+txa.id+' cols='+cols)
9851     }else{
9852       //console.log('VisibleText#'+txa.id+' NO cols')
9853     }
9854     rows = txa.getAttribute('rows');
9855     if( rows != null ){
9856       txa.style.height = '30px';
9857       txa.style.resize = 'both';
9858     }
9859     txa.NoEnter = false;
9860   }else{
9861     txa.NoEnter = true;
9862   }
9863 }
9864 function DestroyVisibleText(txa){
9865   txa.removeAttribute('class');
9866   txa.removeAttribute('placeholder');
9867   txa.removeAttribute('onkeydown');
9868   txa.removeAttribute('onkeyup');
9869   txa.removeAttribute('spellcheck');
9870   cols = txa.removeAttribute('cols');
9871 }
9872 </span>
9873 <script>
9874 js = document.getElementById('BlinderTextScript');
9875 eval(js.innerHTML);
9876 //js.outerHTML = '';
9877 </script>
9878 <span><!-- end of class="gsh-src" -->
9879 </details>
9880 </span>
9881 </>
9882 </>
9883 <script id="GJLinkScript">
9884 function gjkey.hash(text){
9885   return strCRC32(text,text.length) % 0x10000;
9886 }
9887 function gj_addlog_e(msg){
9888   now = new Date().getTime() / 1000.toFixed(3);
9889   tstamp = '['+now+']';
9890   e.value += tstamp + msg;
9891   e.scrollTop = e.scrollHeight;
9892 }
9893 function gj_addlog_c1(msg){
9894   ws0_log.value += '(console.log) ' + msg + '\n';
9895 }
9896 var GJ_Channel = null;
9897 var GJ_Log = null;
9898 var gjx; // the global variable
9899 function gj_join(){
9900   target = gj_join;
9901   if( target.value == 'Leave' ){
9902     GJ_Channel.close();
9903     GJ_Channel = null;
9904     target.value = 'Join';
9905   }
9906   return;
9907 }
9908 var ws0;
9909 var ws0_log;
9910

```

```

9912     sav_console_log = console.error
9913     console.error = gj_addlog;
9914     ws0 = new WebSocket(gj_serv.innerHTML);
9915     console.error = sav_console_log
9916
9917     GJ_Channel1 = ws0;
9918     ws0_log = document.getElementById('ws0_log');
9919     GJ_Log = ws0_log;
9920
9921     now = (new Date().getTime() / 1000).toFixed(3);
9922     const wstats = ['CONNECTING','OPEN','CLOSING','CLOSED'];
9923     cst = wstats[ws0.readyState];
9924     gj_addlog(ws0_log,'stat '+cst+' : GJ Linked\n');
9925
9926     ws0.addEventListener('error', function(event){
9927       gj_addlog(ws0_log,'stat error : transport error?\n');
9928     });
9929     ws0.addEventListener('open', function(event){
9930       GJLinkView.style.zIndex = 10000;
9931       //console.log('#'+GJLinkView.id+'.zIndex=' +GJLinkView.style.zIndex);
9932       date1 = new Date().getTime();
9933       date2 = (date1 / 1000).toFixed(3);
9934       seed = date1.toString(16);
9935
9936       // user name and key
9937       user = document.getElementById('gj_user').value;
9938       if(user.length == 0 ){
9939         gj_user.value = 'nemo';
9940         user = 'nemo';
9941       }
9942       key1 = document.getElementById('gj_ukey').bl_plainText;
9943       ukey = gjkey_hash(seed+user+key1).toString(16);
9944
9945       // session name and key
9946       chan = document.getElementById('gj_chan').value;
9947       if(chan.length == 0 ){
9948         gj_chan.value = 'main';
9949         chan = 'main';
9950       }
9951       key2 = date2 +' JOIN ' + user + '|' + chan + ' ' + ukey + ':' + ckey;
9952       ckey = gjkey_hash(seed+chan+key2).toString(16);
9953
9954       msg = date2 +' '+JOIN +' '+ user + '|' + chan + ' ' + ukey + ':' + ckey;
9955       gj_addlog(ws0_log,'send '+msg+'\n');
9956       ws0.send(msg);
9957
9958       target.value = 'Leave';
9959       //console.log('['+date2+'] #' +target.id+ ' '+target.value+'\n');
9960       gj1_addlog(ws0_log,'label '+target.value+'\n');
9961     });
9962     ws0.addEventListener('message', function(event){
9963       now = (new Date().getTime() / 1000).toFixed(3);
9964       msg = event.data;
9965       if( false ){
9966         gj1_addlog(ws0_log,'recv '+msg+'\n');
9967       }
9968       argv = msg.split(' ')
9969       timestamp = argv[0];
9970       argv.shift();
9971       if(argv[0]=='reload'){
9972         location.reload()
9973       }
9974       gjcmd = argv[0];
9975       timestamp = '';
9976       if( gjcmd == 'CAST' ){ // from reflector
9977         timestamp = argv[0];
9978         argv.shift(); // original time stamp
9979         cfrom = argv[0];
9980         argv.shift(); // original from
9981       }
9982       argv.shift(); // command
9983       from = argv[0];
9984       argv.shift(); // from|to
9985       cmd1 = argv[0];
9986       argv.shift(); // xxxx command
9987
9988       if( false ){
9989         gj1_addlog(ws0_log,'--'
9990           +' '+timestamp+timestamp
9991           +' ,gjcmd=' +gjcmd
9992           +' ,from=' +from
9993           +' ,cmd=' +cmd1+'|'
9994           +' ,argv=' +argv);
9995       }
9996       if( cmd1 == 'auth' ){
9997         // doing authorization required
9998       }
9999       if( cmd1 == 'echo' ){
10000         now = (new Date().getTime() / 1000).toFixed(3);
10001         msg = now+' '+RESP +'argv.join()';
10002         gj1_addlog(ws0_log,'send '+msg+'\n');
10003         ws0.send(msg);
10004
10005       }
10006       if( cmd1 == 'eval' ){
10007         argv.shift();
10008         js = argv.join(' ');
10009         ret = eval(js); //----- eval()
10010         gj1_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
10011         now = (new Date().getTime() / 1000).toFixed(3);
10012         msg = now+' '+RESP +' '+ret;
10013         ws0.send(msg);
10014         gj1_addlog(ws0_log,'send '+msg+'\n');
10015
10016       }
10017       if( cmd1 == 'DRAW' ){
10018         if( false ){
10019           gj1_addlog(ws0_log,'DRAW '+argv[0]+'\n')
10020           Pointillism_RemoteDraw(argv[0]);
10021         }
10022     });
10023     ws0.addEventListener('close', function(event){
10024       if( GJ_Channel == null ){
10025         gj1_addlog(ws0_log,'stat OK : GJ UnLinked\n');
10026         return;
10027       }
10028       GJ_Channel.close();
10029       GJ_Channel = null;
10030       target.value = 'Join';
10031       gj1_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
10032     });
10033   }
10034   function GJ_BcastMessageUserPass(user,chan,msgbody){
10035     now = (new Date().getTime() / 1000).toFixed(3);
10036     msg = now+' BCAST '+user+'|'+chan+' '+msgbody;
10037     if( false ){
10038       gj1_addlog(GJ_Log,'send '+msg+'\n');
10039     }
10040     GJ_Channel.send(msg);
10041   }
10042   function GJ_BcastMessage(msgbody){
10043     if( GJ_Channel == null ){
10044       gj1_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10045       return;
10046     }
10047     //target = event.target;
10048     user = document.getElementById('gj_user').value;
10049     chan = document.getElementById('gj_chan').value;
10050     GJ_BcastMessageUserPass(user,chan,msgbody);
10051   }
10052   function GJ_SendMessageUserPass(user,chan,msgbody){
10053     now = (new Date().getTime() / 1000).toFixed(3);
10054     msg = now+' ISAY '+user+'|'+chan+' '+msgbody;
10055     gj1_addlog(GJ_Log,'send '+msg+'\n');
10056     GJ_Channel.send(msg);
10057   }
10058   function GJ_SendMessage(msgbody){
10059     if( GJ_Channel == null ){
10060       gj1_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10061       return;
10062     }
10063     //target = event.target;
10064     user = document.getElementById('gj_user').value;
10065     chan = document.getElementById('gj_chan').value;
10066     GJ_SendMessageUserPass(user,chan,msgbody);
10067   }
10068   function GJ_Send(){
10069     msgbody = gj_sendText.value;
10070     GJ_SendMessage(msgbody);
10071   }
10072   </script>
10073
10074 <!-- ----- GJLINK ----- -->
10075 <!--
10076   - User can subscribe to a channel
10077   - A channel will be broadcasted
10078   - A channel can be a pattern (regular expression)
10079   - User is like From/(me) and channel is like To/ or Recipient:
10080   - like VIBBUS
10081   - watch message with SENDME, WATCH, CATCH, HEAR, or so
10082   - routing with path expression or name pattern (with routing with DNS like system)
10083 -->
10084
10085
10086 //<span id="GJLinkGolang">
10087 //<details id="GshWebSocket"><summary>Golang / JavaScript Link</summary>
10088 // <span class="gsh-src"><!-- { -->
```

```

10089// 2020-0920 created
10090 // <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
10091 // <a href="https://golang.org/x/net/websocket">websocket</a>
10092// INSTALL: go get golang.org/x/net/websocket
10093// INSTALL: sudo apt-get install git (if git is not installed yet)
10094// import "golang.org/x/net/websocket"
10095 const gshws_origin = "http://localhost:9999"
10096 const gshws_server = "localhost:9999"
10097 const gshws_port = 9999
10098 const gshws_path = "gjlink1"
10099 const GSHWS_MSGSIZE = (8*1024)
10100 func fmtstring(fmts string, params ...interface{})(string){
10101     return fmt.Sprintf(fmts,params...)
10102 }
10103 func GSHWS_MARK(what string)(string){
10104     now := time.Now()
10105     us := fmt.Sprintf("%06d",now.Nanosecond() / 1000)
10106     mark := "\n"
10107     if(i >= ConsoleLineTop ){
10108         mark += "\n"
10109         AtConsoleLineTop = true
10110     }
10111     mark += "["+now.Format(time.Stamp)+"."+us+"] -GJ- "+ what + " "
10112     return mark
10113 }
10114 func qchk(what string,err error){
10115     if( err != nil ){
10116         panic(GSHWS_MARK(what)+err.Error())
10117     }
10118 }
10119 func glog(what string, fmts string, params ...interface{}{
10120     fmt.Println(GSHWS_MARK(what))
10121     fmt.Printf(fmts+"\n",params...)
10122 }
10123 )
10124 var WSV = []*websocket.Conn{
10125 }
10126 func jsend(argv []string{
10127     if len(argv) <= 1 {
10128         fmt.Printf("-Ij & [-m] command arguments\n",argv[0])
10129     return
10130 }
10131     argv = argv[1:]
10132     if( len(WSV) == 0 ){
10133         fmt.Printf("-Ej-- No link now\n")
10134     return
10135 }
10136     if( 1 < len(WSV) ){
10137         fmt.Printf("-Ij-- multiple links (%v)\n",len(WSV))
10138     }
10139     multicast := false // should be filtered with regexp
10140     if( 0 < len(argv) && argv[0] == "-m" ){
10141         multicast = true
10142         argv = argv[1:]
10143     }
10144     args := strings.Join(argv, " ")
10145
10146     now := time.Now()
10147     msec := now.UnixNano() / 1000000;
10148     tstamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
10149     msg := fmtstring("%v SEND gshell| %v",tstamp,args)
10150
10151     if( multicast ){
10152         for i,ws := range WSV {
10153             wn,werr := ws.Write([]byte(msg))
10154             if( werr != nil ){
10155                 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10156             }
10157             glog("SQ",fmtstring("(%v) %v",wn,msg))
10158         }
10159     }else{
10160         i := 0
10161         ws := WSV[i]
10162         wn,werr := ws.Write([]byte(msg))
10163         if( werr != nil ){
10164             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10165         }
10166         glog("SQ",fmtstring("(%v) %v",wn,msg))
10167     }
10168 }
10169 func we_broadcast(msg string)(wn int,werr error){
10170     for i,ws := range WSV {
10171         wn,werr := ws.Write([]byte(msg))
10172         if( werr != nil ){
10173             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10174         }
10175         glog("SQ",fmtstring("(%v) %v",wn,msg))
10176     }
10177     return wn,werr;
10178 }
10179 func serv1(ws *websocket.Conn) {
10180     ws := appendWSV(ws)
10181     //fmt.Println("n")
10182     glog("CO","accepted connections(%v)",len(WSV))
10183     //remoteAddr := ws.RemoteAddr
10184     //fmt.Println("-- accepted %v",remoteAddr)
10185     //fmt.Println("-- accepted %v",ws.Config())
10186     //fmt.Println("-- accepted %v",ws.Config().Header)
10187     //fmt.Println("-- accepted %v %%n",ws.serv1)
10188
10189     var reqb = make([]byte,GSHWS_MSGSIZE)
10190     for {
10191         rn, rerr := ws.Read(reqb)
10192         if( rerr != nil || rn < 0 ){
10193             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
10194             break
10195         }
10196         req := string(reqb[0:rn])
10197         glog("SQ",fmtstring("(%v) %v",rn,req))
10198
10199         argv := strings.Split(req, " ")
10200         argv = argv[1:]
10201         if( 0 < len(argv) ){
10202             if( argv[0] == "RESP" ){
10203                 // should forward to the destination
10204                 continue;
10205             }
10206         }
10207         now := time.Now()
10208         msec := now.UnixNano() / 1000000;
10209         tstamp := fmt.Sprintf("%.3f",float64(msec)/1000.0)
10210         res := fmtstring("%v +%CAST%v",tstamp,req)
10211
10212         wn := 0;
10213         werr := error(nil);
10214         if( 0 < len(argv) && argv[0] == "BCAST" ){
10215             wn, werr = ws.broadcast(res);
10216         }else{
10217             wn, werr = ws.Write([]byte(res))
10218         }
10219         gchk("SE",werr)
10220         glog("SR",fmtstring("(%v) %v",wn,string(res)))
10221
10222     }
10223     glog("SF","WS response finish")
10224
10225     WSV := []*websocket.Conn{
10226     ws :
10227     for i,v := range WSV {
10228         if( v != ws ){
10229             wsx = i
10230             wsx = append(wsx,v)
10231         }
10232     }
10233     WSV = wsx
10234     //glog("CO","closed %v",ws)
10235     glog("CO","closed connection (%v/%v)",wsx+1,len(WSV)+1)
10236     ws.Close()
10237 }
10238 // url ::= [scheme://]host[:port]{/path}
10239 func decomp_URL(url string){
10240 }
10241 func full_wsURL(){
10242 }
10243 func gj_server(argv []string) {
10244     gjserv := gshws_url
10245     diport := gshws_server
10246     gjpath := gshws_path
10247     gjschema := "ws"
10248
10249     //cmd := argv[0]
10250     argc = argv[1];
10251     if( 1 < len(argv) ){
10252         serv := argv[0]
10253         if( 0 < strings.Index(serv,"://") ){
10254             schemev := strings.Split(serv,"://")
10255             gjschema = schemev[0]
10256             serv = schemev[1]
10257         }
10258         if( 0 < strings.Index(serv,"/") ){
10259             pathv := strings.Split(serv,"/")
10260             serv = pathv[0]
10261             gjpath = pathv[1]
10262         }
10263         servv := strings.Split(serv,":")
10264         host := "localhost"
10265         port := 9999

```

```

10266 if( servv[0] != ""){
10267     host = servv[0]
10268 }
10269 if( len(servv) == 2 ){
10270     fmt.Sscanf(servv[1],"%d",&port)
10271 }
10272 //glog("LC","hostport=%v (%v : %v)",servv,host,port)
10273 gjport = fmt.Sprintf("%v:%v",host,port)
10274 gjserv = gjscheme + ":" + gjport + "/" + gjpath
10275
10276 glog("LS",fmtstring("listening at %v",gjserv))
10277 http.Handle("/"+gjpath,websocket.Handler(serv))
10278 err := errors.New("ws")
10279 if( err != nil ){
10280     // https://golang.org/pkg/net/http/#ListenAndServeTLS
10281     //err = http.ListenAndServeTLS(gjport,nil)
10282 }else{
10283     err = http.ListenAndServe(gjport,nil)
10284 }
10285 gchk("LE",err)
10286 }
10287
10288 func gj_client(argv []string) {
10289     glog("CS",fmtstring("connecting to %v",gshws_url))
10290     ws, err := websocket.Dial(gshws_url,"",gshws_origin)
10291     gchk("C",err)
10292
10293     var resb = make([]byte, GSHWS_MSGSIZE)
10294     for qi := 0; qi < 3; qi++ {
10295         req := fmtstring("Hello, GShell! (%v)",qi)
10296         _,err := ws.Write([]byte(req))
10297         glog("OM",fmtstring("(%v) %v",wn,req))
10298         gchk("OB",err)
10299         rn, rerr := ws.Read(resb)
10300         gchk("RE",rerr)
10301         glog("RM",fmtstring("(%v) %v",rn,string(resb)))
10302     }
10303     glog("CF","WS request finish")
10304 }
10305 //</span><-- end of class="gsh_src" -->
10306 //<details></details>
10307
10308 /*
10309 <details id="GJLink_Section"><summary>GJ Link</summary>
10310 <span id="GJLinkView" class="GJLinkView">
10311 <p>
10312 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
10313 </p>
10314
10315 <span id="GJLink_1">
10316 <div id="GJLink_ServerSet"></div>
10317
10318 <div>
10319 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10320 <span id="GJLink_Account"></span>
10321 </div>
10322
10323 <div>
10324 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10325 <span id="GJLink_SendArea"></span>
10326 </div>
10327
10328 <div id="ws0_log_container"></div>
10329 <span>
10330 </span>
10331
10332 <script>
10333 function setupGJLinkArea(){
10334     GJLink_ServerSet.innerHTML = '<'+span id="gj_serv_label"'+
10335         +'<span id="textField.textLabel">Server:<'+/span>
10336         +'<'+span id="gj_serv" class="textField textURL" contenteditable=<'/span>';
10337
10338     GJLink_Account.innerHTML = '<'+textarea id="gj_user" class="textField"><'+/textarea>
10339         +'<'+textarea id="gj_ukey" class="textField"><'+/textarea>
10340         +'<'+textarea id="gj_chan" class="textField"><'+/textarea>
10341         +'<'+textarea id="gj_ckey" class="textField"><'+/textarea>;
10342
10343     GJLink_SendArea.innerHTML =
10344     '<'+textarea id="gj_sendText" class="textField MsgText" cols=60 rows=2><'+/textarea>;
10345     ws0_log_container.innerHTML = '<'+textarea id="ws0_log" class="ws0_log" cols=100 rows=10 spellcheck="false"><'+/textarea>';
10346
10347 }
10348 function clearGJLinkArea(){
10349     GJLink_ServerSet.innerHTML = "";
10350     GJLink_Account.innerHTML = "";
10351     GJLink_SendArea.innerHTML = "";
10352     ws0_log_container.innerHTML = "";
10353 }
10354
10355 </script>
10356
10357 <script>
10358 function SetupGJLink(){
10359     setupGJLinkArea();
10360     SetupVisibleText(GJLink_1,gj_serv,'GJLinkSv');
10361     SetupVisibleText(GJLink_1,gj_user,'UserName');
10362     SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
10363     SetupVisibleText(GJLink_1,gj_chan,'ChannelName');
10364     SetupBlinderText(GJLink_1,gj_ckey,'ChanKey');
10365     SetupVisibleText(GJLink_1,gj_sendText,'Message');
10366     gj_serv.innerHTML = 'ws://localhost:9999/gjlink';
10367 }
10368
10369 function GJLink_init(){
10370     SetupGJLink();
10371 }
10372
10373 function iselem(eid){
10374     return document.getElementById(eid);
10375 }
10376
10377 function DestroyGJLink1(){
10378     clearGJLinkArea();
10379     if( iselem('gj_user') ){
10380         return;
10381     }
10382     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10383     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10384     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10385     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10386     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10387     if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
10388     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10389     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10390 }
10391 DestroyGJLink = DestroyGJLink1;
10392
10393 </script>
10394 //<details>
10395 /*
10396 <style>
10397 .GJDigest {
10398     display:none;
10399 }
10400 </style>
10401 <script id="HtmlCodeview-script">
10402 function showCodeX(otxa,code,prefix,postfix,sign){
10403     txa=document.createElement('textarea');
10404     txa.id=otxa.id;
10405     txa.setAttribute('class','HtmlCodeviewText');
10406     otxa.parentNode.replaceChild(txa,otxa);
10407     txa.setBlinderText('spellcheck', false);
10408     //txa.innerHTML = code.innerHTML;
10409     //txa.innerHTML = code.outerHTML;
10410     txa.innerHTML = prefix + code.outerHTML + postfix;
10411     if( sign ){
10412         text = txa.value;
10413         tlen = txavalue.length;
10414         digit = tlen%32(txavalue,tlen) + ' ' + tlen
10415             + ' ' + code.id + (' '+ DateShort() + ')';
10416         //alert('digest: '+digest);
10417         console.log('digest: '+digest);
10418         txa.innerHTML += '<'+span class="GJDigest">' + digest + '<'+/span>\n';
10419     }
10420     txa.style.display = "block";
10421     txa.style.width = "100%";
10422     txa.style.height = "300px";
10423 }
10424
10425 function showHtmlCodeX(otxa,code,prefix,postfix,sign){
10426     if( event.target.value == 'ShowCode' ){
10427         showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign);
10428         event.target.value = 'HideCode';
10429     }else{
10430         otxa.style.display = "none";
10431         event.target.value = 'ShowCode';
10432     }
10433 }
10434 function showNodeAsHtmlSource(otxa,code){
10435     showNodeAsHtmlSourceX(otxa,code,'','');
10436 }
10437
10438 function showHtmlCode(otxa,code){
10439     if( event.target.value == 'ShowCode' ){
10440         showNodeAsHtmlSource(otxa,code);
10441         event.target.value = 'HideCode';
10442     }else{
10443         otxa.style.display = "none";
10444     }
10445 }

```

