```
/*<html>
<span id="gsh" data-title="GShell" data-author="sato@its-more.jp">
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="icon" id="GshFaviconURL" href=""><!-- place holder -->
<span id="GshVersion" hidden="">gsh--0.6.2--2020-10-07--SatoxITS</span>
<title>GShell-0.6.2 by SatoxITS</title>

<div id="GshSidebar" class="SbSidebar"><div id="GshIndexer"></div></div>
<div id="GshMain">
<header id="GshBanner" height="100px" onclick="shiftBG();">
<div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.6.2 // 2020-10-07 // SatoxITS</note></div>
</header>

//<!-- ---------- Work { ---------- -->
//<span id="Indexer_WorkCodeSpan">
/*
<details open><summary>Indexer</summary>
<!-- ---------- Indexer // 2020-1007 SatoxITS { -->
<h2>Indexer</h2>
<input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
<input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
<input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
<span id="Indexer_WorkCodeView"></span>
</details>
<style id="SidebarIndex">
#gsh {
    display:block;
    overflow:scroll !important;
}
#GshMain {
    position:relative;
    width:80% !important;
    left:19.5% !important;
}
#GshSidebar {
    z-index:0;
    position:relative !important;
    overflow:auto;
    resize:both !important;
    xxoverflow-y:hidden !important;
    height:100px !important;
    xxxdisplay:inline !important;
    left:0px;
    top:0px;
    width:19.5%;
    min-width:80px;
    height:100% !important;
    color:#f00;
    xxbackground-color:rgba(64,64,64,0.5);
}
#GshIndexer {
    z-index:0;
    position:relative;
    height:100%;
    left:0px;
    top:0px;
    scroll-behavior: overflow !important;
    padding-left:4pt;
    color:#f00;
    font-size:0.5em;
    background-color:rgba(64,160,64,0.6) !important;
    white-space:nowrap;
}
.IndexLine {
    font-size:8pt !important;
    font-family:Georgia;
    display:block;
    color:#fff;
    padding-right:4pt;
}
.IndexLine:hover {
    color:#228;
    background-color:#fff;
    xxtext-decoration:underline !important;
}
</style>

<script id="Indexer_WorkScript">
function Indexer_openWorkCodeView(){
    function Indexer_showWorkCode(){
        showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
    }
    Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
}
//Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
Indexer_openWorkCodeView();

var iserno = 0;
var GeneratedId = 0;
function generateIndex(ni,e,chv,nch,ht){
    // https://developer.mozilla.org/en-US/docs/Web/API/Element
    c = '';
    if( e.classList != null ){
        c = e.classList.value;
    }
    console.log('-- <'+e.nodeName+'> #'+e.id+' .'+c+' '+e.attributes);
    if( e.nodeName == '#text' ){ return ''; }
    if( e.nodeName == '#comment' ){ return ''; }
    if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
        id = e.innerHTML;
        GeneratedId += 1;
        eid = 'GenratedId-'+GeneratedId;
        e.id = eid;
    }else
    if( e.nodeName == 'SUMMARY' ){
        id = e.innerHTML;
        GeneratedId += 1;
        eid = 'GenratedId-'+GeneratedId;
        e.id = eid;
    }else
    if( and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxxxentry-content') ){
        console.log('-- DIV entry-content begin');
        id = e.innerHTML;
        GeneratedId += 1;
        eid = 'GenratedId-'+GeneratedId;
        e.id = eid;
        console.log('-- DIV entry-content end hash-child=',e.hasChildNodes());
    }else
    if( e.id == '' || e.id == 'undefined' ){
        return '';
    }else{
        id = '#'+e.id;
        eid = e.id;
    }
    iserno += 1;
    ht = '<'+'div id="GenaratedEref_'+iserno+'" class="IndexLine" href="'+eid+'">'
        + iserno+' '+ni+':'+e.nodeName + ':' + id;
    if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+'/div>'; }
    if( !e.hasChildNodes() ){ return ht + '<'+'/div>'; }
    chv = e.childNodes;
    nch = e.childNodes.length;
    if( chv != null ){ nch = chv.length; }
    ht += ' ('+nch+')' + '<'+'/div>';
    for( let i = 0; i < chv.length; i++ ){
        sec = ni+'.'+i;
        if( ni == '' ){ sec = i; }
        ht += generateIndex(sec,chv[i],null,0);
    }
    return ht;
}
function onClickIndex(e){
    tid = e.target.id;
    tge = document.getElementById(tid);
    eid = tge.getAttribute('href');
    rx = tge.getBoundingClientRect().left.toFixed(0)
    ry = tge.getBoundingClientRect().top.toFixed(0)
    if( false ){
        alert('index clicked mouse(x='+e.x+', y='+e.y+')'
            + '\ntid=#'+ tid + ' rx='+rx + ',ry='+ry
            + '\neid=' + eid + '\n'
            + '\nhtml='+ tge.outerHTML);
    }
    ee = document.getElementById(eid);
    sx = 'NaN';
    sy = ee.getBoundingClientRect().top;
    console.log('sx='+sx+',sy='+sy);
    ee.scrollIntoView()
    window.scrollTo(sx,sy)
    //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
}
function Indexer_afterLoaded(){
    sideindex = document.getElementById('GshIndexer');
    ht = '<'+'h3>GShell Index<'+'/h3>';
    ht += generateIndex("",document.getElementById('gsh'),null,0,'');
    if( (pri = document.getElementById('primary')) != null ){
        ht += generateIndex("",pri,null,0,'');