```

10443     event.target.value = 'ShowCode';
10444   }
10445 }
10446 </script>
10447 <style id="HtmlCodeview-style">
10448   .HtmlCodeviewText {
10449     font-size:10pt;
10450     font-family:Courier New;
10451     white-space:pre;
10452   }
10453   .HtmlCodeViewButton {
10454     padding:pt !important;
10455     line-height:1.1 !important;
10456     border:2px solid #bbb !important;
10457     font-size:1pt !important;
10458     font-weight:normal !important;
10459     font-family:Georgia !important;
10460     border-radius:3px !important;
10461     color:#ddd; background-color:#228 !important;
10462   }
10463 </style>
10464 /*
10465 */
10466 /*
10467 <details><summary>Live HTML Snapshot</summary>
10468 <span id="LiveHTML">
10469   <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
10470   <div class="GshMenu">
10471     <span class="GshMenu1" onclick="html_edit();">Edit</span>
10472     <span class="GshMenu1" onclick="html_save();">Save</span>
10473     <span class="GshMenu1" onclick="html_load();">Load</span>
10474     <span class="GshMenu1" onclick="html_ver0();">Vers</span>
10475   </div>
10476   <div>
10477     <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLcode()">
10478     <span id="LiveHTML_Codeview"></span>
10479   </div>
10480   <script id="LiveHTMLscript">
10481     function showLiveHTMLcode(){
10482       showHTMLCode(LiveHTML_Codeview,LiveHTML);
10483     }
10484     var editable = false;
10485     var savSuppressGJShell = false;
10486     function ToggleEditMode(){
10487       editable = ! editable;
10488       if(!editable){
10489         savSuppressGJShell = SuppressGJShell;
10490         SuppressGJShell = true;
10491         gsh.setAttribute('contenteditable','true');
10492         GshMenuEdit.innerHTML = 'Lock';
10493         GshMenuEdit.style.color = 'rgba(255,0,0,1)';
10494         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10495       }else{
10496         SuppressGJShell = savSuppressGJShell;
10497         gsh.setAttribute('contenteditable','false');
10498         GshMenuEdit.innerHTML = 'Edit';
10499         GshMenuEdit.style.color = 'rgba(16,160,16,1)';
10500         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10501       }
10502     }
10503     function html_edit(){
10504       ToggleEditMode();
10505     }
10506   </script>
10507   // Live HTML (DOM) Snapshot onto browser's localStorage
10508   <!-- 2020-0923 SatoxITS
10509   var htRoot = gsh // -- Element-ID, should be selectable
10510   const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
10511   // -- should be a [map] of URL
10512   // -- should be with CSSOM as inline script
10513   const htVersionTag = 'VersionTag'; // VesionTag Element-ID in the HTML (in DOM)
10514   function showVersion(note,w,v,u,t){
10515     w.alert(note+'\n' + v + '\n' + u + '\n' + t);
10516     + '-- URL: ' + u + '\n'
10517     + '-- Time: ' + t + '(' + DateLong0(t*1000) + ')';
10518   }
10519 }
10520 function html_save(){
10521   u = document.URL;
10522   t = new Date().getTime() / 1000;
10523   v = '<'+span id="htVersionTag"' data-url="'+u+'" data-time="'+t+'">';
10524   v += '<'+span>'+t;
10525   h = v + htRoot.outerHTML;
10526   localStorage.setItem(snappedHTML,h);
10527   showVersion("Saved",window,v,u,t);
10528 }
10529 function html_load(){
10530   h = localStorage.getItem(snappedHTML);
10531   if( h == null ){
10532     alert('No snapshot taken yet');
10533     return;
10534   }
10535   w = window.open('', '', '');
10536   d = w.document;
10537   d.write(h);
10538   w.focus();
10539   html_ver1("Loaded",w,d);
10540 }
10541 function html_ver1(note,w,d){
10542   if( (v = d.getElementById(htVersionTag)) != null ){
10543     h = v.outerHTML;
10544     u = v.getAttribute('data-url');
10545     t = v.getAttribute('data-time');
10546   }else{
10547     u = 'No version info. in the page';
10548     t = 0;
10549   }
10550   showVersion(note,w,v,u,t);
10551 }
10552 function html_ver0(){
10553   html_ver1("Version",window,document);
10554 }
10555 </script>
10556 <!-- LiveHTML -->
10557 <span>
10558 </details>
10559 /*
10560 */
10561 /*
10562 <details><summary>Event sharing</summary>
10563 <span id="EventSharingCodeSpan">
10564
10565 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
10566 <div id="fittestTemplate" class="fittest" hidden="">
10567   <style>body{ color:#323; font-family:Georgie,serif;font-size:10pt; </style>
10568   <span id="frameBody" class="fittestbody" onclick="frameClick()><script>
10569     function docClick(txt){
10570       document.body.append(txt);
10571       window.scrollTo(0,100000);
10572     }
10573     function frameClick(){
10574       xy = '(x:'+event.x + ' y:'+event.y+')';
10575       //docadd('Got Click on #' +event.target.id+ ' '+xy+ '\n');
10576       docadd('Got Click on #' +Fid.value+ ', '+xy+ '\n');
10577       window.scrollTo(0,100000);
10578       window.parent.postMessage('OnClick: '+xy+',');
10579     }
10580     function frameMouseMove(){
10581       if( false ){
10582         document.body.append('Mousemove on #' +event.target.id+ ' ');
10583         x = event.clientX+event.y + '\n';
10584         peerWin = window.frames['frame'];
10585         document.body.append('Send to peer #' +peerWin+ ' '+ '\n');
10586         peerWin.scrollTo(0,100000);
10587         peerWin.postMessage('Hi!', '+');
10588       }
10589     }
10590     function frameKeyDown(){
10591       msg = 'Got Keydown: #' +Fid.value+ ', ('+event.code+')';
10592       docadd(msg+'\n');
10593       window.parent.postMessage(msg,'*');
10594     }
10595     function frameOnMessage(){
10596       docadd('Message ' + event.data + '\n');
10597       window.scrollTo(0,100000);
10598     }
10599     if( document.getElementById('Fid') ){
10600       frameBody.id = Fid.value;
10601       Fid.value = '';
10602       Fid.value = '';
10603       Fid.value = '';
10604       Fid.value = '';
10605       Fid.value = '';
10606       Fid.value = '';
10607       Fid.value = '';
10608       Fid.value = '';
10609       Fid.value = '';
10610       window.addEventListener('click',frameClick);
10611       window.addEventListener('keydown',frameKeyDown);
10612       window.addEventListener('message',frameOnMessage);
10613       window.addEventListener('mousemove',frameMouseMove);
10614       window.parent.postMessage('Hi parent, I am '+Fid.value,'*');
10615     }
10616   </script></span></div>
10617
10618 <style>.frameTest{margin:3px;resize:both;width:370px;height:120px;}</style>
10619 <h2>Inter-window communication</h2>

```

```

10620<note>
10621    frame0 >>> frame1 and frame2<br>
10622    frame1 >>> frame0 and frame2<br>
10623    frame2 >>> frame0 and frame1<br>
10624</note>
10625<div id="iframe-test">
10626<pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
10627<iframe id="frame0" title="iframe0" class="iframeTest"></iframe>
10628<iframe id="frame1" title="iframe1" class="iframeTest"></iframe>
10629<iframe id="frame2" title="iframe2" class="iframeTest"></iframe>
10630</div>
10631<script id="if0-test-script">
10632    function InterframeComm_init(){
10633        setupFrames0();
10634        setupFrames1();
10635        setupFrames2();
10636    }
10637    function setFrameSrcdoc(dst,src){
10638        if( true ){
10639            dst.contentWindow.document.write(src);
10640            // this makes browser waits close, and crash if accumulated !?
10641            // so it should be closed after write
10642            dst.contentWindow.document.close();
10643        }else{
10644            // to be erased before source dump
10645            // but should be set for live snapshot
10646            dst.srcdoc = src;
10647        }
10648    }
10649    function setupFrames0(){
10650        if0 = document.getElementById('frame0').contentWindow.document.body;
10651        iframe0.style.width = "75px";
10652        //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
10653        window.addEventListener('message',messageFromChild);
10654
10655        if0 = '';
10656        if0 += '<pre style="font-family:Courier New;">';
10657        if0 += '<input id="Fid" value="iframe0">';
10658        if0 += iftestTemplate.innerHTML;
10659        setFrameSrcdoc(iframe0,if0);
10660
10661        function clickOnChild(){
10662            console.log('clickOn #' + this.id);
10663        }
10664        function moveOnChild(){
10665            console.log('moveOn #' + this.id);
10666        }
10667        iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10668        iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10669    }
10670    function setupFrames1(){
10671        if1 = '<input id="Fid" value="iframe1">';
10672        if1 += iftestTemplate.innerHTML;
10673        setFrameSrcdoc(iframe1,if1);
10674        //iframe1.name = 'iframe1'; // this seems break contentWindow
10675
10676        if2 = '<input id="Fid" value="iframe2">';
10677        if2 += iftestTemplate.innerHTML;
10678        setFrameSrcdoc(iframe2,if2);
10679
10680        iframe1.addEventListener('message',messageFromChild);
10681        //iframe1.addEventListener('mousemove',moveOnChild);
10682        iframe2.addEventListener('message',messageFromChild);
10683        //iframe2.addEventListener('mousemove',moveOnChild);
10684        iframe1.contentWindow.postMessage('parent Hi iframe1 -- from parent.', '*');
10685        //iframe1.contentWindow.postMessage('parent peer is '+iframe2.contentWindow,'*');
10686        iframe2.contentWindow.postMessage('parent0 Hi iframe2 -- from parent.', '*');
10687        //iframe2.contentWindow.postMessage('You peer is '+iframe1.contentWindow,'*');
10688
10689    function messageFromChild(){
10690        from = null;
10691        forw = null;
10692        if( event.source == iframe0.contentWindow ){
10693            from = 'iframe0';
10694            forw = 'iframe2';
10695        }else{
10696            if( event.source == iframe1.contentWindow ){
10697                from = 'iframe1';
10698                forw = 'iframe2';
10699            }else{
10700                if( event.source == iframe2.contentWindow ){
10701                    from = 'iframe2';
10702                    forw = 'iframe1';
10703                }else
10704                {
10705                    iframeHost.innerHTML += 'Message [unknown] '
10706                    + 'orig=' + event.origin
10707                    + 'data=' + event.data
10708                    //+ 'from=' + event.source
10709                    ;
10710                msglog1 = from + event.data + ' -- '
10711                msglog1 += from + ' event.source'
10712                + ' orig=' + event.origin
10713                + ' name=' + event.source.name
10714                //+ ' port=' + event.ports
10715                //+ ' evid=' + event.lastEventId
10716                + '\n';
10717                if( true ){
10718                    if( forw == 'iframe1' || forw == 'iframe2' ){
10719                        iframe1.contentWindow.postMessage(from+event.data);
10720                    }
10721                    if( forw == 'iframe2' || forw == 'iframe1' ){
10722                        iframe2.contentWindow.postMessage(from+event.data);
10723                    }
10724                }
10725            }
10726        txtadd0(msglog1);
10727
10728        function txtadd0(txt){
10729            iframe0.contentWindow.document.body.append(txt);
10730            iframe0.contentWindow.scrollTo(0,100000);
10731        }
10732    }
10733    function es_ShowSelf(){
10734        iframe1.setAttribute('src',document.URL);
10735        iframe2.setAttribute('src',document.URL);
10736    }
10737</script>
10738<input class="htmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()"/>
10739<input class="htmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()"/>
10740<span id="EventSharingCodeview"></span>
10741<script id="EventSharingScript">
10742function es_showHtmlCode(){
10743    showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
10744}
10745DestroyEventSharingCodeview = function(){
10746    //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
10747    EventSharingCodeview.innerHTML = "";
10748    iframe0.style = "";
10749    //iframe0.srcdoc = "erased";
10750    //iframe1.srcdoc = "erased";
10751    //iframe2.srcdoc = "erased";
10752}
10753</script>
10754<!-- EventSharing -->
10755</span>
10756</details>
10757</div>
10758</div>
10759</div>
10760</div>
10761<!-- ----- "GShell Inside" Notification ( -->
10762<script id="script-gshell-inside">
10763var notices = 0;
10764function noticeGShellInside(){
10765    ver = '';
10766    if( ver = document.getElementById('GshVersion') ){
10767        ver = ver.innerHTML;
10768    }
10769    notices += 1;
10770    console.log('GJShell Inside (^~^) //+'+ver);
10771    notices += 1;
10772    if( 2 <= notices ){
10773        document.removeEventListener('mousemove',noticeGShellInside);
10774    }
10775}
10776document.addEventListener('mousemove',noticeGShellInside);
10777noticedShellInside();
10778
10779const FooterName = 'GshFooter';
10780const FooterNameFooter = '';
10781if( footer = document.getElementById(FooterName) != null ){
10782    //footer.parentNode.removeChild(footer);
10783    empty = document.createElement('div');
10784    empty.id = 'GshFooter0';
10785    footer.parentNode.replaceChild(empty,footer);
10786}
10787
10788function showFooter(){
10789    footer = document.createElement('div');
10790    footer.id = FooterName;
10791    footer.innerHTML = 'QRcodeImage = url:"'+QRMoreQR+"";
10792    //footer.parentNode.appendChild(footer);
10793    if( document.getElementById('GshFooter0') != null ){
10794        GshFooter0.parentNode.replaceChild(footer,GshFooter0);
10795    }
10796}

```

```
10797</script>
10798<!-- } -->
10799<!--
10800<!--
10801 border:20px inset #888;
10802-->
10803
10804 //<span id="VirtualDesktopCodeSpan">
10805<details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
10806<!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
10807<style>
10808.VirtualSpace {
10809    z-index:0;
10810    width:1280px !important; xheight:720px !important;
10811    width:1280px; height:2880px;
10812    border-width:0px;
10813    xxbackground-color:rgba(32,32,160,0.8);
10814    xxbackground-image:url("WD-WallPaper03.png");
10815    xxbackground-size:100% 100%;
10816    color:#22a; xfont-family:Georgia;font-size:10pt;
10817    xoverflow:scroll;
10818}
10819.VirtualGrid {
10820    z-index:0;
10821    position:absolute;
10822    width:800px; height:500px;
10823    border:1px inset #fff;
10824    color:rgba(192,255,192,0.8);
10825    font-family:Georgia, Courier New;
10826    text-align:center;
10827    vertical-align:middle;
10828    font-size:200px;
10829    text-shadow:4px 4px #ccf;
10830}
10831.WD_Gridscroll {
10832    z-index:10000;
10833    background-color:rgba(200,200,200,0.1);
10834}
10835.VirtualDesktopTop {
10836    z-index:0;
10837    position:relative;
10838    resize:both !important;
10839    overflow:scroll;
10840    display:block;
10841    min-width:120px !important; min-height:60px !important;
10842    width:800px;
10843    height:500px;
10844    border:10px inset #228;
10845    border-width:30px; border-radius:20px;
10846    background-image:url("WD-WallPaper03.png");
10847    background-size:100% 100%;
10848    color:#22a;font-family:Georgia;font-size:10pt;
10849}
10850}
10851.comment {
10852    /* overflow=scroll seems to bound childrens' view in the element span
10853     * // specifying overflow seems fix the position of the element
10854 */
10855.VirtualBrowserSpan {
10856    z-index:0;
10857    xxborder:0.5px dashed #fff !important;
10858    border-color:rgba(255,255,255,0.5) !important;
10859    position:relative;
10860    left:100px;
10861    top:100px;
10862    display:block;
10863    resize:both !important;
10864    width:540px;
10865    height:320px;
10866    min-width:120px !important; min-height:20px !important;
10867    max-width:120px !important; max-height:2880px !important;
10868    background-color:rgba(255,200,255,0.1);
10869    xoverflow:scroll;
10870}
10871.xVirtualBrowserlocationBar:focus {
10872    color:#f00;
10873    background-color:rgba(255,128,128,0.2);
10874}
10875.xVirtualBrowserlocationBar:active {
10876    color:#f00;
10877    background-color:rgba(128,255,128,0.2);
10878}
10879a.VirtualBrowserlocation {
10880    color:#ccc !important;
10881    text-decoration:none !important;
10882}
10883a.VirtualBrowserlocation:hover {
10884    color:#fff !important;
10885    text-decoration:underline;
10886}
10887.VirtualBrowserlocationBar {
10888    position:absolute;
10889    z-index:20000;
10890    display:block;
10891    width:400px;
10892    height:20px;
10893    padding-left:2px;
10894    line-height:1.1;
10895    vertical-align:middle;
10896    font-size:14px;
10897    color:#fff;
10898    background-color:rgba(128,128,128,0.2);
10899    font-family:Georgia;
10900}
10901.VirtualBrowserCommandBar {
10902    position:absolute;
10903    z-index:200000;
10904    xxdisplay:inline;
10905    display:block;
10906    width:20px;
10907    height:20px;
10908    line-height:1.1;
10909    vertical-align:middle;
10910    font-size:14px;
10911    color:#fdd;
10912    background-color:rgba(128,128,128,0.1);
10913    font-family:Georgia;
10914    text-align:left;
10915    left:404px;
10916}
10917.VirtualBrowserFrame {
10918    position:relative;
10919    position:absolute;
10920    xxdisplay:inline;
10921    display:block;
10922    z-index:10;
10923    resize:both !important;
10924    width:800px; height:240px;
10925    min-width:0px; min-height:30px;
10926    max-width:120px; max-height:2880px;
10927    border-radius:6px;
10928    background-color:rgba(255,255,255,0.9);
10929    border-top:2px solid;
10930    border-right:4px solid;
10931    border-bottom:10px solid;
10932}
10933.WinPavIcon {
10934    width:16px;
10935    height:16px;
10936    margin:1px;
10937    margin-right:3px;
10938    vertical-align:middle;
10939    background-color:rgba(255,255,255,1.0);
10940}
10941.VirtualDesktopMenuBar {
10942    xposition:absolute;
10943    color:#fff;
10944    font-size:1pt;
10945    text-align:right;
10946    padding-right:4px;
10947    background-color:rgba(128,128,128,0.7);
10948}
10949.VirtualdesktopCalender {
10950    color:#fff;
10951    font-size:2pt;
10952    text-align:right;
10953    padding-right:4px;
10954    xxbackground-color:rgba(255,255,255,0.2);
10955}
10956.xxxxinput { width:260px !important; line-height:1.1 !important; margin:1px;
10957    display:inline !important; font-size:10pt !important; padding:1px !important;
10958}
10959.WD_Config {
10960    display:inline !important;
10961    padding:1px !important;
10962    font-size:10pt !important;
10963    width:100px !important;
10964    height:12pt !important;
10965    line-height:1.0pt !important;
10966    height:15pt !important;
10967}
10968.WD_Button {
10969    display:inline !important;
10970    padding:2px !important;
10971    color:#fff !important;
10972    background-color:#228 !important;
10973    font-size:10pt !important;
```

```

10974 width:60pt !important;
10975 height:12pt !important;
10976 line-height:1.0pt !important;
10977 height:16pt !important;
10978 border:2px inset #44a !important;
10979 }
10980 .WD_Href {
10981 display:inline !important;
10982 padding:2px !important;
10983 font-size:9pt !important;
10984 width:12pt !important;
10985 height:12pt !important;
10986 line-height:1.0pt !important;
10987 height:15pt !important;
10988 }
10989 .LiveHtmlCodeviewText {
10990 font-size:10pt;
10991 font-family:Courier New;
10992 white-space:pre;
10993 }
10994 }
10995 .WD_Panel {
10996 x-index:100 !important;
10997 opacity:1.0 !important;
10998 margin-left:25px !important;
10999 width:800px !important;
11000 padding:4px !important;
11001 border:1px solid #888 !important;
11002 border-radius:6px !important;
11003 background-color:rgba(220,220,220,0.9) !important;
11004 font-size:9pt !important;
11005 font-family:Courier New;
11006 }
11007 }
11008 .WD_Help {
11009 font-size:10pt !important;
11010 font-family:Courier New;
11011 line-height:1.2 !important;
11012 color:#000 !important;
11013 width:100% !important;
11014 background-color:rgba(240,240,255,0.8) !important;
11015 }
11016 }
11017 .WB_Zoom {
11018 }
11019 </style>
11020 <h2>CosmoScreen 0.0.8</h2>
11021 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
11022 q ... grid on/off<br>
11023 i ... zoom in<br>
11024 o ... zoom out<br>
11025 s ... save current scope<br>
11026 r ... restore saved scope<br>
11027 </span>
11028 </menu>
11029 <div class="WD_Panel" draggable="true">
11030 <p><!-- should be on the frame of the WD -->
11031 <input id="WD_Rotate" class="WD_Button" type="button" value="Resize">
11032 <input id="WD_Width_1" class="WD_Config" type="text">
11033 x <input id="WD_Height_1" class="WD_Config" type="text">
11034 wall-paper: <a href="WD_WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaler03.png</a>
11035 </p>
11036 <div>
11037 <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11038 <input id="WS_1_Width" class="WD_Config" type="text">
11039 x <input id="WS_1_Height" class="WD_Config" type="text">
11040 </div>
11041 <div>
11042 <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
11043 <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
11044 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
11045 <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="zoomOut">
11046 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="zoomIn">
11047 </div>
11048 <div>
11049 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
11050 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
11051 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
11052 shift+wheel for horizontal scroll
11053 </div>
11054 <div>
11055 Scopex <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
11056 <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11057 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
11058 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11059 </div>
11060 <div>
11061 Scopely <input id="WD_Zoom_1_Force" class="WD_Button" type="button" value="Force">
11062 <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
11063 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
11064 <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
11065 </div>
11066 <div>
11067 Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
11068 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
11069 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
11070 <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
11071 </div>
11072 <div>
11073 <p>
11074 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
11075 </div>
11076 <div>
11077 Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
11078 scroll imprisons windows inside the display
11079 </div>
11080 </div>
11081 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
11082 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
11083 <span id="CosmoScreen 0.0.8</span><span id="VirtualDesktop_1_Clock"></span>
11084 </div>
11085 <div id="VirtualDesktop_1_Calendar" class="VirtualDesktopCalendar" >0:00</div>
11086 <div align="right">h1>VirtualSpace</div>
11087 <div id="VirtualDesktop_1_Content" class="VirtualSpace">
11088 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11089 <div id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
11090 <iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""/></div>
11091 </div>
11092 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11093 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
11094 <span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
11095 <iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""/></div>
11096 </div>
11097 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11098 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
11099 <span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
11100 <iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""/></div>
11101 </div>
11102 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
11103 </div>
11104 <div>
11105 <input class="HTMLCodeViewButton" type="button" value="ShowCode" onclick="vd_showHTMLCode()">
11106 <span id="VirtualDesktopScript">
11107 <script id="VirtualDesktopScript">
11108 function vd_showHTMLCode(){
11109 codeSpan = document.getElementById('VirtualDesktopCodeSpan');
11110 showHTMLCode(VirtualDesktopCodeview,codeSpan);
11111 VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
11112 }
11113 DestroyEventSharingCodeview = function(){
11114 VirtualDesktopCodeview.innerHTML = "";
11115 }
11116
11117 function wdlog(log){
11118 if(GJ Channel != null ){
11119 GJ SendMessage("WD "+log);
11120 }
11121 console.log(log);
11122 }
11123
11124 function onEnterWin(e){
11125 if(GJ Channel != null ){
11126 GJ SendMessage("WD "+e.id);
11127 }
11128 console.log(log);
11129 var topMostWin = 10000;
11130 function onEnterWin(e){
11131 t = e.target;
11132 oindex = t.style.zIndex;
11133 if(oindex <= 1) oindex = 0;
11134 //t.style.zIndex = oindex;
11135 //t.style.zIndex = 10000;
11136 topMostWin += 1;
11137 t.style.zIndex = topMostWin;
11138 t.style.zIndex = topMostWin;
11139 nindex = t.style.zIndex;
11140 VirtualDesktopCodeview.setAttribute('id'+('+'+oindex+'->'+nindex+''));
11141 e.stopPropagation();
11142 e.preventDefault();
11143 }
11144 function onClickWin(e){ // can detect click on the thick border? t = e.target;
11145 oindex = t.style.zIndex;
11146 topMostWin += 1;
11147 t.style.zIndex = topMostWin;
11148 nindex = t.style.zIndex;
11149 wdlog('Click '+t.id+' #'+t.id+'('+oindex+'->'+nindex+')');
11150 //e.stopPropagation();
}

```

```

11151 //e.preventDefault();
11152 }
11153 function onLeaveWin(e){
11154   t = e.target;
11155   //oindex = t.style.zIndex;
11156   //nindex = t.saved_zIndex;
11157   //t.style.zIndex = nindex;
11158   //wdlog('Leave '+e.target+' #' +e.target.id+'('+oindex+'->' +nindex+')');
11159   e.stopPropagation();
11160   e.preventDefault();
11161 }
11162 }
11163 var Windragstartx; // event
11164 var Windragstartt;
11165 var WindragstartX; // target
11166 var WindragstartTY;
11167
11168 function onWinDragstart(e){
11169   WindragstartX = e.x;
11170   Windragstartt = e.y;
11171
11172   t = e.target;
11173
11174 //WindragstartPX = t.getBoundingClientRect().left.toFixed(0)
11175 //WindragstartPY = t.getBoundingClientRect().top.toFixed(0)
11176 if( t.style.left == ''){
11177   WindragstartX = x0 = 0;
11178   t.style.left = '0px';
11179 }else{
11180   //WindragstartPX = x0 = Number(t.style.left);
11181   WindragstartX = x0 = parseInt(t.style.left);
11182 }
11183 if( t.style.top == '' ){
11184   WindragstartY = y0 = 0;
11185   t.style.top = '0px';
11186 }else{
11187   WindragstartY = y0 = Number(t.style.top);
11188   WindragstartY = y0 = parseInt(t.style.top);
11189 }
11190 if( true ) // to be undo
11191 t.wasAtX = WindragstartX;
11192 t.wasAtY = WindragstartY;
11193
11194 wdlog('DragSTA #' +t.id
11195   + ' event(' +e.x+ ',' +e.y+ ')
11196   + ' position=' + t.style.position
11197   + ' style left,top(' +t.style.left+ ',' +t.style.top+ ')
11198 );
11199 e.stopPropagation();
11200 //e.preventDefault();
11201 return true;
11202 }
11203 function onWinDragEvent(wh,e,set,dolog){
11204   t = e.target;
11205   dx = e.x - WindragstartX;
11206   dy = e.y - WindragstartY;
11207   nx = WindragstartX + dx;
11208   ny = WindragstartY + dy;
11209   log += 'DragEvent'
11210   + ' event(' + 'WinDragstartX', '+WinDragstartY'
11211   + ' event(' +e.x+ ',' +e.y+ ')
11212   + ' diff(' +dx+ ',' +dy+ ')
11213   + ' (' + nx + ',' + ny + ')
11214   + ' (' + t.style.left + ',' + t.style.top + ')
11215   + ' wasAt(' + t.wasAtX + ',' + t.wasAtY + ')'
11216 ;
11217 if( e.x != 0 || e.y != 0 ){
11218   if( set == true ){
11219     //t.style.x = nx + 'px'; // not effective
11220     //t.style.y = ny + 'px'; // not effective
11221     t.style.left = nx + 'px';
11222     t.style.top = ny + 'px';
11223     log += ' Set';
11224   }else{
11225     log += ' What?'; // the type is event start?
11226     if( !dolog ){
11227       log = '';
11228     }
11229   }
11230 }else{
11231   log += ' What?'; // the type is event start?
11232   if( !dolog ){
11233     log = '';
11234   }
11235 }
11236 if( (and(dolog, log != '') ){
11237   wdlog(log);
11238 }
11239 if( true ){
11240   // should be propagated to parent in FireFox ?
11241   e.stopPropagation();
11242 }
11243 e.preventDefault();
11244 return false;
11245 }
11246 function onWinDrag(e){
11247   return onWinDragEvent('Ing',e,true,false);
11248 }
11249 function onWinDragend(e){
11250   return onWinDragEvent('End',e,false,true);
11251 }
11252 function onWinDragexit(e){
11253   return onWinDragEvent('Exit',e,false,true);
11254 }
11255 function onWinDragover(e){
11256   return onWinDragEvent('Over',e,false,true);
11257 }
11258 function onWinDrainger(e){
11259   return onWinDragEvent('Enter',e,false,true);
11260 }
11261 function onWinDragleave(e){
11262   return onWinDragEvent('Leave',e,false,true);
11263 }
11264 function onWinDragdrop(e){
11265   return onWinDragEvent('Drop',e,false,true);
11266 }
11267 function faviconChange(e){
11268   wdlog('--Favicon #' +e.target.id+ ' href=' +e.details.href);
11269 }
11270 var savedSuppressGJShell = false;
11271 function stopGJShell(e){
11272   //wdlog('Enter Gsh STOP');
11273   savedSuppressGJShell = SuppressGJShell;
11274   SuppressGJShell = true;
11275   e.stopPropagation();
11276   e.preventDefault();
11277 }
11278 function contGJShell(e){
11279   //wdlog('leave Gsh STOP');
11280   SuppressGJShell = savedSuppressGJShell;
11281   e.stopPropagation();
11282   e.preventDefault();
11283 }
11284 function WD_onKeyDown(e){
11285   keycode = e.code;
11286   console.log('Keydown #' +e.target.id+ ' ' +keycode);
11287   if(keycode == 'KeyG'){
11288     WD_setGrid1(WD_Grid_1);
11289   }else
11290   if( keycode == 'KeyI' ){
11291     WD_docomIN();
11292   }else
11293   if( keycode == 'KeyO' ){
11294     WD_docomOUT();
11295   }else
11296   if( keycode == 'KeyR' ){
11297     WD_RestoreScope(null);
11298   }else
11299   if( keycode == 'KeyS' ){
11300     WD_SaveScope(null);
11301   }
11302   e.stopPropagation();
11303   e.preventDefault();
11304 }
11305 function WD_onKeyUp(e){
11306   e.stopPropagation();
11307   e.preventDefault();
11308 }
11309 function WD_EventSetup(){
11310   VirtualDesktop_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
11311   VirtualDesktop_1.Content.addEventListener('keydown', e => { WD_onKeyDown(e); });
11312   WD_Help_1.addEventListener('keydown', e => { WD_onKeyDown(e); });
11313   WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
11314 }
11315 function settleWin(s,cmd,f,u,w,h,x,y,c,scale,1){
11316   command = cmd.innerHTML;
11317   if( command == "Reload" ){
11318     href_id = e.target.href.id;
11319     d = document.getElementById(href_id);
11320     wdlog(' href id=' + href_id);
11321     wdlog(' href=' + href_id + '\n elem#' + href_id + '\n href=' + d);
11322     url = d.innerHTML;
11323     wdlog('---- Load href tag #' + href_id + '\n elem #' + href_id + '\n href=' + href_id);
11324     wdlog('---- Load target #' + f.id + '\n href=' + url);
11325     wdlog('---- Load target #' + f.id + '\n href=' + url);
11326     f.src = url;
11327   }
11328 }

```

```

11328     }else{
11329         alert('unknown command'+command+' '+e.target.id+','+l.id+','+f.id);
11330     }
11331 }
11332 function onKeyDown(e){
11333     if( e.code == 'Enter' ){
11334         e.stopPropagation();
11335         e.preventDefault();
11336     }
11337 }
11338 function onKeyUp(e){
11339     if( e.code == 'Enter' ){
11340         e.stopPropagation();
11341         e.preventDefault();
11342         // should reload immediately ?
11343     }
11344 }
11345
11346 if( false ){
11347     wdlog('start settle VirtualBrowser url='+u+'\n'
11348         + 'id=' + s.id + '\n'
11349         + 'width=' + s.style.width + '\n'
11350         + 'height=' + s.style.height
11351 );
11352 }
11353 // very important for WordPress ???
11354 s.style.width = f.style.width = 501; // for WordPress ...??
11355 s.style.height = f.style.height = 271; // for WordPress ...??
11356 if( false ){
11357     wdlog('midway settle VirtualBrowser url='+u+'\n'
11358         + 'id=' + s.id + '\n'
11359         + 'width=' + s.style.width + '\n'
11360         + 'height=' + s.style.height
11361 );
11362 }
11363 s.width = 502; // for WordPress ...??
11364 s.height = 272; // for WordPress ...??
11365 if( false ){
11366     wdlog('midway-2 settle VirtualBrowser url='+u+'\n'
11367         + 'id=' + s.id + '\n'
11368         + 'span-width' + s.width + '\n'
11369         + 'span-height' + s.height
11370 );
11371 }
11372
11373 s.style.width = w + 'px';
11374 s.style.height = w + 'px';
11375 f.style.width = w + 'px';
11376 f.style.height = h + 'px';
11377 //f.style.setProperty('--transform','scale('+scale+')');
11378 f.style.setProperty('transform','scale('+scale+')');
11379
11380 //wdlog('--x-- u=' + u + ' width=' + s.style.width + ',f=' + f.style.width);
11381 //wdlog('--x-- u=' + u + ' width=' + s.style.width + ',f=' + f.style.width);
11382 s.setAttribute('draggable','true');
11383 f.setAttribute('draggable','false'); // why necessary?
11384 l.setAttribute('draggable','false'); // why necessary?
11385 cmd.setAttribute('draggable','false'); // why necessary?
11386 s.addEventListener('dragstart',e => { onWinDragStart(e); });
11387 s.addEventListener('drag', e => { onWinDrag(e); });
11388 s.addEventListener('dragexit', e => { onWinDragExit(e); });
11389 s.addEventListener('dragend', e => { onWinDragEnd(e); });
11390 s.addEventListener('dragenter', e => { onWinDragEnter(e); });
11391 s.addEventListener('dragover', e => { onWinDragOver(e); });
11392 s.addEventListener('dragleave', e => { onWinDragLeave(e); });
11393 s.addEventListener('drop', e => { onWinDrop(e); });
11394
11395 s.addEventListener('mouseenter',e => { onEnterWin(e); });
11396 s.addEventListener('mouseleave',e => { onLeaveWin(e); });
11397
11398 if( false ){
11399     s.style.position = "absolute";
11400     s.style.x = x+'px';
11401     s.style.left = x+'px';
11402     s.style.y = y+'px';
11403     s.style.top = y+'px';
11404 }
11405 else{
11406     s.style.setProperty('position','absolute','important');
11407     s.style.setProperty('x',x+'px','important');
11408     s.style.setProperty('left',x+'px','important');
11409     s.style.setProperty('y',y+'px','important');
11410     s.style.setProperty('top',y+'px','important');
11411 }
11412
11413 favicon = './favicon.ico';
11414 u1 = u.split('/');
11415 if( 2 <= u1.length ){
11416     u2 = u1[1].split('/');
11417     if( 2 <= u2.length ){
11418         if( u1[0] == 'file' ){
11419             //favicon = 'file://' + u2.slice(0,uv2.length-1).join('/');
11420             //favicon = '/'+favicon.ico;
11421             favicon = './favicon.ico';
11422         }else{
11423             favicon = u1[0] + ':' + u2[0] + '/favicon.ico';
11424         }
11425     }
11426 }
11427 //wdlog('---- favicon-url=' + favicon);
11428 href_id = l.id + '_href';
11429 l.innerHTML = '';
11430 l.innerHTML += '<a href="#" id="VirtualBrowserLocation" href="'+u+'">' + u + '</a>';
11431 l.addEventListener('click',e => { onWinClick(e); });
11432 l.addEventListener('mouseleave',e => { stopShell(e); });
11433 l.addEventListener('mouseenter',e => { contShell(e); });
11434 l.addEventListener('keydown',e => { onKeyDown(e); });
11435 l.addEventListener('keyup',e => { onKeyUp(e); });
11436
11437 cmd.href_id = href_id;
11438 wdlog('()cmd#'+cmd.id);
11439 wdlog('()href id#'+href_id);
11440 wdlog('()2ref id#'+cmd.href_id);
11441 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11442
11443 f.style.borderColor = c;
11444 f.srcc = u;
11445 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
11446 f.addEventListener('mouseleave',e => { onLeaveWin(e); });
11447
11448 //s.addEventListener('click', e => { onClickWin(e); });
11449 //f.addEventListener('click', e => { wdlog('click wbl'); });
11450 f.addEventListener('mozbrowsericonchange',onFaviconChange);
11451
11452 wdlog('done settle VirtualBrowser url=' + u + '\n'
11453     + 'id=' + s.id + '\n'
11454     + 'width=' + s.style.width + '\n'
11455     + 'height=' + s.style.height + '\n'
11456     + 'cmd=' + cmd.id);
11457 );
11458
11459
11460 function WD_EventSetup2(){
11461     dt = VirtualDesktop_1;
11462     dt.style.width = "800px";
11463     dt.style.height = "500px";
11464     dt.addEventListener('dragstart',e => { onWinDragStart(e); });
11465     dt.addEventListener('drag', e => { onWinDrag(e); });
11466     dt.addEventListener('exit', e => { onWinDragExit(e); });
11467 }
11468
11469 function GRonClick(){
11470     WD_SaveScreen();
11471     WD_SetEvent();
11472     t = event.target;
11473     x = t.getAttribute('data-leftx');
11474     y = t.getAttribute('data-topy');
11475     zoom = WD_Zoom_1_XY.value;
11476     x *= zoom;
11477     y *= zoom;
11478     WD_doScroll1XY(event,x,y);
11479 }
11480 function WD_SetGrid(e){
11481     t = e.target;
11482     WD_SetGrid(t);
11483 }
11484 function WD_SetGrid(t){
11485     //ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11486     ds = VirtualDesktop_1_GridPlane;
11487     if( t.value == 'GridOn' ){
11488         for( col = 0; col < 16; col++ ){
11489             for( row = 0; row < 16; row++ ){
11490                 g1 = document.createElement('span');
11491                 g1.setAttribute('class','VirtualGrid');
11492                 leftx = col * 800;
11493                 topx = row * 500;
11494                 g1.col = col;
11495                 g1.row = row;
11496                 label = '<' + span +
11497                     + 'id=' + gid + '' + 'class="WD_GridScroll"' +
11498                     + 'contenteditable="false" onclick="GRonClick()" ' +
11499                     + 'data-leftx' + leftx + ' ' + 'data-topy' + topx + ' ' +
11500                     + 'grid=' + grid + '' + span + '';
11501                 console.log('grid '+label);
11502                 g1.innerHTML = label;
11503                 g1.position = 'relative';
11504                 g1.leftx = leftx;

```

```

11505         gl.topy = topy;
11506         gl.style.left = gl.leftx + 'px';
11507         gl.style.top = gl.topy + 'px';
11508         if( col % 2 == row % 2 ){
11509             gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
11510         }
11511         ds.appendChild(gl);
11512     }
11513     t.value = 'GridOff';
11514 }else{
11515     ds.innerHTML = '';
11516     t.value = 'GridOn';
11517 }
11518 }
11519 }
11520
11521 var sav_scrollLeft;
11522 var sav_scrollTop;
11523 var sav_nscale;
11524 function WD_SaveScope(e){
11525     sav_scrollLeft = WD_Left_1.value;
11526     sav_scrollTop = WD_Top_1.value;
11527     sav_nscale = WD_Zoom_1_XY.value;
11528     //console.log('saved zoom=' +sav_oscale+',' +sav_nscale);
11529 }
11530 function WD_RestoreScope(e){
11531     WD_Zoom_1_XY.value = sav_nscale;
11532     WD_doZoom();
11533 }
11534 WD_Left_1.value = sav_scrollLeft;
11535 WD_Top_1.value = sav_scrollTop;
11536 WD_doscroll(null);
11537 }
11538 function ignoreEvent(e){
11539     e.stopPropagation();
11540     //e.preventDefault();
11541 }
11542 function zoomMag(){
11543     return WD_Zoom_1_MAG.value;
11544 }
11545 function WD_EventSetupP(){
11546     dt = VirtualDesktop_1_Content;
11547     dt_Save_1_Save.addEventListener('click', e => { WD_SetGrid(e); });
11548     WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11549     WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
11550     WD_Width_1.value = dt.style.width;
11551     WD_Width_1.addEventListener('keydown', ignoreEvent);
11552     WD_Width_1.addEventListener('keyup', ignoreEvent);
11553     WD_Height_1.addEventListener('keydown', ignoreEvent);
11554     WD_Height_1.addEventListener('keyup', ignoreEvent);
11555     WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
11556     WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
11557 }
11558
11559 function escale(e,oscale,nscale){
11600     e.style.setProperty('transform', 'scale(' + nscale + ')');
11601     rscale = oscale / nscale;
11602     w = parseInt(e.style.width);
11603     h = parseInt(e.style.height);
11604     w = w * rscale; //oscale/nscale;
11605     h = h * rscale; //oscale/nscale;
11606     e.style.width = w + 'px';
11607     e.style.height = h + 'px';
11608 }
11609 function scaleWD(ds,oscale,nscale){
11610     if( true )
11611         escale(VirtualBrowser_1,oscale,nscale);
11612     escale(VirtualBrowser_1_Location,oscale,nscale);
11613     escale(VirtualBrowser_1_Command,oscale,nscale);
11614     escale(VirtualBrowser_1_Frame,oscale,nscale);
11615 }
11616 function scaleWD(ds,oscale,nscale){
11617     escale(VirtualBrowser_2,oscale,nscale);
11618     escale(VirtualBrowser_2_Location,oscale,nscale);
11619     escale(VirtualBrowser_2_Command,oscale,nscale);
11620     escale(VirtualBrowser_2_Frame,oscale,nscale);
11621 }
11622 function scaleWD(ds,oscale,nscale){
11623     escale(VirtualBrowser_3,oscale,nscale);
11624     escale(VirtualBrowser_3_Location,oscale,nscale);
11625     escale(VirtualBrowser_3_Command,oscale,nscale);
11626     escale(VirtualBrowser_3_Frame,oscale,nscale);
11627 }
11628
11629 function WD_doZoom(){
11630     ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11631     oscale = WD_Zoom_1_XY.value; // hidden value for current zoom
11632     nscale = WD_Zoom_1_XY.value;
11633     ds.style.zoom = oscale;
11634     WD_Zoom_1_XY.value = ds.style.zoom;
11635     scaledWD(ds,oscale,nscale);
11636 }
11637 function WD_EventSetupP(){
11638     WD_Zoom_1_AddEventListener('click', WD_doZoom);
11639     WD_Zoom_1_XY.addEventListener('keydown', ignoreEvent);
11640     WD_Zoom_1_XY.addEventListener('keyup', ignoreEvent);
11641 }
11642
11643 function WD_doZoomOUT(){
11644     ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11645     oscale = WD_Zoom_1_XY.value;
11646     if( oscale == 0 || oscale == '' ){
11647         oscale = 1;
11648     }
11649     oscale = oscale / zoomMag();
11650     ds.style.zoom = oscale;
11651     WD_Zoom_1_XY.value = ds.style.zoom;
11652     WD_Zoom_1_XY.value = ds.style.zoom;
11653     scaledWD(ds,oscale,nscale);
11654 }
11655 function WD_doscrollIN(){
11656     ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11657     oscale = WD_Zoom_1_XY.value;
11658     if( oscale == 0 || oscale == '' ){
11659         oscale = 1;
11660     }
11661     oscale = oscale * zoomMag();
11662     if( 4 < oscale ){
11663         alert('maybe too large, zoom=' + nscale);
11664     }
11665     ds.style.zoom = oscale;
11666     WD_Zoom_1_XY.value = ds.style.zoom;
11667     WD_Zoom_1_XY.value = ds.style.zoom;
11668     scaledWD(ds,oscale,nscale);
11669 }
11670 function WD_EventSetupP(){
11671     WD_Zoom_1_OUT.addEventListener('click', WD_doscrollIN);
11672     WD_Zoom_1_IN.addEventListener('click', WD_doscrollIN);
11673 }
11674
11675 function WD_doRSize(e){
11676     dt = VirtualDesktop_1;
11677     dt.style.width = WS_1_Width_1.value;
11678     dt.style.height = WD_Height_1.value;
11679     WD_Width_1.value = dt.style.width;
11680     WD_Height_1.value = dt.style.height;
11681 }
11682
11683 function WD_EventSetupP(){
11684     ds = VirtualDesktop_1_Content; // should be VirtualSpace_1
11685     ds.style.width = '5120px';
11686     ds.style.height = '2880px';
11687     WS_1_Width_1.value = ds.style.width;
11688     WS_1_Height_1.value = ds.style.height;
11689     WS_1_Width_1.addEventListener('keydown', ignoreEvent);
11690     WS_1_Width_1.addEventListener('keyup', ignoreEvent);
11691     WS_1_Height_1.addEventListener('keydown', ignoreEvent);
11692     WS_1_Height_1.addEventListener('keyup', ignoreEvent);
11693     WS_Resize_1.addEventListener('click', e => { WD_doRSize(e); });
11694 }
11695
11696 function WD_doscrollXY(e,sleft,stop){
11697     dt = VirtualDesktop_1;
11698     dt.scrollLeft = sleft;
11699     dt.scrollTop = stop;
11700     WD_Left_1.value = dt.scrollLeft;
11701     WD_Top_1.value = dt.scrollTop;
11702     console.log('--Scroll #' + dt.id + ' (' + sleft + ',' + stop + ')');
11703 }
11704
11705 function WD_doscroll(e){
11706     //dt = VirtualDesktop_1_Content;
11707     //dt = VirtualDesktop_1;
11708     //dt = VirtualDesktop_1_Content;
11709     //sleft = parseInt(WD_Left_1.value);
11710     //stop = parseInt(WD_Top_1.value);
11711     dt.scrollLeft = sleft;
11712     dt.scrollTop = stop;
11713     WD_Left_1.value = dt.scrollLeft;

```

```

11682     WD_Top_1.value = dt.scrollTop;
11683     console.log('--Scroll #' +dt.id+ ' ('+'sleft+', '+'stop+')');
11684   }
11685   function showScrollPosition(){
11686     if( false )
11687       console.log(
11688         'wstop=' + VirtualDesktop_1.style.top + ',' +
11689         'ws=' + VirtualDesktop_1.style.y + ',' +
11690         'was=' + VirtualDesktop_1.scrollTop + ',' +
11691         'wdtop=' + VirtualDesktop_1.Content.style.top + ',' +
11692         'wdx=' + VirtualDesktop_1.Content.style.y + ',' +
11693         'wds=' + VirtualDesktop_1_Content.scrollTop + ',' +
11694       );
11695     WD_Left_1.value = VirtualDesktop_1.scrollLeft;
11696     WD_Top_1.value = VirtualDesktop_1.scrollTop;
11697   }
11698   function WD_EventSetup7(){
11699     WD_Scroll_1.addEventListener('click',e => { WD_DoScroll(e); });
11700     WD_Left_1.addEventListener('keydown',ignoreEvent);
11701     WD_Top_1.addEventListener('keyup',ignoreEvent);
11702     WD_Top_1.addEventListener('keydown',ignoreEvent);
11703     WD_Top_1.addEventListener('keyup',ignoreEvent);
11704   }
11705   function WD_EventSetup8(){
11706     VirtualDesktop_1.addEventListener('scroll',showScrollPosition);
11707     VirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
11708   }
11709   if( false ){
11710     w = 1000 + 'px';
11711     dt.style.height = '300px';
11712     dt.style.resize = 'both';
11713     dt.style.borderWidth = 50 + 'px';
11714     dt.style.borderRadius = 25 + 'px';
11715     console.log('--2----- #' +dt.id+ ' style=' + dt.style);
11716     console.log('----- #' +dt.id+ ' width=' +dt.style.width);
11717     console.log('----- #' +dt.id+ ' lefts' + dt.style.left);
11718     console.log('----- #' +dt.id+ ' border=' +dt.style.border);
11719   }
11720   function onDTResize(e){
11721     dt = e.target;
11722     h = parseInt(dt.style.height);
11723     dt.style.borderWidth = (h * 0.075) + 'px';
11724     console.log('----- borderWidgh=' +dt.style.borderWidth);
11725   }
11726   VirtualDesktopDetails.addEventListener('toggle',VirtualDesktop_init);
11727   function VirtualDesktop_init(){
11728     if( !VirtualDesktopDetails.open ){
11729       return;
11730     }
11731     //GJ Join();
11732     VirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
11733     //console.log('----- #' +dt.id+
11734     // +' borderWidth=' +dt.style.getProperty('border-width'));
11735     settleWin(
11736       VirtualBrowser_1,
11737       VirtualBrowser_1_Location,
11738       VirtualBrowser_1_Command,
11739       VirtualBrowser_1_Frame,
11740       document.URL,
11741       500,280,50,20,'#262',1.0);
11742     settleWin(
11743       VirtualBrowser_2,
11744       VirtualBrowser_2_Location,
11745       VirtualBrowser_2_Command,
11746       VirtualBrowser_2_Frame,
11747       "https://its-more.jp/ja_jp/",
11748       500,280,150,100,'#448',1.0);
11749     settleWin(
11750       VirtualBrowser_3,
11751       VirtualBrowser_3_Location,
11752       VirtualBrowser_3_Command,
11753       VirtualBrowser_3_Frame,
11754       '/gshell/gsh.go.html',
11755       'https://golang.org',
11756       500,280,250,180,'#444',1.0);
11757       //1000,720,0,0,'#444',0.125);
11758       //1000,720,100,50,'#444',0.4);
11759     function WD_ClockUpdate(e){
11760       VirtualDesktop_1.Clock.innerHTML = DateShort();
11761       VirtualDesktop_1.Calender.innerHTML = DateHourMin();
11762     }
11763     window.setInterval(WD_ClockUpdate,500);
11764   }
11765   WD_EventSetup1();
11766   WD_EventSetup2();
11767   WD_EventSetup3();
11768   WD_EventSetup4();
11769   WD_EventSetup5();
11770   WD_EventSetup6();
11771   WD_EventSetup7();
11772   WD_EventSetup8();
11773   //VirtualDesktop_init();
11774   Destroy_VirtualDesktop = function(){
11775     VirtualDesktop_1.style = '';
11776     VirtualBrowser_1.removeAttribute('style');
11777     VirtualBrowser_1_Location.innerHTML = '';
11778     VirtualBrowser_1_Frame.removeAttribute('src');
11779     VirtualBrowser_1_Frame.removeAttribute('style');
11780     VirtualBrowser_1_Frame.style="";
11781     VirtualBrowser_2.removeAttribute('style');
11782     VirtualBrowser_2_Location.innerHTML = '';
11783     VirtualBrowser_2_Frame.removeAttribute('src');
11784     VirtualBrowser_2_Frame.style="";
11785     VirtualBrowser_3.removeAttribute('style');
11786     VirtualBrowser_3_Location.innerHTML = '';
11787     VirtualBrowser_3_Frame.removeAttribute('src');
11788     VirtualBrowser_3_Frame.style="";
11789     GJFactory_1.style = "";
11790     iframe0.style = "";
11791     VirtualDesktop_1.style = "";
11792   }
11793 }
11794
11795 </script>
11796 <!-- VirtualDesktop -->
11797 <details>
11798 </> //</span>
11799
11800 //<-- ===== Work { ===== -->
11801 <span id="SBSidebar_WorkCodeSpan">
11802 <!-->
11803 <details><summary>SBSidebar // 2020-0928 SatoxITS { -->
11804 <h2>SBSidebar</h2>
11805 <input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11806 <input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11807 <input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11808 <span id="SBSidebar_WorkCodeView"></span>
11809 <script id="SBSidebar_WorkScript">
11810   function SBSidebar_openWorkCodeView(){
11811     SBSidebar_showWorkCode();
11812   }
11813   function SBSidebar_openWorkCodeView(){
11814     SBSidebar_showWorkCode();
11815   }
11816   SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
11817 }
11818 SBSidebar_openWorkCodeView(); // should be invoked by an event
11819
11820 console.log('-- Sbslider // 2020-1006-01 SatoxITS --');
11821 function SetSidebar(){
11822   sidebar = document.getElementById('secondary');
11823   sidebar.parentElement.getPrimaryElement('secondary');
11824   // console.log('primary=' +primary+ ' secondary=' +secondary+ ' main=' +main+ ' ');
11825   wrap = sidebar.parentNode;
11826   wrap = sidebar.parentNode;
11827   wrap = sidebar.parentNode;
11828   wrap = sidebar.parentNode;
11829   wrap = sidebar.parentNode;
11830   wrap = sidebar.parentNode;
11831   wrap = sidebar.parentNode;
11832   wrap = sidebar.parentNode;
11833   wrap = sidebar.parentNode;
11834   wrap = sidebar.parentNode;
11835   wrap = wrap.parentNode;
11836   wrap = wrap.parentNode;
11837   wrap = wrap.parentNode;
11838   wrap = wrap.parentNode;
11839   // console.log('-- ShSlider parent is ' +wrap+ ', #' +wrap.id+ '.' +wrap.class);
11840   nsb = sidebar.cloneNode();
11841   nsb = sidebar;
11842   nsb.style.width = '100%';
11843   sidebar.appendChild(nsb);
11844   slider.id = 'ShSlider';
11845   slider.setAttribute('class','SbSlider');
11846   slider.style.position = 'relative';
11847   slider.style.position = 'fixed';
11848   slider.style.zIndex = '1000'; // inline';
11849   slider.style.zIndex = 100000;
11850   // nsb.style.zIndex = 200000;
11851   nsb.style.position = 'absolute';
11852   nsb.style.minWidth = '80px';
11853   nsb.style.left = '0px';
11854   nsb.style.top = '0px';
11855   w = window.innerWidth;
11856   console.log('SliderWidth '+w+: '(w/3)*'px');
11857   if( w < 640 ){
```

```

11859     slider.style.setProperty('width',(w/3) + 'px','important');
11860   }
11861   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
11862   slider.style.resize = "both";
11863   slider.draggable = "true";
11864   wrap.appendChild(slider);
11865   console.log('-- added SbSlider');
11866   /*nsb.addEventListener('scroll',SbScrolled);
11867
11868   buttons = document.createElement('div');
11869   buttons.id = 'NaviButtons';
11870   buttons.setAttribute('class','NaviButtons');
11871   buttons.innerHTML = '<+' + p +>a href="#TopOfPost" draggable="true">TOP<+' + a +></' + p +>';
11872   buttons.innerHTML += '<+' + p +>a href="#EndOfPost" draggable="true">END<+' + p +>';
11873   page.appendChild(buttons);
11874   buttons.style.position = 'fixed';
11875   buttons.style.zIndex = 3000;
11876   buttons.style.left = '100px';
11877   buttons.style.top = '320px';
11878   buttons.style.left = parseInt(w) * 1.0 + 'px';
11879   console.log('-- SbSlider installed (^~^) / SatoxITS');
11880 }
11881 //window.addEventListener('load',SetSidebar); // after load
11882 DestroyNaviButtons = function(){
11883   nb = document.getElementById('NaviButtons');
11884   if( nb != null ){
11885     nb.parentNode.removeChild(NaviButtons);
11886   }
11887 }
11888 </script>
11889 // 2020-100 its-more.jp-blog-60000-style.css
11890 <!-- {
11891   <style>
11892     #NaviButtons {
11893       position:fixed;
11894       display:block;
11895       width:100px;
11896       xtop:100px;
11897       xzIndex:3000;
11898       zIndex:3000;
11899       font-size:10pt;
11900       color:#2ff !important;
11901       text-align:center;
11902       background-color:rgba(230,230,230,0.01);
11903     }
11904     #NaviButtons a {
11905       color:#2a2 !important;
11906       font-size:20px;
11907       text-align:center;
11908       xtext-decoration:2px 2px #8ff;
11909       padding:0px;
11910       margin:10px;
11911       border:1px solid #288 !important;
11912       border-radius:3px;
11913       background-color:rgba(160,160,160,0.05);
11914     }
11915     #SbSlider {
11916       overflow:auto;
11917       resize:both !important;
11918       xxoverflow-y:hidden !important;
11919       height:100px !important;
11920       display:inline !important;
11921       position:fixed !important;
11922       left:0px;
11923       top:0px;
11924       xxz-index:60000;
11925       scroll-behavior: overflow !important;
11926       xxbackground:xxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11927       padding-left:4pt;
11928       color:#fff;
11929       font-size:0.5em;
11930       background-color:rgba(64,160,64,0.6) !important;
11931       white-space:nowrap;
11932     }
11933     #secondary {
11934       position:fixed;
11935       left:0px;
11936       top:0px;
11937       xxxx-index:60000;
11938       scroll-behavior: overflow !important;
11939       xxbackground:xxxxxxxxxxxxxxxxxxxxwidth:18% !important;
11940       padding-left:4pt;
11941       color:#fff;
11942       font-size:0.5em;
11943       background-color:rgba(64,160,64,0.6) !important;
11944       white-space:nowrap;
11945     }
11946     #secondary a {
11947       color:#fff !important;
11948       text-decoration:disabled !important;
11949     }
11950   }
11951   #primary {
11952     position:relative;
11953     width:75% !important;
11954     left:25% !important;
11955   }
11956   #main {
11957     position:relative;
11958     width:75% !important;
11959     left:25% !important;
11960   }
11961   #site-navigation {
11962     position:relative;
11963     left:120px;
11964   }
11965   #dswc_countertext {
11966     color:#4169el;
11967     font-size:16pt !important;
11968     xxfont-size:10px !important;
11969     font-weight:bold;
11970   }
11971   #nowline {
11972     color:#0ffa0;
11973     font-size:16pt !important;
11974     xxfont-size:10px !important;
11975     font-weight:bold;
11976     text-shadow:1px 1px #fff;
11977   }
11978   .navigation-top {
11979     color:#2fa !important;
11980     border:0px;
11981     background-color:rgba(220,220,220,0.1);
11982   }
11983   .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11984     display: block;
11985     xxwidth: 1em;
11986     xxoverflow: auto;
11987     xxheight: 1em;
11988   }
11989   .invisible-scrollbar ::-webkit-scrollbar {
11990     xxdisplay: none;
11991     width:1px !important;
11992     height:1px !important;
11993     height:1px !important;
11994   }
11995   .mostly-customized-scrollbar ::-webkit-scrollbar {
11996     width: 2px;
11997     height: 1px;
11998     xxbackground-color: #aaa; xxor add it to the track;
11999   }
12000   .mostly-customized-scrollbar ::-webkit-scrollbar-thumb {
12001     background: #000;
12002   }
12003 </style>
12004 -->
12005
12006 <details>
12007   <!-- SbSlider_WorkCodeSpan -->
12008   <!-- //</span> -->
12009   <!-- ===== Work -->
12010  <!-- ===== Work -->
12011
12012  <!-- ===== Work { ===== -->
12013  <!-- Affiliate_WorkCodeSpan >
12014  <details id="Affiliate_Test"><summary>Affiliate</summary>
12015  <!-- ===== Affiliate // 2020-1010 SatoxITS { -->
12016  <div id="AffViewDoc" style="display:none;">
12017    <div id="AfvView" class="AfvView" draggable="true" style="">
12018      <div id="AfvSet" class="AfvPlate">
12019        <iframe id="aff_0" class="AfvItem"></iframe>
12020        <iframe id="aff_1" class="AfvItem"></iframe>
12021        <iframe id="aff_2" class="AfvItem"></iframe>
12022        <iframe id="aff_3" class="AfvItem"></iframe>
12023        <iframe id="aff_4" class="AfvItem"></iframe>
12024        <iframe id="aff_5" class="AfvItem"></iframe>
12025      </div>
12026    </div>
12027  </div>></div>
12028  <h2>Supportive Affiliate</h2>
12029  <div>
12030    <AfvView {
12031      z-index:0;
12032      overflow-x:scroll;
12033      overflow-y:scroll;
12034      position:fixed;
12035    }

```

```

12036 max-width:2560px;
12037 max-height:100%;
12038 width:100px;
12039 left:75px;
12040 height:95%;
12041 resize:both;
12042 xleft:-10px;
12043 margin-left:140px;
12044 xleft:0px;
12045 xxalign:right;
12046 display:block;
12047 border:4px inset rgba(255,255,255,0.1);
12048 background-color:rgba(255,255,255,0.1);
12049 }
12050 .AffView:hover {
12051 z-index:1;
12052 width:300px;
12053 overflow-y:scroll;
12054 border:4px solid #fcc;
12055 background-color:rgba(80,80,255,0.2);
12056 background-color:#fff;
12057 }
12058 .Affplate:hover {
12059 border-left:px dashed #888;
12060 }
12061 .Affplate{
12062 overflow-x:visible;
12063 border-left:4px dashed rgba(255,255,255,0.1);
12064 max-width:2560px;
12065 max-height:2880px;
12066 margin-bottom:10px;
12067 margin-left:4px;
12068 width:300px;
12069 xheight:1440px;
12070 }
12071 .Affitem {
12072 overflow-x:visible;
12073 xoverflow-y:scroll;
12074 max-width:2560px;
12075 max-height:1440px;
12076 z-index:1;
12077 display:block;
12078 xposition:fixed;
12079 xposition:absolute;
12080 position:relative;
12081 //left:300px;
12082 xmargin-left:0px;
12083 padding:0px;
12084 width:600px;
12085 height:400px;
12086 max-height:800px !important;
12087 margin-left:0px;
12088 margin-left:0px;
12089 margin-right:0px !important;
12090 border:16px inset #ccc;
12091 transform:scale(0.5);
12092 background-color:rgba(255,255,255,0.2);
12093 xxalign:right;
12094 }
12095 .Affitem:hover {
12096 border:16px inset #bbf;
12097 }
12098 <style>
12099 <input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12100 <input id="Affiliate_WorkCodeViewSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12101 <input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12102 <span id="Affiliate_WorkCodeView"></span>
12103 <script id="Affiliate_WorkScript">
12104 function Affiliate_OpenWorkCodeView(){
12105   function Affiliate_showWorkCode(){
12106     showHTMLCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
12107   }
12108   Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
12109 }
12110 Affiliate_openWorkCodeView(); // should be invoked by an event
12111
12112 ///<iframe id="aff_8" xsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
12113 ///<iframe id="aff_9" xsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
12114
12115 var Aff_isSetup = false;
12116 Affiliate_Reset.addEventListener('click',Aff_Setup);
12117 function Aff_Setup(){
12118   if (Aff_isSetup){ return; } Aff_isSetup = true;
12119   parent = document.documentElement;
12120   parent.appendChild(AffView);
12121   AffView.style.top = '0px';
12122   AffView.style.left = (window.innerWidth - 280) + 'px';
12123   AffView.style.width = '280px';
12124   var off = 100;
12125   zoom = 0.5;
12126   ozoom = 0.3;
12127   leftx = window.innerWidth - 300;
12128   leftx += leftx + 'px';
12129   left = '-130 + 'px';
12130   console.log('aff-init window.innerWidth=' + window.innerWidth);
12131   w = 1000;
12132   h = 560;
12133
12134   aff_0.src = './gshell/gsh.go.html';
12135   aff_1.src = 'https://golang.org';
12136   aff_2.src = 'https://drafts.csswg.org/';
12137   aff_3.src = 'https://html.spec.whatwg.org/dev/';
12138   aff_4.src = 'https://wikispec.org';
12139   aff_5.src = 'https://www.bing.com/translator';
12140
12141   /parent.appendChild(aff_0);
12142   aff_0.style.width = zoom*w+'px';
12143   aff_0.style.height = zoom*h+'px';
12144   aff_0.style.left = left;
12145   //aff_0.style.top = off+'px'; off += ozoom*h;
12146   aff_0.draggable = 'true';
12147
12148   /parent.appendChild(aff_1);
12149   aff_1.style.width = zoom*w+'px';
12150   aff_1.style.height = zoom*h+'px';
12151   aff_1.style.left = left;
12152   //aff_1.style.top = off+'px'; off += ozoom*h;
12153   aff_1.style.top = '-150px';
12154   aff_1.draggable = 'true';
12155
12156   /parent.appendChild(aff_2);
12157   aff_2.style.width = zoom*w+'px';
12158   aff_2.style.height = zoom*h+'px';
12159   aff_2.style.left = left;
12160   //aff_2.style.top = off+'px'; off += ozoom*h;
12161   aff_2.style.top = '-300px';
12162   aff_2.draggable = 'true';
12163
12164   /parent.appendChild(aff_3);
12165   aff_3.style.transform = 'scale(0.25)';
12166   aff_3.style.width = 2*zoom*w+'px';
12167   aff_3.style.height = 2*zoom*h+'px';
12168   aff_3.style.border = '25px inset #ccc';
12169   //aff_3.style.left = '-390 + 'px; //left+2;
12170   //aff_3.style.left = (leftx - 265) + 'px';
12171   aff_3.style.left = '-395 + 'px';
12172   //aff_3.style.top = (-155+off)+ 'px'; off += ozoom*h;
12173   aff_3.style.top = '-600px';
12174   aff_3.draggable = 'true';
12175
12176   /parent.appendChild(aff_4);
12177   aff_4.style.width = zoom*w+'px';
12178   aff_4.style.height = zoom*h+'px';
12179   aff_4.style.left = left;
12180   //aff_4.style.top = off+'px'; off += ozoom*h;
12181   aff_4.style.top = '-900px';
12182   aff_4.draggable = 'true';
12183
12184   /parent.appendChild(aff_5);
12185   aff_5.style.transform = 'scale(0.300)';
12186   aff_5.style.width = zoom*(w*1.67) +'px';
12187   aff_5.style.height = zoom*(h*1.67) +'px';
12188   aff_5.style.border = '25px inset #ccc';
12189   aff_5.style.left = '-308px';
12190   //aff_5.style.left = (leftx - 155+leftx) + 'px';
12191   aff_5.style.left = '0px';
12192   //aff_5.style.top = '0px';
12193   aff_5.style.top = '-1150px';
12194   //aff_5.style.align = 'right';
12195   aff_5.draggable = 'true';
12196
12197   window.addEventListener('resize',affresize);
12198
12199 function affresize(){
12200   AffView.style.left = (window.innerWidth - 280) + 'px';
12201   AffView.style.width = window.innerWidth - 400;
12202   left = leftx + 'px';
12203   console.log('aff-resize window.innerWidth=' + window.innerWidth);
12204
12205   //window.addEventListener('resize',affresize);
12206   //document.addEventListener('resize',affresize);
12207   //gsh.addEventListener('resize',affresize);
12208
12209
12210 function ResetAffView(){
12211   AffViewDock.appendChild(AffView);
12212   AffView.removeAttribute('style');

```

```

12213 aff_0.removeAttribute('src');
12214 aff_1.removeAttribute('style'); aff_0.removeAttribute('draggable');
12215 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
12216 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
12217 aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
12218 aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
12219 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12220 aff_6.removeAttribute('style'); aff_6.removeAttribute('draggable');
12221 aff_7.removeAttribute('style'); aff_7.removeAttribute('draggable');
12222 aff_8.removeAttribute('style'); aff_8.removeAttribute('draggable');
12223 aff_9.removeAttribute('style'); aff_9.removeAttribute('draggable');
12224 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12225
12226</script>
12227</details>
12228<!-- Affiliate_WorkCodeSpan } -->
12229/* //</span>
12230//<!-- ===== Work } ===== -->
12231
12232
12233
12234//<!-- ===== Work { ===== -->
12235<span id="TextCanvas_WorkCodeSpan">
12236<!--
12237<details id="TextCanvas_Section"><summary id="TextCanvas_Summary">TextCanvas</summary>
12238<!-- ----- Textcanvas // 2020-1013 SatoxITS { -->
12239<details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selection</summary>
12240<h2>Font Selection</h2>
12241<div id="FontList"></div>
12242</div>
12243<style>
12244#FontList {
12245    overflow:visible;
12246    background-color:rgba(240,245,255,1.0) !important;
12247}
12248#FontList td {
12249    font-size:16px;
12250    padding:0px;
12251    padding-left:2px;
12252    padding-right:2px;
12253    margin:0px;
12254    line-height:1.2;
12255    border:0px;
12256}
12257#FontList td:hover {
12258    background-color:#228;
12259}
12260#FontList tr:hover {
12261    color:ffff;
12262    background-color:#000;
12263    xborder:1px solid #000;
12264}
12265.xcourier { colr:#000; font-size:16px; font-family:courier; }
12266.xcursive { colr:#000; font-size:16px; font-family:cursive; }
12267.xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
12268.xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
12269.xmonospace { colr:#000; font-size:16px; font-family:monospace; }
12270</style>
12271<script>
12272function fontstr(name,text){
12273    /tr = '<'+tr style="font-family:" +name+";\n';
12274    tr = '<'+tr style="font-family:" +name+";\n';
12275    tr += '<'+td style="font-size:12pt;">' +name+ '</' +td +';
12276    tr += '<'+td data-fsty="n">' +text + '</' +td +';
12277    tr += '<'+td data-fsty="b">' +text + '<+' +td +';
12278    tr += '<'+td data-fsty="i">' +text + '<+' +td +';
12279    tr += '<'+td data-fsty="bi">' +text + '<+' +td +';
12280    tr += '<'+td data-fsty="i">' +text + '</' +td +';
12281    tr += '<'+td data-fsty="b">' +text + '</' +td +';
12282    tr += '<'+td data-fsty="bi">' +text + '</' +td +';
12283    return tr;
12284}
12285function lsfont(){
12286    text = 'GShell-Go012';
12287
12288    fl = '';
12289    fl += '<table>\n';
12290    fl += fontstr('Arial',text);
12291    fl += fontstr('Courier',text);
12292    fl += fontstr('Courier New',text);
12293    fl += fontstr('Georgia',text);
12294    fl += fontstr('Gelvetica',text);
12295    fl += fontstr('Verdana',text);
12296    fl += fontstr('Times',text);
12297
12298    fl += fontstr('Osaka',text);
12299    fl += fontstr('Meiryo',text);
12300    fl += fontstr('YuMincho',text);
12301
12302    //fl += fontstr('Roman',text);
12303    //document.fonts.load("30px cursive");
12304    fl += fontstr('Serif',text);
12305    fl += fontstr('System-Serif',text);
12306    fl += fontstr('System-UI',text);
12307    fl += fontstr('Monospace',text);
12308    fl += fontstr('Cursive',text);
12309    fl += fontstr('Fantasy',text);
12310
12311    fl += '</table>\n';
12312
12313    if( false ){
12314        fss = document.fonts.entries(); // FontFaceSet
12315        console.log('FSS=' +fss);
12316        while( true ){
12317            font = fss.next();
12318            if( font.done ) {
12319                break;
12320            }
12321            fl += font.value[0] + '<br>';
12322        }
12323        FontList.innerHTML = fl;
12324    }
12325    function selectFont(e){
12326        t = e.target;
12327        let fsty = '';
12328        for( i=0; i< 4; i++ ){
12329            //console.log('FontSelect '+t.nodeName+ '#'+t.id+ ' '+t.style);
12330            if( t.nodeName == 'TD' ){
12331                //console.log('FontSelect ' +t.outerHTML);
12332                if( t.hasAttribute('data-fsty') ){
12333                    fsty = t.getAttribute('data-fsty');
12334                    //console.log('FontSelect = ' + fsty);
12335                }
12336            }
12337            if( t.nodeName != 'TD' ){
12338                if( t.style == '' ){
12339                    if( t.style.fontFamily != '' ){
12340                        break;
12341                    }
12342                }
12343            }
12344            t = t.parentNode;
12345        }
12346        if( t.style != '' ){
12347            font = t.style.fontFamily;
12348            //console.log('FontSelect: '+font);
12349            //console.log('FontSelect == ' + fsty);
12350            if( fsty != font ){
12351                sel = document.getElementById("TextCanvas_1_Font");
12352                if( sel != null ){
12353                    if( fsty != '' ){
12354                        TextCanvas_1_Bold.checked = 0 <= fsty.indexOf('b');
12355                        TextCanvas_1_Italic.checked = 0 <= fsty.indexOf('i');
12356                    }
12357                    sel.value = font;
12358                    RedrawTextCanvas();
12359                }else{
12360                    alert('Event: ' + e.target.nodeName + ' #' + font);
12361                }
12362            }
12363        }
12364    }
12365    FontList.addEventListener('click',selectFont);
12366    document.fonts.onloadingdone = function(fse){
12367        //alert('font-loaded ' +fse.fontfaces.length);
12368    }
12369    function FontList_Setup(){
12370        if( FontSelect_Summary.open ){
12371            lsfont();
12372        }
12373    }
12374    FontSelect_Summary.addEventListener('click',lsfont);
12375</script>
12376</details>
12377
12378<h2>Drawing Text on Canvas</h2>
12379<!-- 2020-1012 -- Drawing Text on Canvas // SatoxITS { -->
12380<div id="TextCanvas_1_Panel" class="CanvasLabel">
12381<input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
12382<input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
12383<input id="TextCanvas_1_Paste" class="PanelButton" type="button" value="Paste">
12384<input id="TextCanvas_1_Bold" class="CanvasPanel" type="checkbox" checked="checked" value="Bold" checked="">>Bold
12385<input id="TextCanvas_1_Italic" class="CanvasPanel" type="checkbox" checked="checked" value="Italic" checked="">>Italic
12386<br>
12387<input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12388<input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">

```

```

12390<!-- to be PBlue series ? -->
12391<p>
12392<input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
12393</p>
12394</div>
12395<canvas id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></canvas>
12396</div>
12397<div class="CanvasLabel">
12398<input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
12399<input id="TextCanvas_1_ToPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
12400<input id="TextCanvas_1_ToJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
12401<input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
12402<img id="TextCanvas_1_Image" class="CanvasImage" src=""/>(inline image)</div>
12403<div id="TextCanvas_1_BgImage" class="CanvasImage"><br>(background image)</div>
12404<div (Data URL)>
12405<div id="TextCanvas_1_DataUrlView" class="DataUrlView"><span id="TextCanvas_1_DataUrlText"></span></div>
12406</div>
12407<style>
12408 .CommandUsageText {
12409   font-family:Courier New;
12410 }
12411 .TextCanvas {
12412   border:1px solid #000;
12413   resize:both;
12414   display:inline important;
12415 }
12416 .DataUrlView {
12417   width:100%;
12418   font-size:10pt;
12419   font-family:Courier New, monospace;
12420   color:#000;
12421   background-color:#eee;
12422   margin-bottom:10px;
12423   xdisplay:block;
12424   xoverflow:scroll;
12425 }
12426 .CanvasImage {
12427   border:1px dashed #000;
12428 }
12429 .TextCanvasText {
12430   font-size:12pt;
12431   width:100%;
12432 }
12433 .CanvasLabel {
12434   font-size:10pt;
12435   color:#000;
12436 }
12437 .CanvasImage {
12438   width:100%;
12439   height:160px;
12440   margin-bottom:10px;
12441   color:#000;
12442   text-shadow:3px 3px 2px #eee;
12443   background-color:#eee;
12444   xborder:1px solid #000;
12445   font-size:18pt;
12446   vertical-align:middle;
12447 }
12448 .PanelRadio {
12449   font-size:2pt !important;
12450   color:#000 !important;
12451   vertical-align:middle;
12452 }
12453 .CanvasBox {
12454   vertical-align:middle;
12455   margin-left:4px !important;
12456   margin-right:2px !important;
12457 }
12458 .CanvasPanel {
12459   vertical-align:middle !important;
12460   height:14pt !important;
12461   width:inherit !important;
12462   padding:2px !important;
12463   margin:4px !important;
12464   margin-left:2px !important;
12465   margin-right:2px !important;
12466   font-size:10pt !important;
12467   font-family:Georgia !important;
12468   color:#000;
12469   display:inline important;
12470 }
12471 .TextCanvasPanel {
12472   vertical-align:middle;
12473   font-size:10pt !important;
12474   font-family:Georgia !important;
12475   color:#000;
12476   display:inline important;
12477 }
12478 .PanelButton {
12479   font-size:10pt !important;
12480   font-family:Georgia !important;
12481   vertical-align:middle !important;
12482   width:5pt !important;
12483   height:14pt !important;
12484   line-height:1.2 !important;
12485   padding:2px !important;
12486   margin:1px !important;
12487   border:1px solid #000 !important;
12488   padding:1px !important;
12489   color:#fff !important;
12490   background-color:#228 !important;
12491 }
12492
12493 </style>
12494 <script>
12495 function DrawTextCanvas(){
12496   ctx = TextCanvas_1.getContext('2d');
12497   var textfont = '7';
12498   if( TextCanvas_1_Iitalic.checked ) textfont += ' italic';
12499   if( TextCanvas_1_Bold.checked ) textfont += ' bold';
12500   textfont += 'TextCanvas_1_Font.value'+'px';
12501   textfont += 'TextCanvas_1_Size.value'+'px';
12502   //ctx.font = 'italic bold 64px Georgia';
12503   //console.log('TxFont'+textfont);
12504   ctx.fillStyle = TextCanvas_1_Color.value; // '#22a';
12505   ctx.font = textfont;
12506   ctx.fillText(textfont,TextCanvas_1_Text.value,10,80);
12507 }
12508 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12509 function ClearTextCanvas(){
12510   cv = TextCanvas_1;
12511   cv = cv.getContext('2d');
12512   cv.clearRect(0,0,cv.width,cv.height);
12513 }
12514 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12515 function RedrawTextCanvas(){
12516   clearTextCanvas();
12517   DrawTextCanvas();
12518 }
12519
12520 function ab2str(buf) {
12521   return String.fromCharCode.apply(null, new Uint16Array(buf));
12522 }
12523
12524 // 2020-1024, canvas to image
12525 function CanvasToImage(){
12526   canvas = TextCanvas_1;
12527   // https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
12528   if( TextCanvas_1_ToImage.checked ){
12529     url = canvas.toDataURL("image/png");
12530   }else{
12531     url = canvas.toDataURL("image/jpeg",1.0);
12532   }
12533   //alert('CanvasToImage: length=' +url.length+'\n'+url);
12534   TextCanvas_1_Image.src = url;
12535   TextCanvas_1_BgImage.style.backgroundImage = 'url('+url+')';
12536   if( TextCanvas_1_DataURL.checked ){
12537     txa = TextCanvas_1_DataUrlView.innerHTML = url;
12538     utxa = document.createElement('textarea');
12539     utxa.id = 'TextCanvas_1_DataUrlText';
12540     utxa.style.width = '100%';
12541     utxa.style.height = '50pt';
12542     utxa.value = url;
12543     txa.parentNode.replaceChild(utxa,txa);
12544   }
12545   return TextCanvas_1_Image;
12546 }
12547 var image = new Image();
12548 image.src = url;
12549 url = 'blob:' +url;
12550 blob = new Blob(urla,{type:'text/plain'});
12551 link = document.createElement('a');
12552 link.href = URL.createObjectURL(blob);
12553 link.download = 'character.txt';
12554 link.click();
12555 return image;
12556 }
12557
12558 TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12559
12560 if( TextCanvas_Section.open ){
12561   DrawTextCanvas();
12562 }
12563 TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
12564 </script>
12565 <!-- } -->
12566

```

```

12567<script>
12568</TextCanvas_1_Panel.addEventListener('mousein',OffGJShell);
12569</TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
12570</script>
12571<!-- Clicking the textures to see upto the end of text. why? -->
12572<input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12573<input id="TextCanvas_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12574<input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12575<script id="TextCanvas_WorkScript">
12576 function TextCanvas_openWorkCodeView(){
12577     function TextCanvas_showWorkCode(){
12578         showHTMLCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
12579     }
12580     TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
12581 }
12582</TextCanvas_openWorkCodeView(); // should be invoked by an event
12583</script>
12584</details>
12585<span id="TextCanvas_WorkCodeSpan" >-->
12586</span> //</span>
12587//<!-- ===== Work } ===== -->
12588
12589
12590
12591<!-- ===== Work { ===== -->
12592<span id="Shading_WorkCodeSpan">
12593<!--
12594<details><summary>Shading Canvas</summary>
12595<!-- ----- Shading Canvas // 2020-10-11 SatoxITS { -->
12596<h2>Shading Canvas</h2>
12597<div id="Shading_1_Html" style="background-color:#fff; border:1px solid #000; padding:10px; font-family:Courier New; font-size:12pt; height:400px; width:100%;">
12598<b>Commands</b><br>
12599<b>Placement Mode</b><br>
12600<input type="checkbox" checked="" value="absolute" /> ... apply (into absolute position)<br>
12601<input type="checkbox" checked="" value="relative" /> ... bring down (ArrowDown)<br>
12602<input type="checkbox" checked="" value="absolute" /> ... bring up (ArrowUp)<br>
12603<input type="checkbox" checked="" value="absolute" /> ... bring left (ArrowLeft)<br>
12604<input type="checkbox" checked="" value="absolute" /> ... bring right (ArrowRight)<br>
12605<input type="checkbox" checked="" value="absolute" /> ... z-index = 0<br>
12606<input type="checkbox" checked="" value="absolute" /> ... z-index += 1<br>
12607<input type="checkbox" checked="" value="absolute" /> ... z-index -= 1<br>
12608<input type="checkbox" checked="" value="absolute" /> ... return to here (relative position)<br>
12609<input type="checkbox" checked="" value="absolute" /> ... clear the log text<br>
12610<input type="checkbox" checked="" value="absolute" /> Note: the HTM text must be contenteditable to catch Key Event.<br>
12611</note>
12612
12613<div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
12614<div id="Shading_1_Html" class="ShadingHTML" draggable="true"></div>
12615<div id="Shading_1_Log" class="ShadingLog" draggable="true" style="background-color:#fff; border:1px dashed #000; height:40px; width:100%; font-family:Courier New; font-size:12pt; margin-top:10px; margin-bottom:10px; padding:5px; white-space:pre; overflow:scroll; background-color:rgba(200,255,200,0.4); height:400px; width:100%;">
12616<style>
12617 .ShadingPlate {
12618     z-index:0;
12619     position:static;
12620     overflow:scroll;
12621     display:block;
12622     width:100px;
12623     height:40px;
12624     font-size:9pt;
12625     font-family:Courier New;
12626     border:1px dashed #000;
12627     color:#444;
12628 }
12629 .ShadingLog {
12630     z-index:0;
12631     position:relative;
12632     display:block;
12633     width:100px;
12634     left:0px;
12635     overflow:scroll;
12636     width:100px;
12637     font-size:9pt;
12638     font-family:Courier New;
12639     color:#666;
12640     font-size:9pt;
12641     font-family:Courier New;
12642     color:#666;
12643     overflow:scroll;
12644     background-color:rgba(200,255,200,0.4);
12645     height:400px;
12646 }
12647 .ShadingHTML {
12648     z-index:2;
12649     position:relative;
12650     display:block;
12651     top:0px;
12652     left:0px;
12653     overflow:scroll;
12654     width:100px;
12655     font-size:12pt;
12656     font-family:Courier New;
12657     color:#666;
12658     overflow:scroll;
12659     background-color:rgba(200,255,200,0.4);
12660     height:400px;
12661 }
12662 .ShadingCanvas {
12663     z-index:3;
12664     position:relative;
12665     width:100px;
12666     height:100px;
12667     left:100px;
12668     resize:both;
12669     border:1px solid #000;
12670 }
12671</style>
12672<input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12673<input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12674<input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12675<span id="Shading_WorkScript" >-->
12676</script>
12677<function Shading_openWorkCodeView(){>
12678     function Shading_showWorkCode(){
12679         showHTMLCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
12680     }
12681     Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
12682 }
12683</function>
12684 const BR = '<'+<br>;
12685 Shading_shownWorkCode(); // should be invoked by an event
12686 function sh_onClick(e){
12687     Shading_1_Log.innerHTML += ' Click '+e.target.nodeName+'#'+e.target.id
12688     + ' at '+e.offsetX+' '+e.offsetY+' ';
12689     + ' client('+e.clientX+', '+e.clientY+')';
12690     + ' page('+e.pageX+', '+e.pageY+')';
12691     + ' screen('+e.screenX+', '+e.screenY+')'
12692     +BR;
12693     e.stopPropagation();
12694     e.preventDefault();
12695 }
12696 function sh_onKeyup(e){
12697     if( Shading_1.style.zIndex == '' ){
12698         Shading_1.style.zIndex = 0;
12699     }
12700     zi = parseInt(Shading_1.style.zIndex);
12701     if( e.key.length == 1 ){
12702         Shading_1_Html.innerHTML += e.key;
12703     }
12704     if( e.key == '0' ) { zi = 0; } else
12705     if( e.key == '+' ) { zi += 1; } else
12706     if( e.key == '-' ) { zi -= 1; } else
12707     if( e.key == 'c' ){
12708         Shading_1_Log.innerHTML = '';
12709     }
12710     if( e.key == 'r' ){
12711         Shading_1.style.position = "relative";
12712         Shading_1.style.top = '0px';
12713         Shading_1.style.left = '0px';
12714         zi = 0;
12715     }
12716     if( e.key == 'j' || e.code == 'ArrowDown' ){
12717         topx = parseInt(Shading_1.style.top) + 50;
12718         Shading_1.style.top = topx + 'px';
12719     }
12720     if( e.key == 'k' || e.code == 'ArrowUp' ){
12721         topx = parseInt(Shading_1.style.top) - 50;
12722         Shading_1.style.top = topx + 'px';
12723     }
12724     if( e.key == 'l' || e.code == 'ArrowRight' ){
12725         lefty = parseInt(Shading_1.style.left) + 50;
12726         Shading_1.style.left = lefty + 'px';
12727     }
12728     if( e.key == 'h' || e.code == 'ArrowLeft' ){
12729         lefty = parseInt(Shading_1.style.left) - 50;
12730         Shading_1.style.left = lefty + 'px';
12731     }
12732     if( e.key == 'a' ){
12733         Shading_1.style.position = "absolute";
12734         Shading_1.style.top = '0px';
12735         Shading_1.style.left = '0px';
12736     }
12737     else{
12738     }
12739     Shading_1.style.zIndex = zi;
12740     Shading_1_Log.innerHTML += 'Keyup.'+e.target.nodeName+'#'+e.target.id
12741     + 'Up'+e.key+'/'+e.code+
12742     + 'z-index:'+zi+'/'+Shading_1.style.Index
12743     + 'top:' + Shading_1.style.top
12744

```

```

12744     +BR;
12745     e.stopPropagation();
12746     e.preventDefault();
12747 }
12748 function sh_onKeyDown(e){
12749     Shading_1_Log.innerHTML += 'keydown'+e.target.nodeName+'#'+e.target.id
12750     +' Down'+e.keyCode+'+'+e.code+')'+BR;
12751     e.stopPropagation();
12752     e.preventDefault();
12753 }
12754
12755 function Shading_Setup(){
12756     Shading_1_Log.style.top = "-400px";
12757     Shading_1_Log.style.left = "200px";
12758 }
12759
12760 Shading_1.append(sh_onKeyDown);
12761 Shading_1.addEventListener('keydown',sh_onKeyDown);
12762 Shading_1.addEventListener('keyup',sh_onKeyUp);
12763 Shading_1.addEventListener('click',sh_onClick);
12764 Shading_1.addEventListener('click',sh_onClick);
12765
12766 Shading_1_Log.style.top = "-400px";
12767 Shading_1_Log.style.left = "200px";
12768
12769 Shading_1.appendChild(Shading_1_Canvas);
12770 Shading_1_Canvas.style.width = "300px";
12771 Shading_1_Canvas.style.height = "300px";
12772 Shading_1_Canvas.style.position = "relative";
12773 Shading_1_Canvas.style.top = "-750px";
12774 Shading_1_Canvas.style.left = "100px";
12775
12776 const ctxt = Shading_1_Canvas.getContext('2d');
12777 ctxt.fillStyle = 'rgba(160,0,0,0.9)';
12778 ctxt.fillRect(50,50,40,40);
12779 ctxt.fillStyle = 'rgba(160,0,0,0.9)';
12780 ctxt.fillRect(60,60,40,40);
12781 ctxt.fillStyle = 'rgba(0,0,160,0.9)';
12782 ctxt.fillRect(70,70,40,40);
12783
12784
12785 function Reset_ShadingCanvas(){
12786     Shading_1_Log.removeAttribute('style');
12787     Shading_1_Log.innerHTML = '';
12788     Shading_1_Canvas.style = "";
12789     //Shading_1_Canvas.removeAttribute('style');
12790 }
12791
12792 </-- ===== Work ===== -->
12793 <!-- Shading_WorkCodeSpan -->
12794 <!-- </span>
12795 //<!-- ===== Work } ===== -->
12796
12797 </-- ===== Work ===== -->
12798 <span id="Charmap_WorkCodeSpan">
12799 <!-- <span>
12800 //<-- ===== Work } ===== -->
12801 <div id="Charmap_1_Frame">
12802 <div id="Charmap_1_Text" class="Charmap">
12803 </div>
12804 <br>
12805 <style>
12806 #Charmap_1_Frame {
12807     overflow:scroll;
12808     height:3200px; width:3200px;
12809     transform:scale(0.5);
12810     zoom:0.25;
12811     resize:both;
12812     background-color:#fff;
12813 }
12814 .Charmap {
12815     zoom:0.25;
12816     font-size:16px;
12817     line-height:1.6;
12818     xfont-family:Georgia;
12819     color:#000;
12820 }
12821 <style>
12822 <script>
12823     function charmagpen(){
12824         text = '';
12825         for(cc = 0; cc < 0x10000; cc++){
12826             text += String.fromCharCode(cc);
12827         }
12828     }
12829     Charmap_1_Text.innerHTML = text;
12830 }
12831 Charmap_Work.addEventListener('click',charmagpen);
12832 //charmagpen();
12833 </script>
12834
12835 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12836 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12837 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12838 <span id="Charmap_WorkCodeView"></span>
12839 <script id="Charmap_WorkScript">
12840     function Charmap_OpenWorkCodeView(){
12841         Charmap_showWorkCode();
12842         showHTMLCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12843     }
12844     Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCode);
12845 }
12846 Charmap_openWorkCodeView(); // should be invoked by an event
12847 </script>
12848 </details>
12849 <!-- Charmap_WorkCodeSpan -->
12850 <!-- </span>
12851 //<!-- ===== Work } ===== -->
12852
12853 </-- ===== Work ===== -->
12854 <span id="Pointillism_WorkCodeSpan">
12855 </span>
12856 <!-- Pointillism -->
12857 <summary>Collaborated Pointillism</summary>
12858 <!-- Collaborated Pointillism -->
12859 <h2>a name="Pointillism" class="Pointillism"><a href="#Pointillism">Collaborated Pointillism</a>
12860
12861 <input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
12862 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
12863 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
12864 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Repeat">
12865 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
12866 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Save">
12867 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Load">
12868 <div id="Pointillism_1" class="Pointillism">
12869 <div id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" style="width:300px; height:300px"></div>
12870
12871 <span id="Pointillism_1_Unit_1" class="Pointillism Unit">
12872 <span id="Pointillism_1_XY_1" class="Pointillism XY">XY</span>
12873 <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY-remote</span>
12874 <span id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" style="width:300px; height:300px"></span>
12875
12876 <span id="Pointillism_1_Unit_2" class="Pointillism Unit">
12877 <span id="Pointillism_1_XY_2" class="Pointillism XY">XY</span>
12878 <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
12879 <span id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" style="width:300px; height:300px"></span>
12880
12881 </div>
12882 <br>
12883 <style>
12884 .Pointillism {
12885     display:block;
12886     resize:both;
12887     width:680px;
12888     height:400px;
12889     min-width:240px;
12890     min-height:270px;
12891     background-color:#eee;
12892     overflow:scroll;
12893     font-size:16px;
12894     font-family:Georgia;
12895     color:#000;
12896     padding:10px;
12897     margin:5px;
12898     vertical-align:middle;
12899 }
12900 .Pointillism_Unit {
12901     display:relative;
12902     top:0px;
12903     display:block;
12904     overflow:scroll;
12905     width:300px;
12906     height:350px;
12907     margin:5px;
12908     padding:10px;
12909     background-color:rgba(255,255,127,0.7);
12910 }
12911 .Pointillism_XY {
12912     display:flex;
12913     justify-content:center;
12914     width:290px;
12915     xheight:20px;
12916     font-size:12px;
12917     line-height:1.2;
12918     padding:5px;
12919     margin:0px;
12920     color:#fff;
12921 }

```

```

12921     background-color:#44c;
12922 }
12923 .Pointillism_XY_Remote {
12924     display:block;
12925     vertical-align:middle;
12926     width:290px;
12927     xheight:20px;
12928     font-size:12px;
12929     line-height:1.2;
12930     padding:5px;
12931     color:fff;
12932     background-color:#4a4;
12933 }
12934 .Pointillism_Canvas {
12935     display:block;
12936     position:relative;
12937     xpadding:20px;
12938     xleft:0px;
12939     xtop:20px;
12940     background-color:#333;
12941 }
12942 </style>
12943 <script>
12944 var points = [];
12945 var replay = [];
12946 var replayx = 0;
12947 function pClearCanvas(can){
12948     ctx = can.getContext('2d');
12949     ctx.clearRect(0,0,can.width,can.height);
12950 }
12951 function Pointillism_1_ClearCanvas(){
12952     pClearCanvas(Pointillism_1_Canvas_1);
12953     pClearCanvas(Pointillism_1_Canvas_2);
12954 }
12955 function PointsReset(){
12956     points = [];
12957     replay = [];
12958     inRepeat = false;
12959     inReplay = false;
12960     Pointillism_1_ClearCanvas();
12961 }
12962 function Pointillism_1_ResetCanvas(){
12963     PointsReset();
12964     if( Pointillism_1_Share.checked ){
12965         //alert('--broad cast reset\n');
12966         GJ_BcastMessage('DRAW RESET');
12967     }
12968 }
12969 function Pointillism_1_ResetCanvasReceive(){
12970     //alert('--received reset\n');
12971     PointsReset();
12972 }
12973 function drawPoint(can,x,y,r,g,b){
12974     const ctx = can.getContext('2d');
12975     ctx.fillStyle = 'rgba('+r+','+g+','+b+',0.7)';
12976     ctx.fillRect(x,y,8,8);
12977 }
12978 function waitMs(serno,ms){
12979     console.log('wait '+serno+' '+ms+' ms');
12980     until = new Date();
12981     now = until.getTime();
12982     untilMs = now + ms;
12983     for( wi = 0 ; wi++ ){
12984         now = new Date();
12985         remMs = untilMs - nowMs();
12986         remMs = untilMs - nowMs();
12987         //console.log('wait '+wi': '+remMs+'/'+ms);
12988         if( remMs < 0 ){
12989             break;
12990         }
12991     }
12992 }
12993 var inReplay = false;
12994 function replay1(){
12995     rx = replayx;
12996     if( replay.length <= rx ) {
12997         return;
12998     }
12999     replayx += 1;
13000     pl = replay[rx];
13001     if( pl[1] == 1 ){
13002         can = Pointillism_1_Canvas_1;
13003     }else{
13004         can = Pointillism_1_Canvas_2;
13005     }
13006     drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
13007     if( inReplay == false ){
13008         console.log('wait ' +replayx+ ' Stopped');
13009     }
13010     return;
13011 }
13012 if( rx < replay.length-1 ){
13013     prevMs = replay[rx][0].getTime();
13014     nextMs = replay[rx+1].getTime();
13015     delayMs = nextMs - prevMs;
13016     //console.log('wait '+replayx+ ' delayMs'+ms');
13017     window.setTimeout(replay1,delayMs);
13018 }else{
13019     console.log('wait '+replayx+ ' Finished');
13020     if( inReplay ){
13021         window.setTimeout(replay1,1000);
13022     }
13023 }
13024 function Pointillism_1_ReplayCanvas(can){
13025     Pointillism_1_ClearCanvas();
13026     replay = points;
13027     replayx = 0;
13028     inReplay = true;
13029     replay();
13030 }
13031 var inRepeat = false;
13032 function repeat1(){
13033     Pointillism_1_ClearCanvas();
13034     replay = points;
13035     replayx = 0;
13036     replay();
13037     if( !inRepeat ){
13038         //window.setTimeout(repeat1,1000);
13039     }
13040 }
13041 function Pointillism_1_RepeatCanvas(can){
13042     if( inRepeat ){
13043         inRepeat = false;
13044         inReplay = false;
13045     }else{
13046         inRepeat = true;
13047         inReplay = true;
13048         repeat1();
13049     }
13050 }
13051
13052 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
13053 function Pointillism_Setup(){
13054     var moveCount1 = 0;
13055     var moveCount2 = 0;
13056
13057     var gjlinked = false;
13058     function copyLocal(msg){
13059         if( gjlinked == false ){
13060             if(!GJLink_Section.open == true;
13061                 GJ_Join();
13062             gjlinked = true;
13063         }
13064         GJ_BcastMessage('DRAW '+msg);
13065     }
13066     function showXY1(e){
13067         moveCount1 += 1;
13068         x = e.offsetX;
13069         y = e.offsetY;
13070         Pointillism_1_XY_1.innerHTML = 'XY1: '+'x=' +x+' , y=' +y+' /'+moveCount1+'/'+points.length;
13071         Pointillism_1_XY_2_Remote.innerHTML = 'XY1: '+'x=' +x+' , y=' +y+' /'+moveCount1;
13072         if( e.buttons || CopyLocal() ){
13073             points.push((new Date()).getTime(),x,y,64,255));
13074             drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
13075             if( CopyLocal() ){
13076                 drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13077             }
13078             GJdraw('1,'+x+','+y);
13079         }
13080     }
13081     function showXY2(e){
13082         moveCount2 += 1;
13083         x = e.offsetX;
13084         y = e.offsetY;
13085         Pointillism_1_XY_2.innerHTML = 'XY2: '+'x=' +x+' , y=' +y+' /'+moveCount2+'/'+points.length;
13086         Pointillism_1_XY_1_Remote.innerHTML = 'XY2: '+'x=' +x+' , y=' +y+' /'+moveCount2;
13087         if( e.buttons || CopyLocal() ){
13088             points.push((new Date()).getTime(),x,y,64,255));
13089             drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
13090             if( CopyLocal() ){
13091                 drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
13092                 //GJdraw('2,'+x+','+y);
13093             }
13094         }
13095     }
13096     Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
13097     Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);

```

```

13098 Pointillism_1_Unit_2.style.left = '340px';
13099 Pointillism_1_Unit_2.style.top = '-375px';
13100 }
13101 function Pointillism_RemoteDraw(arg){
13102 //alert('Draw at '+arg);
13103 //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13104 if( arg == 'RESET' ){
13105 Pointillism_1_ResetCanvasReceive();
13106 }else{
13107 argv = arg.split(',');
13108 x = argv[1];
13109 y = argv[2];
13110 Pointillism_1_XY_2_Remote.innerHTML = 'XYR: '+ 'x=' +x +', y=' +y +' /'+points.length;
13111 drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
13112 }
13113 }
13114 </script>
13115
13116 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13117 <input id="Pointillism_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13118 <input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13119 <span id="Pointillism_WorkCodeView"></span>
13120 <script id="Pointillism_WorkScript">
13121 function Pointillism_OpenWorkCodeView(){
13122   function Pointillism_ShowWorkCode(){
13123     showHTMLCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
13124   }
13125 }
13126 Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_ShowWorkCode);
13127
13128 Pointillism_OpenWorkCodeView(); // should be invoked by an event
13129 </script>
13130 </details>
13131 <!-- Pointillism_WorkCodeSpan -->
13132 <!-- ===== Work -->
13133 //<!-- ===== Work -->
13134
13135
13136
13137 <!-- ===== Work -->
13138 <span id="StatCounter_WorkCodeSpan">
13139 <details><summary>StatCounter</summary>
13140 <!-- ----- StatCounter // 2020-1018 SatoxITS ( -->
13141 <h2>StatCounter</h2>
13142 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
13143 <!-- counter as image tag--></div>
13144 <style>
13145   .statcounter {
13146     vertical-align:middle;
13147   }
13148   #sc_SatoxITS {
13149     color:#000;
13150     font-size:2pt;
13151     height:30px;
13152     width:100%;
13153     background-color:#ddd;
13154   }
13155 </style>
13156 <div>
13157 <script>
13158   var sc_project=12411639;
13159   var sc_invisible=0;
13160   var sc_security="laeb2a3a";
13161   var sc_https=1;
13162   var scHost = "https://";
13163 </script>
13164 <!-- script src="https://statcounter.com/counter/counter.js" -->
13165 </script> --> (counter by inline script)
13166 </div>
13167
13168 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13169 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13170 <input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13171 <span id="StatCounter_WorkCodeView"></span>
13172 <script id="StatCounter_WorkScript">
13173 function StatCounter_OpenWorkCodeView(){
13174   function StatCounter_ShowWorkCode(){
13175     showHTMLCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
13176   }
13177   StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_ShowWorkCode);
13178 }
13179 StatCounter_OpenWorkCodeView(); // should be invoked by an event
13180 </script>
13181 </details>
13182 <!-- StatCounter_WorkCodeSpan -->
13183 <!-- ===== Work -->
13184 //<!-- ===== Work -->
13185
13186
13187 <!-- ===== Work -->
13188 <span id="CascadedCanvasBook_WorkCodeSpan">
13189 <details id="CascadedCanvasBook_Section"><summary id="CascadedCanvasBook_Summary">CascadedCanvasBook</summary>
13190 <!-- ----- CascadedCanvasBook // 2020-1031 SatoxITS ( -->
13191 <h2>Cascaded Canvas Book</h2>
13192 <div>
13193 <!-- ----- CBPanel -->
13194 <div id="CBPanel" class="CBPanel">
13195 <input id="CS_new" type="button" value="NewCanvas">
13196 </div>
13197 <div>
13198 <br>
13199 <h3>Undo / Redo / Replay</h3>
13200 <div id="CanvasTool_UndoRedo">
13201 <span id="DrawReplay" class="CanvasTool" draggable="true" contenteditable="">>
13202 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13203 <input class="CV_Button" type="button" value="Redraw" type="text" value="0" onwheel="OnWheelInt()">
13204 To <input id="DrawingSernoView" data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()"/>
13205 </span>
13206 </div>
13207 <script>
13208 function childByName(node,name){
13209   for( let i = 0; i < node.children.length; i++ ){
13210     ch = node.children[i];
13211     name = ch.getAttribute('data-name');
13212     if( name == name ){
13213       return ch;
13214     }
13215   }
13216   return null;
13217 }
13218 function OnWheelInt(){
13219   event.preventDefault();
13220   t = event.target;
13221   n = t.nodeName;
13222   i = t.id;
13223   p = t.parentNode;
13224   y = event.deltaY;
13225   y = event.deltaY;
13226   if(y < 0) log('OnWheelInt '+y+' '+n+'#'+i+' '+t.value);
13227   if( y < 0 ) // scroll forward (up)
13228     inc = -y;
13229   else{
13230     inc = -y;
13231   }
13232   inc /= 6;
13233   val = parseFloat(t.value) + inc;
13234   t.value = val.toFixed(0);
13235   return val;
13236 }
13237 var DrawingSerno = 0;
13238 function saveDrawing(){
13239   DrawingSerno += 1;
13240   DrawingSernoView.value = DrawingSerno;
13241 }
13242 function to02x(x){
13243   if( x <= 0xF ){
13244     return '0'+x.toString(16);
13245   }else{
13246     return x.toString(16);
13247   }
13248 }
13249 </script>
13250 <div id="InstaColorPicker" draggable="true">
13251 <h3>Color Picker</h3>
13252 <div id="CanvasColor">
13253 Select value by
13254 <input id="ICPmotion" type="checkbox" checked="">Mouse Motion
13255 <input id="ICPautoAddMotion" type="checkbox" checked="">Auto. add to history
13256 <input id="ICPwheel" type="checkbox" checked="">Mouse Wheel
13257 <input id="ICPautoAddWheel" type="checkbox" checked="">Auto. add to history
13258
13259 <div data-name="Fore" id="CanvasTool_Color_Fore" class="CanvasTool" onchange="showColor1Sample()">
13260 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13261 <input class="CV_Button" type="button" value="Fore" type="text" value="32" onwheel="OnWheelHex()"/>
13262 R<input data-name="R" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()"/>

```

```

13275 G<input data-name="G" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13276 <input data-name="B" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13277 A<input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13278 RGBA<input data-name="C" class="ColorParam" type="text" value="#000000ff">
13279 <span data-name="Sample">Sample</span>
13280 </div>
13281 <div data-name="fill" id="CanvasTool_Color_Fill" class="CanvasTool" onchange="showColor1Sample()">
13282 <input class="CV_Button" type="button" value="clone" onclick="CV_cloneTool()"/>
13283 <input class="CV_Button" type="button" value="fill">
13284 R<input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13285 G<input data-name="G" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13286 B<input data-name="B" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13287 A<input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13288 RGBA<input data-name="C" class="ColorParam" type="text" value="#000000ff">
13289 <span data-name="Sample">Sample</span>
13290 </div>
13291 <div data-name="back" id="CanvasTool_Color_Back" class="CanvasTool" onchange="showColor1Sample()">
13292 <input class="CV_Button" type="button" value="clone" onclick="CV_cloneTool()"/>
13293 <input class="CV_Button" type="button" value="Back">
13294 R<input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13295 G<input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13296 B<input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13297 A<input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13298 RGBA<input data-name="C" class="ColorParam" type="text" value="#ffff00ff">
13299 <span data-name="Sample">Sample</span>
13300 </div>
13301 <div data-name="border" id="CanvasTool_Color_Border" class="CanvasTool" onchange="showColor1Sample()">
13302 <input class="CV_Button" type="button" value="clone" onclick="CV_cloneTool()"/>
13303 <input class="CV_Button" type="button" value="Border">
13304 R<input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13305 G<input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13306 B<input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13307 A<input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13308 RGBA<input data-name="C" class="ColorParam" type="text" value="#ffff00ff">
13309 <span data-name="Sample">Sample</span>
13310 </div>
13311 <div id="ColorComposition" class="ColorComposition">
13312 <span data-name="Sample">Sample</span>
13313 </div>
13314 <div id="ColorHistory" class="ColorHistory">
13315 <span data-name="Sample">Sample</span>
13316 </div>
13317 <b>
13318 <input class="LargeButton CV_Button" type="button" value="Clear Color History" onclick="ClearColorHistory()"/>
13319 <input id="LenColorHistory" class="CanvasParam" type="text" value="0">
13320 <div id="ColorHistoryList" class="CanvasParam" type="text" value="200">
13321 <div id="ColorHistory" class="ColorHistory">
13322 <div id="Color1" class="Color" onclick="SelectThisColor()"/>
13323 <div id="Color2" class="Color" onclick="SelectThisColor()"/>
13324 <div id="Color3" class="Color" onclick="SelectThisColor()"/>
13325 <div id="Color4" class="Color" onclick="SelectThisColor()"/>
13326 <div id="Color5" class="Color" onclick="SelectThisColor()"/>
13327 <div id="Color6" class="Color" onclick="SelectThisColor()"/>
13328 <div id="Color7" class="Color" onclick="SelectThisColor()"/>
13329 <div id="Color8" class="Color" onclick="SelectThisColor()"/>
13330 <div id="Color9" class="Color" onclick="SelectThisColor()"/>
13331 <div id="Color10" class="Color" onclick="SelectThisColor()"/>
13332 <div id="Color11" class="Color" onclick="SelectThisColor()"/>
13333 <div id="Color12" class="Color" onclick="SelectThisColor()"/>
13334 <div id="Color13" class="Color" onclick="SelectThisColor()"/>
13335 <div id="Color14" class="Color" onclick="SelectThisColor()"/>
13336 <div id="Color15" class="Color" onclick="SelectThisColor()"/>
13337 <div id="Color16" class="Color" onclick="SelectThisColor()"/>
13338 <div id="Color17" class="Color" onclick="SelectThisColor()"/>
13339 <div id="Color18" class="Color" onclick="SelectThisColor()"/>
13340 <div id="Color19" class="Color" onclick="SelectThisColor()"/>
13341 <div id="Color20" class="Color" onclick="SelectThisColor()"/>
13342 <div id="Color21" class="Color" onclick="SelectThisColor()"/>
13343 <div id="Color22" class="Color" onclick="SelectThisColor()"/>
13344 <div id="Color23" class="Color" onclick="SelectThisColor()"/>
13345 <div id="Color24" class="Color" onclick="SelectThisColor()"/>
13346 <div id="Color25" class="Color" onclick="SelectThisColor()"/>
13347 <div id="Color26" class="Color" onclick="SelectThisColor()"/>
13348 <div id="Color27" class="Color" onclick="SelectThisColor()"/>
13349 <div id="Color28" class="Color" onclick="SelectThisColor()"/>
13350 <div id="Color29" class="Color" onclick="SelectThisColor()"/>
13351 <div id="Color30" class="Color" onclick="SelectThisColor()"/>
13352 <div id="Color31" class="Color" onclick="SelectThisColor()"/>
13353 <div id="Color32" class="Color" onclick="SelectThisColor()"/>
13354 <div id="Color33" class="Color" onclick="SelectThisColor()"/>
13355 <div id="Color34" class="Color" onclick="SelectThisColor()"/>
13356 <div id="Color35" class="Color" onclick="SelectThisColor()"/>
13357 <div id="Color36" class="Color" onclick="SelectThisColor()"/>
13358 <div id="Color37" class="Color" onclick="SelectThisColor()"/>
13359 <div id="Color38" class="Color" onclick="SelectThisColor()"/>
13360 <div id="Color39" class="Color" onclick="SelectThisColor()"/>
13361 <div id="Color40" class="Color" onclick="SelectThisColor()"/>
13362 <div id="Color41" class="Color" onclick="SelectThisColor()"/>
13363 <div id="Color42" class="Color" onclick="SelectThisColor()"/>
13364 <div id="Color43" class="Color" onclick="SelectThisColor()"/>
13365 <div id="Color44" class="Color" onclick="SelectThisColor()"/>
13366 <div id="Color45" class="Color" onclick="SelectThisColor()"/>
13367 <div id="Color46" class="Color" onclick="SelectThisColor()"/>
13368 <div id="Color47" class="Color" onclick="SelectThisColor()"/>
13369 <div id="Color48" class="Color" onclick="SelectThisColor()"/>
13370 <div id="Color49" class="Color" onclick="SelectThisColor()"/>
13371 <div id="Color50" class="Color" onclick="SelectThisColor()"/>
13372 <div data-name="OnWheelHex()>
13373 event.preventDefault();
13374 t = event.target;
13375 n = t.nodeName;
13376 i = t.id;
13377 p = t.parentNode;
13378 y = event.deltaY;
13379 inc = -y; // scroll forward (up)
13380 inc /= 6;
13381
13382 val = parseFloat(t.value) + inc;
13383 val = val.toFixed(0);
13384 if( val < 0 ) val = 0;
13385 if( 255 < val ) val = 255;
13386 if( t.value != val || event.ctrlKey ){
13387 t.value = val;
13388 if( !ICPautoAddWheel.checked ){
13389 showColor1Sample();
13390 }
13391 }
13392 return val;
13393 }
13394 function SelectThisColor(){
13395 ok = confirm("Pick this color ? "+event.target.innerHTML);
13396 if( ok ){
13397 // add to Picked
13398 }
13399 }
13400 var LastMotion = 0;
13401 function motionColor(){
13402 if( !ICPmotion.checked ){
13403 return;
13404 }
13405 d = new Date();
13406 if( d.getTime() - LastMotion < 100 ){
13407 return;
13408 }
13409 LastMotion = d.getTime();
13410
t = CanvasTool_Color_Fore;
13411 R = childByName(t,'R');
13412 G = childByName(t,'G');
13413 B = childByName(t,'B');
13414 A = childByName(t,'A');
13415 // console.log('mouse motion '+event.x+','+event.y+' target#'+it);
13416 updated = false;
13417
13418 val = 255 * (event.x/window.innerWidth);
13419 val = val.toFixed(0);
13420 if( B.value != val || event.ctrlKey ){
13421 B.value = val;
13422 updated = true;
13423 }
13424
13425 val = 255 * (event.y/window.innerHeight);
13426 val = val.toFixed(0);
13427 if( G.value != val || event.ctrlKey ){
13428 G.value = val;
13429 updated = true;
13430 }
13431 if( updated ){
13432 showColor1Sample(t,ICPautoAddMotion.checked);
13433 }
13434 }
13435 function scrollColor(){
13436 if( !ICPmotion.checked ){
13437 return;
13438 }
13439 d = new Date();
13440 if( d.getTime() - LastMotion < 100 ){
13441 return;
13442 }
LastMotion = d.getTime();
13443
t = CanvasTool_Color_Fore;
13444 R = childByName(t,'R');
13445 G = childByName(t,'G');
13446 B = childByName(t,'B');
13447 A = childByName(t,'A');
13448
13449
13450
13451

```

```

13452     updated = false;
13453
13454     y = gsh.getBoundingClientRect().top.toFixed(0)
13455     if(y < 0) y *= -1;
13456     size = 10000;
13457     val = 255 * (y/size);
13458     val = val.toFixed(0);
13459     if(255 != val) val = 255;
13460     if(val != val || event.ctrlKey ){
13461         R.value = val;
13462         updated = true;
13463     }
13464     if( updated ){
13465         showColorSample1(t,ICPautoAddMotion.checked);
13466     }
13467 }
13468
13469 function showColor1Sample(){
13470     t = event.target.parentNode;
13471     showColor1Sample1(t,ICPautoAddWheel);
13472 }
13473 function showColorSample1(t,add){
13474     name = t.getAttribute('data-name');
13475
13476     R = childByName(t,'R').value;
13477     G = childByName(t,'G').value;
13478     B = childByName(t,'B').value;
13479     A = childByName(t,'A').value;
13480
13481     R = parseInt(R);
13482     G = parseInt(G);
13483     B = parseInt(B);
13484     A = parseInt(A);
13485
13486     R = to02x(R); //R.toString(16);
13487     G = to02x(G); //G.toString(16);
13488     B = to02x(B); //B.toString(16);
13489     A = to02x(A); //A.toString(16);
13490
13491     color = '#'+R+G+B+A;
13492     //console.log(name+' color='+color);
13493
13494     C = childByName(t,'C');
13495     C.value = color;
13496     S = childByName(t,'Sample');
13497     S.style.color = color;
13498
13499     ColorID += 1;
13500     cl = Colorl;
13501     clid = cl.id;
13502     cl.id = "color_"+ColorID;
13503     cl.innerHTML = cl.id + ' ';
13504     cl.innerHTML += 'font color=black>' + color + '<+' + font color=white>' + color + '<+' + font>';
13505     if( name == 'Font' ){
13506         ColorComposition.style.setProperty('color',color,'important');
13507         cl.style.setProperty('color',color,'important');
13508         cl.style.setProperty('background-color',color,'important');
13509     }
13510     if( name == 'Fill' ){
13511         ColorComposition.style.setProperty('border-color',color,'important');
13512         cl.style.setProperty('color',color,'important');
13513         cl.style.setProperty('background-color',color,'important');
13514     }
13515     if( name == 'Back' ){
13516         ColorComposition.style.setProperty('background-color',color,'important');
13517         cl.style.setProperty('background-color',color,'important');
13518     }
13519     if( name == 'Border' ){
13520         ColorComposition.style.setProperty('border-color',color,'important');
13521         cl.style.setProperty('background-color',color,'important');
13522         cl.style.setProperty('border-color',color,'important');
13523     }
13524     ccl = cl.cloneNode(true);
13525     cl.id = clid;
13526
13527     if( add ){
13528         max = parseInt(MaxColorHistory.value);
13529         if( HistLength < max ){
13530             HistLength++;
13531             LenColorHistory.value = HistLength;
13532             ColorHistory.insertBefore(ccl,LastColor);
13533             LastColor = ccl;
13534         }
13535         if( max <= HistLength ){
13536             LenColorHistory.style.setProperty('background-color','#f00','important');
13537         }else{
13538             LenColorHistory.style.setProperty('background-color','#fff','important');
13539         }
13540     }
13541 }
13542 function InstaColor_Setup1(){
13543     window.addEventListener('mousemove',motionColor);
13544     window.addEventListener('scroll',scrollColor);
13545     //window.addEventListener('click',motionColor); // click is generated for animation
13546
13547     fi = document.getElementById('FeaturesView');
13548     if( fi != null ){
13549         fi.appendChild(InstaColorPicker);
13550         //cs.appendChild(ElementById('instaColoSpan'));
13551         //cs.appendChild(InstaColorPicker);
13552         //ci.hidden = false;
13553         //ci.open = true;
13554     }
13555 }
13556 function InstaColor_Setup(){
13557     if( CascadedCanvasBook_Section.open ){
13558         InstaColor_Setup1();
13559     }
13560     CascadedCanvasBook_Summary.addEventListener('click',InstaColor_Setup1);
13561 }
13562 
```

```

13629 <input class="CV_Button" type="button" value="Yellow">
13630 #<span data-name="cv" class="ColorParam" type="text">>0.5</span>
13631 FC<input data-name="FC" class="ColorParam" type="text" value="1">
13632 <input data-name="FO" class="CanvasParam" type="text" value="#fae600">
13633 <span data-name="FCS" class="ColorSample">xxxx</span>
13634 <span data-name="BCS" class="ColorSample">xxxx</span>
13635 BC<input data-name="BC" class="ColorParam" type="text" value="0.9">
13636 BO<input data-name="BO" class="CanvasParam" type="text" value="#fffff">
13637 #<span data-name="BO" class="ColorParam" type="text" value="0.9">
13638 </div>
13639 </div>
13640 <script>
13641 // https://developer.mozilla.org/en-US/docs/Web/CSS/color_value
13642 function genCSSColorStyle(color,opa){
13643   opa = parseFloat(opa);
13644   opa = 0xFF;
13645   opa = opa.toFixed(0);
13646   if( !Opafloat) opa= 0xFF;
13647   opa = parseInt(opa);
13648   opa = opa.toString(16);
13649   if( opa < 0x10 ) opa = '0' + opa;
13650   color += opa;
13651   color += opa;
13652   return color;
13653 }
13654 function CV_GenColorStyle(cole,forFill){
13655   if( forFill ){
13656     col = childByName(cole,'FC').value;
13657     opa = childByName(cole,'FO').value;
13658   }else{
13659     col = childByName(cole,'BC').value;
13660     opa = childByName(cole,'BO').value;
13661   }
13662   color = genCSSColorStyle(col,opa);
13663   return color;
13664 }
13665 function xxxxshowColorSample(){
13666   t = event.target;
13667   p = t.parentNode;
13668   alert('showColorSample '+event.target.nodeName+'/'+p.id);
13669 }
13670 function showColorSample(){
13671   var csv = CanvasColors.children;
13672   //console.log('colors'+csv.length);
13673   for( i = 0; i < csv.length; i++ ){
13674     fc = childByName(csv[i],'FC').value;
13675     bc = childByName(csv[i],'BC').value;
13676   //console.log('color'+<span>+csv[i].id+' fc='+fc+' bc='+bc);
13677   fcs = childByName(csv[i], 'FCS');
13678   fcs.style.color = fc;
13679   fcs.style.borderColor = fc;
13680   fcs.style.backgroundColor = bc;
13681   bcs = childByName(csv[i], 'BCS');
13682   bcs.style.color = bc;
13683   bcs.style.borderColor = fc;
13684   bcs.style.backgroundColor = fc;
13685   bcs.style.backgroundParts();
13686 }
13687 CV_redrawParts();
13688 </script>
13689 <style>
13690 .ColorSample {
13691   color:#ff;
13692   background-color:#ff0;
13693   border:1px solid #000;
13694   margin:0px;
13695   padding:0px;
13696   width:12pt !important;
13697   height:12pt !important;
13698 },
13699 .ColorParam {
13700   color:#000 !important;
13701   font-family:Courier New, Monospace !important;
13702   font-size:9pt !important;
13703   padding:2px !important;
13704   line-height:1.1 !important;
13705   height:14pt !important;
13706   width:55pt !important;
13707   text-align:left !important;
13708   display:inline !important;
13709   vertical-align:middle !important;
13710 },
13711 .CanvasParam {
13712   color:#000 !important;
13713   font-family:Courier New, Monospace !important;
13714   font-size:9pt !important;
13715   padding:2px !important;
13716   line-height:1.1 !important;
13717   height:14pt !important;
13718   width:30pt !important;
13719   text-align:right !important;
13720   display:inline !important;
13721   vertical-align:middle !important;
13722 },
13723 .CV_Button {
13724   padding:2pt !important;
13725   border:2px inset #bbb !important;
13726   font-size:9pt !important;
13727   font-weight:bold !important;
13728   border-radius:3px !important;
13729   color:#000;
13730   font-size:9pt;
13731   line-height:1.2;
13732   width:50pt;
13733   color:#add; background-color:#66a !important;
13734   width:50pt;
13735 },
13736 .xxHtmlCodeviewText {
13737   font-size:9pt;
13738   font-family:Courier New;
13739   white-space:pre;
13740 },
13741 .xxCV_Button {
13742   font-family:Arial, Monospace, Courier New;
13743   color:#000;
13744   font-size:9pt;
13745   line-height:1.2;
13746   width:50pt;
13747 },
13748 </style>
13749 <h3>Parts</h3>
13750 <div id="AppendToCanvas" class="CanvasTool" draggable="true">
13751 Resize<input class="CanvasParam" type="text" value="100" onwheel="OnWheelResize()">%>
13752 Zoom<input class="CanvasParam" type="text" value="100" onwheel="OnWheelZoom()">%>
13753 <input id="RedrawImmediate" type="checkbox" value="Redraw" checked="">Redraw Immediate
13754 <div id="CanvasTools" class="CanvasTools" draggable="true" contenteditable="">>
13755 <input id="CanvasTool_Clear" class="CanvasTool" type="button" value="Clear" onclick="CV_clearTool()"/>
13756 <input data-name="rdt" class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()"/>
13757 #<span data-name="cvid" class="CanvasParam" type="text">>CL-0</span>
13758 P<input data-type="checkbox" value="1" checked="checked" />
13759 X<input data-name="X" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13760 Y<input data-name="Y" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13761 Z<input data-name="Z" class="CanvasParam" type="text" value="1" onwheel="OnWheelIntRedraw()"/>
13762 W<input data-name="W" class="CanvasParam" type="text" value="1000" onwheel="OnWheelIntRedraw()"/>
13763 R<input data-name="R" class="CanvasParam" type="text" value="1000" onwheel="OnWheelIntRedraw()"/>
13764 RO<input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13765 #CO<input data-name="C" class="CanvasParam" type="text" value="0">
13766 </div>
13767 <div id="CanvasTool_Rect" class="CanvasTool">
13768 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13769 <input data-name="rdt" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()"/>
13770 #<span data-name="cvid" class="CanvasParam" type="text">>RE-0</span>
13771 F<input data-name="F" type="checkbox" value="Fill" checked="checked" />
13772 X<input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()"/>
13773 Y<input data-name="Y" class="CanvasParam" type="text" value="60" onwheel="OnWheelIntRedraw()"/>
13774 Z<input data-name="Z" class="CanvasParam" type="text" value="12" onwheel="OnWheelIntRedraw()"/>
13775 H<input data-name="H" class="CanvasParam" type="text" value="80" onwheel="OnWheelIntRedraw()"/>
13776 RO<input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13777 #CO<input data-name="C" class="CanvasParam" type="text" value="1">
13778 </div>
13779 <div id="CanvasTool_Circle" class="CanvasTool">
13780 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13781 <input data-name="rdt" class="CV_Button" type="button" value="Circle" onclick="CV_drawCircle()"/>
13782 #<span data-name="cvid" class="CanvasParam" type="text">>CI-0</span>
13783 F<input data-name="F" type="checkbox" value="Fill" checked="checked" />
13784 X<input data-name="X" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()"/>
13785 Y<input data-name="Y" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()"/>
13786 Z<input data-name="Z" class="CanvasParam" type="text" value="24" onwheel="OnWheelIntRedraw()"/>
13787 R<input data-name="R" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()"/>
13788 S<input data-name="S" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()"/>
13789 #CO<input data-name="C" class="CanvasParam" type="text" value="2">
13790 </div>
13791 <div id="CanvasTool_Packman" class="CanvasTool">
13792 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13793 <input data-name="rdt" class="CV_Button" type="button" value="Packman" onclick="CV_drawPackman()"/>
13794 #<span data-name="cvid" class="CanvasParam" type="text">>P-0</span>
13795 P<input data-type="checkbox" value="Fill" checked="checked" />
13796 X<input data-name="X" class="CanvasParam" type="text" value="240" onwheel="OnWheelIntRedraw()"/>
13797 Y<input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()"/>
13798 Z<input data-name="Z" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13799 R<input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13800 S<input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>

```

```

13806 E<input data-name="E" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
13807 #CO<input data-name="C" class="CanvasParam" type="text" value="5">
13808 </div>
13809 <div id="CanvasTool_Ellipse" class="CanvasTool">
13810 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13811 <input data-name="rdr" class="CV_Button" type="button" value="Ellipse" onclick="CV_drawEllipse()">
13812 #span data-name="cvid" class="CanvasParam" type="checkbox" value="1" checked="checked">
13813 X<input data-name="X" class="CanvasParam" type="text" value="10" onwheel="OnWheelIntRedraw()">
13814 Y<input data-name="Y" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
13815 Z<input data-name="Z" class="CanvasParam" type="text" value="4" onwheel="OnWheelIntRedraw()">
13816 W<input data-name="RX" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
13817 H<input data-name="RY" class="CanvasParam" type="text" value="20" onwheel="OnWheelIntRedraw()">
13818 RO<input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13819 S<input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13820 E<input data-name="E" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()">
13821 #CO<input data-name="C" class="CanvasParam" type="text" value="4">
13822 </div>
13823 <div id="CanvasTool_Balloon" class="CanvasTool">
13824 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13825 #CO<input data-name="rdr" class="CV_Button" type="button" value="Balloon" onclick="CV_drawBalloon()">
13826 #span data-name="cvid" class="CanvasParam" type="text" value="BA-0c">
13827 F<input data-name="F" type="checkbox" value="Fill">
13828 X<input data-name="X" class="CanvasParam" type="text" value="280" onwheel="OnWheelIntRedraw()">
13829 Y<input data-name="Y" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
13830 Z<input data-name="Z" class="CanvasParam" type="text" value="5" onwheel="OnWheelIntRedraw()">
13831 W<input data-name="RX" class="CanvasParam" type="text" value="50" onwheel="OnWheelIntRedraw()">
13832 H<input data-name="RY" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
13833 RO<input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13834 S<input data-name="S" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()">
13835 E<input data-name="E" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
13836 #CO<input data-name="C" class="CanvasParam" type="text" value="4">
13837 </div>
13838 <div id="CanvasTool_Piechart" class="CanvasTool">
13839 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13840 #span data-name="rdr" class="CV_Button" type="button" value="Piechart" onclick="CV_drawPiechart()">
13841 #span data-name="cvid" class="CanvasParam" type="text" value="BA-0c">
13842 F<input data-name="F" type="checkbox" value="Fill" checked="checked">
13843 X<input data-name="X" class="CanvasParam" type="text" value="350" onwheel="OnWheelIntRedraw()">
13844 Y<input data-name="Y" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()">
13845 Z<input data-name="Z" class="CanvasParam" type="text" value="7" onwheel="OnWheelIntRedraw()">
13846 W<input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13847 H<input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13848 RO<input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13849 S<input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
13850 E<input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
13851 #CO<input data-name="C" class="CanvasParam" type="text" value="3">
13852 </div>
13853 <div id="CanvasTool_XArc1" class="CanvasTool">
13854 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13855 #span data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
13856 #span data-name="cvid" class="CanvasParam" type="text" value="XA-0c">
13857 F<input data-name="F" type="checkbox" value="Fill" checked="checked">
13858 X<input data-name="X" class="CanvasParam" type="text" value="460" onwheel="OnWheelIntRedraw()">
13859 Y<input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
13860 Z<input data-name="Z" class="CanvasParam" type="text" value="10" onwheel="OnWheelIntRedraw()">
13861 W<input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13862 H<input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13863 RO<input data-name="RO" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()">
13864 S<input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
13865 E<input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
13866 #CO<input data-name="C" class="CanvasParam" type="text" value="4">
13867 </div>
13868 <div id="CanvasTool_XArc2" class="CanvasTool">
13869 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13870 #span data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()">
13871 #span data-name="cvid" class="CanvasParam" type="text" value="XA-1c">
13872 F<input data-name="F" type="checkbox" value="Fill" checked="checked">
13873 X<input data-name="X" class="CanvasParam" type="text" value="580" onwheel="OnWheelIntRedraw()">
13874 Y<input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
13875 Z<input data-name="Z" class="CanvasParam" type="text" value="8" onwheel="OnWheelIntRedraw()">
13876 W<input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13877 H<input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
13878 RO<input data-name="RO" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
13879 S<input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()">
13880 E<input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()">
13881 #CO<input data-name="C" class="CanvasParam" type="text" value="2">
13882 </div>
13883 <canvas id="CV_partsCanvas" data-name="canvas" class="CS_Canvas" width="740" height="200"></canvas>
13884 </span>
13885 <script>
13886 function CV_redrawParts(){
13887   var parts = CV_parts.children;
13888   //console.log("parts=" + parts.length);
13889   np = [];
13890   for( i = 0; i < parts.length; i++ ){
13891     p = parts[i];
13892     z = childByName(p, 'z');
13893     if( z != null ){
13894       //console.log('#'+p.id+' z=' + z.value);
13895       np.push((z.value,p));
13896     }
13897   }
13898   np.sort(function(npl,np2){ return npl[0] - np2[0]; });
13899   CV.clearRect();
14000   for( i = 0; i < np.length; i++ ){
14001     p = np[i][1];
14002     redraw = childByName(p, 'rdr');
14003     //console.log("Redraw Z=" + np[i][0] + '#'+np[i][1].id+ ' redraw=' + redraw.onclick);
14004     if( redraw != null ){
14005       redraw.click();
14006     }
14007   }
14008 }
14009 function DrawingCanvas_Setup(){
14010   showColorSample();
14011   CV_redrawParts();
14012 }
14013 function OnWheelZoom(){
14014   val = OnWheelInt();
14015   canvas = CV_partsCanvas;
14016   canvas.style.zoom = val + 's';
14017   CV_redrawParts();
14018 }
14019 function OnWheelResize(){
14020   val = OnWheelInt();
14021   canvas = CV_partsCanvas;
14022   if( ! canvas.hasAttribute('data-width') ){
14023     w = canvas.width;
14024     h = canvas.height;
14025     canvas.setAttribute('data-width',w);
14026     canvas.setAttribute('data-height',h);
14027     sw = canvas.getAttribute('data-width');
14028     sh = canvas.getAttribute('data-height');
14029     console.log('Zoom save original w=' + w + ',h=' + h + ', sw=' + sw + ', sh=' + sh);
14030   }
14031   w = canvas.getAttribute('data-width');
14032   h = canvas.getAttribute('data-height');
14033   console.log('Zoom got original size w=' + w + ', h=' + h);
14034   nw = w * (val/100.0);
14035   nh = h * (val/100.0);
14036   //console.log('Zoom nw=' + nw + ', nh=' + nh);
14037   CV_partsCanvas.width = nw;
14038   CV_partsCanvas.height = nh;
14039   CV_redrawParts();
14040 }
14041 function OnWheelIntRedraw(){
14042   OnWheelInt();
14043   t = event.target;
14044   n = t.nodeName;
14045   i = t.id;
14046   p = t.parentNode;
14047   y = element.deltaY.toFixed(0);
14048   //console.log('OnWheelIntRedraw '+y+' '+n+'#'+i+' '+t.value+' #' + p.id);
14049   if( true ){
14050     CV_redrawParts();
14051   }else{
14052     if( RedrawImmediate.checked ){
14053       CV_clearRect();
14054     }if( p.id == 'CanvasTool_Circle' ){ CV_drawCircle(CanvasTool_Circle); }
14055     }if( p.id == 'CanvasTool_Rect' ){ CV_drawRect(CanvasTool_Rect); }
14056     CV_drawCircle(CanvasTool_Circle);
14057     CV_drawRect(CanvasTool_Rect);
14058   }
14059 }
14060 </script>
14061 function CV_setCtxStyle(ctx,p){
14062   C = childByName(p,'C').value;
14063   c = document.getElementById('CanvasTool_Color_'+C);
14064   ctx.fillStyle = CV_GenColorStyle(c,true);
14065   ctx.strokeStyle = CV_GenColorStyle(c,false);
14066   return ctx;
14067 }
14068 function CV_clearRect(){
14069   cv = document.getElementById('CV_partsCanvas');
14070   ctx = cv.getContext('2d');
14071   ctx.clearRect(0,0,cv.width,cv.height);
14072 }

```

```

13983}
13984 function CV_drawRect1(rect){
13985   canvas = document.getElementById('CV_partsCanvas');
13986   ctx = canvas.getContext('2d');
13987   ctx = CV_setCtxStyle(ctx,p);
13988
13989   p = rect;
13990   F = childByName(p,'F').checked;
13991   X = childByName(p,'X').value;
13992   Y = childByName(p,'Y').value;
13993   Z = childByName(p,'Z').value;
13994   p.style.zIndex = Z;
13995   W = childByName(p,'W').value;
13996   H = childByName(p,'H').value;
13997
13998   if( F ){
13999     ctx.fillRect(X,Y,W,H);
14000   }else{
14001     ctx.strokeRect(X,Y,W,H);
14002   }
14003   saveDrawing();
14004 }
14005 function CV_drawRect(rect){
14006   CV_drawRect1(CanvasTool_Rect);
14007 }
14008 function CV_drawCircle1(circle){
14009   canvas = document.getElementById('CV_partsCanvas');
14010   ctx = canvas.getContext('2d');
14011   ctx = CV_setCtxStyle(ctx,p);
14012
14013   p = circle;
14014   F = childByName(p,'F').checked;
14015   X = childByName(p,'X').value;
14016   Y = childByName(p,'Y').value;
14017   Z = childByName(p,'Z').value;
14018   p.style.zIndex = Z;
14019   R = childByName(p,'R').value;
14020   S = childByName(p,'S').value;
14021   E = childByName(p,'E').value;
14022
14023 //console.log('Circle'+X+','+Y+' F=' + F);
14024 ctx.beginPath();
14025 SA = (S / 180) * Math.PI;
14026 EA = (E / 180) * Math.PI;
14027 ctx.arc(X,Y,R,SA,EA);
14028 if( F ){
14029   ctx.fill();
14030 }else{
14031   ctx.stroke();
14032 }
14033 saveDrawing();
14034 }
14035 function CV_drawCircle(){
14036   CV_drawCircle1(CanvasTool_Circle);
14037 }
14038 function CV_drawEllipse1(circle){
14039   canvas = document.getElementById('CV_partsCanvas');
14040   ctx = canvas.getContext('2d');
14041   ctx = CV_setCtxStyle(ctx,p);
14042
14043   p = circle;
14044   X = childByName(p,'X').value;
14045   Y = childByName(p,'Y').value;
14046   Z = childByName(p,'Z').value;
14047   RX = childByName(p,'RX').value;
14048   RY = childByName(p,'RY').value;
14049   p.style.zIndex = Z;
14050   R0 = childByName(p,'R0').value;
14051   S = childByName(p,'S').value;
14052   E = childByName(p,'E').value;
14053
14054   ctx.beginPath();
14055   SA = (S / 180) * Math.PI;
14056   EA = (E / 180) * Math.PI;
14057   ROA = (RO / 180) * Math.PI;
14058   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14059
14060   F = childByName(p,'F').checked;
14061
14062   if( F ){
14063     ctx.fill();
14064   }else{
14065     // if( S ){
14066     {
14067       ctx.stroke();
14068     }
14069   }
14070   saveDrawing();
14071 }
14072 function CV_drawEllipse(){
14073   CV_drawEllipse1(CanvasTool_Ellipse);
14074 }
14075 function CV_drawBalloon1(balloon){
14076   canvas = document.getElementById('CV_partsCanvas');
14077   ctx = canvas.getContext('2d');
14078   ctx = CV_setCtxStyle(ctx,p);
14079
14080   p = balloon;
14081   F = childByName(p,'F').checked;
14082   X = childByName(p,'X').value;
14083   Y = childByName(p,'Y').value;
14084   Z = childByName(p,'Z').value;
14085   RX = childByName(p,'RX').value;
14086   RY = childByName(p,'RY').value;
14087   p.style.zIndex = Z;
14088   R0 = childByName(p,'R0').value;
14089   S = childByName(p,'S').value;
14090   E = childByName(p,'E').value;
14091
14092 //console.log('Ellipse'+X+','+Y+' F=' + F);
14093 ctx.beginPath();
14094
14095   SA = (S / 180) * Math.PI;
14096   EA = (E / 180) * Math.PI;
14097   ROA = (RO / 180) * Math.PI;
14098   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14099
14100   PX = parseInt(X);
14101   PY = parseInt(Y);
14102
14103 //console.log('Ellipse A '+PX+','+PY+' F=' + F);
14104   PX -= 25;
14105   PY += 10;
14106 //console.log('Ellipse B '+PX+','+PY+' F=' + F);
14107   ctx.lineTo(PX,PY);
14108   ctx.closePath();
14109
14110   if( F ){
14111     ctx.fill();
14112   }else{
14113     ctx.stroke();
14114   }
14115   saveDrawing();
14116 }
14117 function CV_drawBalloon(){
14118   CV_drawBalloon1(CanvasTool_Balloon);
14119 }
14120 function CV_drawPacman1(circle){
14121   canvas = document.getElementById('CV_partsCanvas');
14122   ctx = canvas.getContext('2d');
14123   ctx = CV_setCtxStyle(ctx,p);
14124
14125   p = circle;
14126   F = childByName(p,'F').checked;
14127   X = childByName(p,'X').value;
14128   Y = childByName(p,'Y').value;
14129   Z = childByName(p,'Z').value;
14130   p.style.zIndex = Z;
14131   R = childByName(p,'R').value;
14132   S = childByName(p,'S').value;
14133   E = childByName(p,'E').value;
14134
14135 //console.log('Pacman'+X+','+Y+' F=' + F);
14136 ctx.beginPath();
14137   SA = (S / 180) * Math.PI;
14138   //SA += 0.180 * Math.PI;
14139   EA = (E / 180) * Math.PI;
14140   EA += (E / 180) * Math.PI;
14141   EA -= (E / 180) * Math.PI;
14142   ctx.arc(X,Y,R,SA,EA);
14143   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14144
14145   ctx.beginPath();
14146   E = 180 - E;
14147   SA = (E / 180) * Math.PI;
14148   EA += (E / 180) * Math.PI;
14149   EA -= (E / 180) * Math.PI;
14150   ctx.arc(X,Y,R,SA,EA);
14151   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14152
14153   saveDrawing();
14154 }
14155 function CV_drawPacman(){
14156   CV_drawPacman1(CanvasTool_Pacman);
14157 }
14158 function CV_drawPiechart1(balloon){
14159   canvas = document.getElementById('CV_partsCanvas');
14160   ctx = canvas.getContext('2d');
14161   ctx = CV_setCtxStyle(ctx,p);
14162
14163   p = balloon;
14164   F = childByName(p,'F').checked;

```

```

14160 X = childByName(p,'X').value;
14161 Y = childByName(p,'Y').value;
14162 Z = childByName(p,'Z').value;
14163 RX = childByName(p,'RX').value;
14164 RY = childByName(p,'RY').value;
14165 p.style.zIndex = Z;
14166 RO = childByName(p,'RO').value;
14167 S = childByName(p,'S').value;
14168 E = childByName(p,'E').value;
14169
14170 //console.log('Ellipse'+X+', '+Y+' F='+F);
14171 ctx.beginPath();
14172
14173 SA = (S / 180) * Math.PI;
14174 EA = (E / 180) * Math.PI;
14175 ROA = (RO / 180) * Math.PI;
14176 ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14177
14178 PX = parseInt(X);
14179 PY = parseInt(Y);
14180 //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14181 //PX = 25;
14182 //PY = 40;
14183 //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14184 ctx.lineTo(PX,PY);
14185 ctx.closePath();
14186
14187 if( F ){
14188   ctx.fill();
14189 }else{
14190   ctx.stroke();
14191 }
14192 saveDrawing();
14193 }
14194 function CV_drawPiechart(){
14195   CV_drawPiechart1(CanvasTool_Piechart);
14196 }
14197
14198 function CV_drawArc1(balloon){
14199   canvas = document.getElementById('CV_partsCanvas');
14200   ctx = canvas.getContext('2d');
14201   ctx = CV_setCtxStyle(ctx,p);
14202
14203   p = balloon;
14204   F = childByName(p,'F').checked;
14205   X = childByName(p,'X').value;
14206   Y = childByName(p,'Y').value;
14207   Z = childByName(p,'Z').value;
14208   RX = childByName(p,'RX').value;
14209   RY = childByName(p,'RY').value;
14210   p.style.zIndex = Z;
14211   RO = childByName(p,'RO').value;
14212   S = childByName(p,'S').value;
14213   E = childByName(p,'E').value;
14214
14215 //console.log('Ellipse'+X+', '+Y+' F='+F);
14216 ctx.beginPath();
14217
14218 if( true ){
14219   d = new Date();
14220   ms = d.getTime();
14221   id = (ms % 300) / 10;
14222   //S = parseFloat(E) + id/2;
14223   E = (S + id) / 2;
14224   xd = (ms % 10000) / 5;
14225   if( 1000 < xd ){
14226     xd = 2000 - xd;
14227   }
14228   xd *= 0.8;
14229   X = parseFloat(X) + xd - 600;
14230   //console.log('Ellipse S='+S+', E='+E+' id='+id);
14231
14232   SA = (S / 180) * Math.PI;
14233   EA = (E / 180) * Math.PI;
14234   ROA = (RO / 180) * Math.PI;
14235   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14236 }
14237
14238 PX = parseInt(X);
14239 PY = parseInt(Y);
14240
14241 //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14242 //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14243
14244 ctx.lineTo(PX,PY);
14245 ctx.closePath();
14246
14247 if( F ){
14248   ctx.fill();
14249 }else{
14250   ctx.stroke();
14251 }
14252 saveDrawing();
14253 }
14254 function CV_drawArc2(){
14255   t = event.target;
14256   CV_drawArc1(t.parentNode);
14257 }
14258 var AnimeItvl = window.setInterval(CV_redrawParts,30);
14259
14260</script>
14261
14262 <h3>Animation</h3>
14263 <span id="Animation" class="CanvasTool" draggable="true" contenteditable="true">
14264   <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14265   <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14266   <span data-name="cvid" class="CanvasParam" type="text">ANIMA-0</span>
14267   <input data-name="r" class="ColorParam" type="text" value="Rotate">
14268   <input data-name="x" class="CanvasParam" type="text" value="30" onwheel="OnWheelInt()"/>ms
14269   DURA:<input data-name="t" class="CanvasParam" type="text" value="10" onwheel="OnWheelInt()"/>s
14270 </span>
14271
14272 <h3>Canvas</h3>
14273 <span id="AppendToCanvas" class="CanvasTool" draggable="true">
14274   <input class="CV_Button" type="button" value="Append"> the above part
14275 </span>
14276 <span id="CanvasWrapTemplate" class="CanvasWrap" draggable="true">
14277   <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14278   <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14279   #Canvas data-name="cvid" class="CanvasParam" type="text"><br>
14280   W:<input data-name="width" class="CanvasParam" type="text" onchange="CS_setSize()" value="700">
14281   H:<input data-name="height" class="CanvasParam" type="text" onchange="CS_setSize()" value="400">
14282   Zoom:<input data-name="zoom" class="CanvasParam" type="text" onchange="CS_setSize()" value="100" onwheel="OnWheelCanvasZoom()">
14283   <input class="CV_Button" type="button" value="Remove" onclick="removeParent!"/><br>
14284   <span id="DrawingCanvas" data-name="canvas" class="CS_Canvas" width="700" height="400"></span>
14285 </span>
14286 <script>
14287 function OnWheelCanvasZoom(){
14288   val = OnWheelInt();
14289   DrawingCanvas.style.zoom = val + '%';
14290   //CanvasWrapTemplate.style.width = (700*(val/100.0)+20) + 'px';
14291   CanvasWrapTemplate.style.height = (400*(val/100.0)+30) + 'px';
14292 }
14293 </script>
14294
14295 <h3>CanvasBook</h3>
14296 <div id="CanvasBook"></div>
14297 <br>
14298 <style>
14299 .CanvasBook {
14300   overflow:scroll;
14301 }
14302 .CPPanel {
14303 }
14304 .CanvasTool {
14305   font-size:9pt;
14306   font-family:Courier New;
14307   color:#000 !important;
14308   xborder:1px solid #aaaf;
14309   background-color:rgba(127,127,127,0.5);
14310   width:740px;
14311   white-space:nowrap;
14312   xborder:1px solid #aaaf;
14313   margin:4px;
14314   padding-left:4px;
14315   padding-right:4px;
14316   overflow:auto;
14317   display:inline-block;
14318   res:both;
14319   vertical-align:top;
14320   zoom:1.0;
14321 }
14322 .CanvasWrap {
14323   font-size:9pt;
14324   font-family:Courier New;
14325   color:#000 !important;
14326   border:1px solid #aaaf;
14327   background-color:rgba(200,200,200,0.2);
14328   width:740px;
14329   height:400px;
14330   margin:4px;
14331   padding-left:4px;
14332   padding-right:4px;
14333   overflow:auto;
14334   display:inline-block;
14335   res:both;
14336   vertical-align:top;
14337 }

```

```

14337    zoom:1.0;
14338}
14339.CS_Panel {
14340    padding:3px;
14341    vertical-align:middle;
14342}
14343.CS_Canvas {
14344    border:1px dashed #fcc;
14345    resize:both;
14346    zoom:1.0;
14347}
14348</style>
14349<script>
14350    var canvasID = 0;
14351    function removeParent(){
14352        e = event.target;
14353        p = e.parentNode;
14354        if( p == CanvasWrapTemplate ){
14355            return;
14356        }
14357        pp = p.parentNode;
14358        alert('removeParent #' + pp.id + '/' + p.id + '#' + e.id);
14359        p.parentNode.removeChild(p);
14360    }
14361    function cloneParent(){
14362        b = event.target;
14363        w = b.parentNode;
14364        CanvasID += 1;
14365        //pp = w.parentNode;
14366        cw = w.cloneNode(true);
14367        cw.setAttribute('id','cvid'+CanvasID);
14368        childByName(cw,'cvid').innerHTML = CanvasID;
14369        childByName(cw,'cvid').value = CanvasID;
14370        CanvasBook.appendChild(cw);
14371    }
14372    function CS_getSize(){
14373        b = event.target;
14374        p = e.parentNode;
14375        console.log('resize '+e.nodeName+' '+p.nodeName);
14376        c = childByName(p,'canvas');
14377        w = childByName(p,'width').value;
14378        h = childByName(p,'height').value;
14379        console.log('resize '+c.nodeName+' '+w+', '+h);
14380        c.width = w;
14381        c.height = h;
14382        p.style.width = w + 'px';
14383        p.style.height = h + 'px';
14384        console.log('c=' + c + ' ' + w + '/' + c.width + ', ' + h + '/' + c.height);
14385    }
14386    function CS_newFunc(){
14387        cvt = CanvasWrapTemplate;
14388        cvw = CanvasWrapTemplate.cloneNode(true); //needs an argument, otherwise 'functiuon notfound'
14389        CanvasID = 1;
14390        w = 'Canvas' + CanvasID;
14391        //childByName(cvw,'cvid').contenteditable = false;
14392        childByName(cvw,'cvid').innerHTML = CanvasID;
14393        childByName(cvw,'cvid').value = CanvasID;
14394        childByName(cvw,'width').value = w = childByName(cvt,'width').value;
14395        childByName(cvw,'width').value = w = childByName(cvt,'height').value;
14396        childByName(cvw,'width').value = w + 'px';
14397        //ncv = document.createElement('canvas');
14398        ncv = childByName(cvw,'canvas');
14399        //ncv.setAttribute('class','CS_Canvas');
14400        //ncv.setAttribute('data-name','canvas');
14401        ncv.width = w;
14402        ncv.height = h;
14403        //cvw.replaceChild(ncv,childByName(cvw,'canvas'));
14404        CanvasBook.appendChild(cvw);
14405    }
14406    //CS_new.addEventListener('click',CS_newFunc);
14407</script>
14408
14409
14410<input id="CanvasBook_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14411<input id="CanvasBook_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14412<input id="CanvasBook_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14413<span id="CanvasBook_WorkCodeView"></span>
14414<script id="CanvasBook_WorkScript">
14415function CanvasBook_openWorkCodeView(){
14416    function CanvasBook_showWorkCode(){
14417        showHtmlCode(CanvasBook_WorkCodeView,CascadedCanvasBook_WorkCodeSpan);
14418    }
14419    CanvasBook_WorkCodeViewOpen.addEventListener('click',CanvasBook_showWorkCode);
14420}
14421CanvasBook.openWorkCodeView(); // should be invoked by an event
14422</script>
14423</details>
14424<div>CanvasBook_WorkCodeSpan >-->
14425</div>!-- ===== Work } ===== -->
14426</div>!-- ===== Work } ===== -->
14427
14428
14429<!-- ===== Work { ===== -->
14430//<span id="SVG_WorkCodeSpan" open="><a href="#SVG">SVG</a></a name="SVG">SVG</a>
14431/*
14432<details id="SVG_Section" open="><summary id="SVG_Summary">SVG Getting Started</summary>
14433<!-- ===== SVG 2020-1108 SatoxITS { -->
14434<a href="#">Getting Started SVG</a>
14435
14436<div>
14437<svg id="xSVG_01" class="SVG100" viewBox="0 0 100 100">
14438<circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14439<circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14440<circle cx="65" cy="35" r="03" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14441<circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14442</svg>
14443
14444<svg id="xSVG_02" class="SVG100" viewBox="0 0 100 100">
14445<circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14446<circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14447<circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14448<circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14449</svg>
14450
14451<svg id="xSVG_03" class="SVG100" viewBox="0 0 100 100">
14452<circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14453<circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14454<circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14455<path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14456</svg>
14457
14458<svg id="xSVG_04" class="SVG100" viewBox="0 0 100 100">
14459<circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14460<circle cx="35" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14461<circle cx="65" cy="35" r="05" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14462<path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14463</svg>
14464
14465<svg id="xSVG_05" class="SVG100" viewBox="0 0 100 100">
14466<circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="#fffff00"></circle>
14467<circle cx="38" cy="42" r="04" stroke="black" stroke-width="1" fill="#00000000"></circle>
14468<circle cx="62" cy="42" r="04" stroke="black" stroke-width="1" fill="#00000000"></circle>
14469<path stroke="black" d="M 20 50 A 10 10 0 0 0 80 50 Z"></path>
14470</svg>
14471</div>
14472<style>
14473    .SVG100 {
14474        width:100px;
14475        height:100px;
14476    }
14477</style>
14478</div>
14479<div>
14480    var svg_deg = 0;
14481    function rotateSVG(){
14482        //ms = new Date().getTime();
14483        //deg = (ms_toFixed(0)/10) % 360;
14484        svg_deg += 2;
14485        xSVG_04.style.transform = 'rotate('+svg_deg+'deg)';
14486        xSVG_04.style.transform = 'rotate('+svg_deg+'deg)';
14487
14488    }
14489    window.setInterval(rotateSVG,30);
14490</script>
14491
14492<input id="SVG_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14493<input id="SVG_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14494<input id="SVG_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14495<span id="SVG_WorkCodeView"></span>
14496<script id="SVG_WorkScript">
14497    function SVG_openWorkCodeView(){
14498        function SVG_showWorkCode(){
14499            showHtmlCode(SVG_WorkCodeView,SVG_WorkCodeSpan);
14500        }
14501        SVG_WorkCodeViewOpen.addEventListener('click',SVG_showWorkCode);
14502    }
14503    SVG.openWorkCodeView(); // should be invoked by an event
14504</script>
14505</details>
14506<!-- SVG_WorkCodeSpan } -->
14507</div>!-- ===== Work } ===== -->
14508//<div>!-- ===== Work } ===== -->
14509
14510
14511
14512
14513

```

```

14514<!-- ===== Work { ===== -->
14515</span id="X3DROFL_WorkCodeSpan"><a href="#X3DROFL">X3DROFL</a></a name="X3DROFL">X3DROFL</a>
14517<b>details id="X3DROFL_Section" open=""><summary id="X3DROFL_Summary">X3DROFL Getting Started</summary>
14518<!-- ----- X3DROFL // 2020-1111 SatoshiTS { -->
14519<h2>X3D-ROFL</h2>
14520
14521<!-- x3dom JavaScript and CSS BEGIN( -->
14522<h3><a href="/gshell/x3dom-1.8.2-dev/">x3dom-1.8.2-dev</a></h3>
14523
14524
14525<!--
14526<canvas id="Smilly_1" width="150" height="150"/>
14527<script>
14528function draw_smiley1() {
14529    var canvas = Smilly_1;
14530    if( canvas.getContext ) {
14531        var ctx = canvas.getContext('2d');
14532        ctx.beginPath();
14533        ctx.arc(75, 75, 50, 0, Math.PI * 2, true); // Outer circle
14534        ctx.fillStyle = 'rgba(255,240,0,0.5)';
14535        ctx.fill();
14536
14537        ctx.beginPath();
14538        ctx.arc(60, 65, 5, 0, Math.PI * 2, true); // Left eye
14539        ctx.moveTo(110, 75);
14540        ctx.arc(75, 75, 35, 0, Math.PI, false); // Mouth (clockwise)
14541        ctx.moveTo(65, 65);
14542        ctx.stroke();
14543
14544        ctx.beginPath();
14545        ctx.arc(60, 65, 5, 0, Math.PI * 2, true); // Right eye
14546        ctx.moveTo(95, 65);
14547        ctx.arc(90, 65, 5, 0, Math.PI * 2, true);
14548        ctx.stroke();
14549        ctx.fillStyle = 'rgba(0,0,0,0.8)';
14550        ctx.fill();
14551    }
14552}
14553function getSmillyUrl(){
14554    draw_smiley1();
14555    url = Smilly_1.toDataURL("image/png");
14556    return url;
14557}
14558//getSmillyUrl();
14559function putTextureTag(){
14560    url = getSmillyUrl();
14561    document.write('<'+ 'ImageTexture url="'+url+'><'+ '/ImageTexture>');
14562}
14563</script>
14564
14565<div>
14566    Viewpoint
14567    <position><input id="X3dRofl_ViewPosition_Value" class="X3DOM_Param" type="text" value="0 0 0">
14568    <orientation><input id="X3dRofl_ViewOrientation_Value" class="X3DOM_Param" type="text" value="0 0 0">
14569    </div>
14570<div>
14571    Box
14572    <position><input id="X3dRofl_BoxPosition" class="X3DOM_Param" type="text" value="0 0 0 0">
14573    <rotation><input id="X3dRofl_BoxRotation_Value" class="X3DOM_Param" type="text" value="0 0 0 0">
14574    </div>
14575
14576<span id="x3domInclude">
14577<script type="text/javascript" src="/gshell/x3dom-1.8.2-dev/x3dom.js"></script>
14578<link rel="stylesheet" type="text/css" href="/gshell/x3dom-1.8.2-dev/x3dom.css">
14579</span>
14580<!-- x3dom JavaScript and CSS }END -->
14581
14582<div class="x3dRofl">
14583    <div id="x3dRoflScene" class="x3dRoflScene" width="600px" height="300px" draggable="false">
14584        <viewpoint id="X3dRofl_ViewPoint" position="0.8 0.7 4"></viewpoint>
14585        <transform id="Box1Trans" translation="1.0 0.5 0.5" rotation="1 1 1 1">
14586            <shape id="xBox1Box">
14587                <appearance>
14588                    <material id="Box1Material" type="diffuse">
14589                        <script>putTextureTag();</script>
14590                        <imagetexture id="Box1Texture" url="GShellInside00.png"></imagetexture>
14591                    </material>
14592                    <apperance>
14593                        <shape id="xBox1Box" size="1 1 2"></box>
14594                    </shape>
14595                </transform>
14596            </scene>
14597        </x3d>
14598        <style>
14599            .x3dRofl {
14600                background-color:#e0f0ff80;
14601                position: relative;
14602                display: block;
14603                overflow: visible !important;
14604                resize:both !important;
14605                resizerboth: !important;
14606            }
14607            .x3dRoflScene {
14608                z-index:100;
14609                background-color:#e0f0ff80;
14610                position: relative;
14611                display: block;
14612                overflow:visible !important;
14613                resize:both !important;
14614                resizerboth: !important;
14615                resizerboth: !important;
14616                resizerboth: !important;
14617                resizerboth: !important;
14618                resizerboth: !important;
14619            }
14620            .X3DOM_Param {
14621                color:#0000 !important;
14622                font-family:Consolas, "Lucida Console", Monospace !important;
14623                font-size:1em !important;
14624                padding:2px !important;
14625                line-height:1.1 !important;
14626                height:14pt !important;
14627                width:120pt !important;
14628                text-align:left !important;
14629                display:inline !important;
14630                vertical-align:middle !important;
14631            }
14632        </style>
14633
14634        <script>
14635            //<transform translation="0 0 0">
14636            //<transform id="Box1Box" translation="0 0 0">
14637        </script>
14638        function showVP(){
14639            console.log('Box1Material.getAttribute('diffusecolor'));
14640            m = document.getElementById('Box1Material');
14641            //console.log('color:'+m.getFieldValue('diffusecolor'));
14642            //console.log('color:' +m.getFieldValue('diffusecolor'));
14643            //console.log('rotation:' +Box1Box.getFieldValue('rotation'));
14644            //console.log('Box1TransRotation:' +Box1Trans.getFieldValue('rotation'));
14645
14646            e = document.getElementById('Box1View');
14647            console.log('Box1View:' +e.runtime.properties());
14648            e = document.getElementById('X3dRofl_ViewPoint');
14649            console.log('Box1Box:' +e.outerHTML);
14650
14651            //e.runtime.showWall();
14652            //e.runtime.resetView();
14653
14654            //console.log('Transform:' +x3dom.runtime.getCurrentTransform(Box1View));
14655            //console.log('Transform:' +Box1View.runtime.getRuntimeTransform());
14656            //console.log('Transform:' +X3dRofl_ViewPoint.runtime.getRuntimeTransform());
14657            //console.log('TransformPosition:' +X3dRofl_ViewPoint.runtime.position);
14658            console.log('VPposition:' +X3dRofl_ViewPoint.getFieldValue('position'));
14659            console.log('Orientation:' +X3dRofl_ViewPoint.getFieldValue('orientation'));
14660            //console.log('VPositionProp:' +Box1View.runtime.properties());
14661            //console.log('VPpositionProp:' +X3dRofl_ViewPoint.runtime.properties());
14662            //id = document.getElementById('Box1Scene');
14663            //console.log('Box1Scene:' +id.runtime.properties());
14664            //X3dRofl_ViewPoint.value = X3dRofl_ViewPoint.position;
14665
14666            X3dRofl_ViewPosition_Value.value = X3dRofl_ViewPoint.getFieldValue('position');
14667            X3dRofl_ViewOrientation_Value.value = X3dRofl_ViewPoint.getFieldValue('orientation');
14668            X3dRofl_BoxPosition_Value = Box1Trans.getFieldValue('translation');
14669            X3dRofl_BoxRotation_Value.value = Box1Trans.getFieldValue('rotation');
14670
14671            box = document.getElementById('Box1');
14672            //X3dRofl_ViewOrientation_Value.value = box1.runtime.viewpoint();
14673
14674            window.setInterval(showVP,500);
14675            function newVP(){
14676                console.log('viewPoint changed:' +event);
14677            }
14678            vp = document.getElementById('X3dRofl_ViewPoint');
14679            vp.addEventListener('viewpointChanged',newVP,false);
14680            X3dRofl.runtime.ready = function(){
14681                console.log('-- X3DOM ready');
14682            }
14683            //x3dom.runtime.debug(true);
14684
14685            var svg_deg = 0;
14686            function rotateX3DROFL(){
14687                svg_deg += 2;
14688            }
14689            function X3DROFL_Setup(){
14690                X3dRofl_ViewPoint.setAttribute('position','0.8 0.7 5');

```

```

14691 X3dRofl_ViewPoint.setAttribute('position','0.8 0.7 4');
14692 Box_Material.setAttribute('diffusecolor','0 1 0');
14693 Box1Trans.setAttribute('translation','1.0 0.5 0.5');
14694 Box1Trans.setAttribute('rotation','1 1 1 1');
14695 }
14696 function X3DROFL_Setup(){
14697 if( X3DROFL_Section.open == true ){
14698 X3DROFL_Setup1();
14699 }
14700 }
14701 X3DROFL_Summary.addEventListener('click',X3DROFL_Setup);
14702 X3DROFL_Setup();
14703 </script>
14704 <input id="X3DROFL_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14705 <input id="X3DROFL_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14706 <input id="X3DROFL_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14707 <span id="X3DROFL_WorkCodeView"></span>
14708 <script id="X3DROFL_WorkCodeSpan">
14709 function X3DROFL_openWorkCodeView(){
14710 function X3DROFL_showWorkCode(){
14711 showHtmlCode(X3DROFL_WorkCodeView,X3DROFL_WorkCodeSpan);
14712 }
14713 }
14714 X3DROFL_WorkCodeViewOpen.addEventListener('click',X3DROFL_showWorkCode);
14715 }
14716 X3DROFL_openWorkCodeView(); // should be invoked by an event
14717 </script>
14718 </details>
14719 <!--_J3DROFL_WorkCodeSpan -->
14720 //</span>
14721 //<!-- ===== Work -->
14722 //<!-- ===== Work { ===== -->
14723 <span id="Template_WorkCodeSpan">
14724 <details><summary>Work Template</summary>
14725 <h2>Template of Work</h2>
14726 <style>
14727 </style>
14728 <script>
14729 </script>
14730 <input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14731 <input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14732 <input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14733 <span id="Template_WorkCodeView"></span>
14734 <script id="Template_WorkScript">
14735 function Template_openWorkCodeView(){
14736 function Template_showWorkCode(){
14737 showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
14738 }
14739 Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
14740 }
14741 Template_openWorkCodeView(); // should be invoked by an event
14742 </script>
14743 </details>
14744 <!--_Template_WorkCodeSpan -->
14745 //<!-- ===== Work -->
14746 //<!-- ===== Work { ===== -->
14747 <span id="Originalsouce_WorkCodespan">
14748 <details open=><summary>Original Source</summary>
14749 <!--_Originalsouce // 2020-1009 SatoxITS { -->
14750 <h2>Original Source of GSh</h2>
14751 <input id="Originalsouce_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14752 <input id="Originalsouce_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14753 <input id="Originalsouce_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14754 <span id="OriginalsouceTextElement"></span>
14755 <script id="Originalsouce_WorkCodeView"></span>
14756 <script id="Originalsouce_WorkScript">
14757 function Originalsouce_operateWorkCode(){
14758 function Originalsouce_showWorkCode(){
14759 showHtmlCode(Originalsouce_WorkCodeView,Originalsouce_WorkCodeSpan);
14760 }
14761 }
14762 <OriginalsouceCode = document.createElement.cloneNode();
14763 //OriginalsouceCode = gsh.cloneNode(true); //=====
14764 //console.log(`src0=\n${gsh.outerHTML}`);
14765 //console.log(`src1=\n${gsh.outerHTML}`);
14766 //console.log(`src2=\n${OriginalsouceNode.innerHTML}`);
14767 Originalsouce.openWorkCodeView(); // should be invoked by an event
14768 //showNodeAsHtmlSource(Originalsouce_WorkCodeView,OriginalsouceNode);
14769 function cloneOriginalNode(){
14770 if( false ){
14771 m0 = performance.memory;
14772 m0U = m0.usedJSHeapSize;
14773 console.log('-- heap bef clone: '
14774 `+m0.usedJSHeapSize+'/'+m0.totalHeapSize);
14775 }
14776 OriginalsouceNode = gsh.cloneNode(true);
14777 if( false ){
14778 m1 = performance.memory;
14779 m1U = m1.usedJSHeapSize;
14780 m0U = m0.usedJSHeapSize;
14781 //alert(`-- clone: used heap '+mu0+' -> '+mul+' = '+mu+` bytes`);
14782 console.log(`-- heap aft clone: '
14783 `+m1.usedJSHeapSize+'/'+m1.totalHeapSize);
14784 //OriginalsouceNode = document.documentElement.cloneNode(true);
14785 }
14786 }
14787 }
14788 function Gsh_setupPage(){
14789 GshSetImages();
14790 //Indexer_afterLoaded();
14791 //KeyCommand_afterLoaded();
14792 //GConsole_initConsole();
14793 GJConsole_initFactory();
14794 GJLink_init();
14795 InterFrameCom_init();
14796 Gshell_initTopbar();
14797 //VirtualTopo_init();
14798 //Df_topo_init();
14799 //Aft_Setup();
14800 Shading_Setup();
14801 window.setInterval(ShowResourceUsage,1000);
14802 //document.addEventListener('keydown',jgshCommand); // should be applied later?
14803 PointList_Setup();
14804 Fontlist_Setup();
14805 DrawingCanvas_Setup();
14806 InstaColor_Setup();
14807 }
14808 function OnLoad(){
14809 SaveOriginalnode();
14810 Gsh_setupPage();
14811 }
14812 document.addEventListener('load',Gsh_setupPage);
14813 </script>
14814 <!--_Originalsouce_WorkCodeSpan -->
14815 //<!-- ===== Work -->
14816 //<!-- ===== Work { ===== -->
14817 <div>
14818 <br><script>OnLoad();</script><br>
14819 </div>

```