    }
    ht += '<'+'br>';
    ht += '<'+'br>';
    ht += '<'+'br>';
    ht += '<'+'br>';
    sideindex.innerHTML = ht;
    sideindex.addEventListener('click',onClickIndex);

    if( (pri = document.getElementById('primary')) != null ){
        console.log('-- Seems in WordPress');
        pri.style.zIndex = 2000;

        GshSidebar.style.setProperty('position','relative','important');
        GshSidebar.style.top = '-1400px';
        //GshSidebar.style.setProperty('position','absolute','important');
        //GshSidebar.style.top = '0px';

        GshSidebar.style.setProperty('width','200px','important');
        GshSidebar.style.setProperty('overflow','scroll','important');
        GshSidebar.style.resize = 'both';

        GshSidebar.style.left = '-100px';
        GshIndexer.style.left = '100px';
        GshIndexer.style.height = '1400px';
        gsh.appendChild(GshSidebar); // change parent
    }else{
        console.log('-- Seems not in WordPress');
        GshSidebar.style.setProperty('position','fixed','important');
    }
}
//document.addEventListener('load',Indexer_afterLoaded);

DestroyIndexBar = function(){
    sideindex = document.getElementById('GshIndexer');
    sideindex.innerHTML = "";
}
</script>

<!-- Indexer_WorkCodeSpan } -->
*/ //</span>
//<!-- ---------- Work } ---------- -->


/*
<h2>GShell // a General purpose Shell built on the top of Golang</h2>
<p>
<note>
It is a shell for myself, by myself, of myself. --SatoxITS(^-^)
<a href="gsh-0.6.1.go.html">prev.</a>
</note>
</p>
<div id="GJFactory_x"></div>

<div>
<span id="GshMenu">
<span class="GshMenu1" id="GshMenuEdit" onclick="html_edit();">Edit</span>
<span class="GshMenu1" id="GshMenuSave" onclick="html_save();">Save</span>
<span class="GshMenu1" id="GshMenuLoad" onclick="html_load();">Load</span>
<span class="GshMenu1" id="GshMenuVers" onclick="html_ver0();">Vers</span>
<span id="gsh-WinId" onclick="win_jump('0.1');">0</span>
<span class="GshMenu1" id="gsh-menu-exit" onclick="html_close();"></span>
<span class="GshMenu1" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
<span class="GshMenu1" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
<span class="GshMenu1" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
<span class="GshMenu1" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
<span class="GshMenu1" id="GshMenuSign" onclick="html_sign(this);" style="">Source</span>
<!-- | <span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span> -->
</span>
</div>
*/

/*
<details id="GshStatement" class="gsh-document"><summary>Statement</summary>
<h3>Fun to create a shell</h3>
<p>For a programmer, it must be far easy and fun to create his own simple shell
rightly fitting to his favor and necessities, than learning existing shells with
complex full features that he never use.
I, as one of programmers, am writing this tiny shell for my own real needs,
totally from scratch, with fun.
</p><p>
For a programmer, it is fun to learn new computer languages.  For long years before
```

```
writing this software, I had been specialized to C and early HTML2 :-).
Now writing this software,  I'm learning Go language, HTML5, JavaScript and CSS
on demand as a novice of these, with fun.
</p><p>
This single file "gsh.go", that is executable by Go, contains all of the code written
in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
HTML file that works as the viewer of the code of itself, and as the "home page" of
this software.
</p><p>
Because this HTML file is a Go program, you may run it as a real shell program
on your computer.
But you must be aware that this program is written under situation like above.
Needless to say, there is no warranty for this program in any means.
</p>
<address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
</details>
*/
/*
<details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
</p>
<h3>Cross-browser communication</h3>
<p>
... to be written ...
</p>
<h3>Vi compatible command line editor</h3>
<p>
The command line of GShell can be edited with commands compatible with
<a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
As in vi, you can enter <i><b>command mode</b></i> by <b>ESC</b> key,
then move around in the history by <b><code>j k / ? n N</code></b>,
or within the current line by <b><code>l h f w b 0 $ %</code></b> or so.
</p>
</details>
*/
/*
<details id="gsh-gindex">
<summary>Index</summary><div class="gsh-src">
Documents
    <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
    Package structures
        <a href="#import">import</a>
        <a href="#struct">struct</a>
    Main functions
        <a href="#comexpansion">str-expansion</a>   // macro processor
        <a href="#finder">finder</a>        // builtin find + du
        <a href="#grep">grep</a>        // builtin grep + wc + cksum + ...
        <a href="#plugin">plugin</a>        // plugin commands
        <a href="#ex-commands">system</a>        // external commands
        <a href="#builtin">builtin</a>      // builtin commands
        <a href="#network">network</a>      // socket handler
        <a href="#remote-sh">remote-sh</a>  // remote shell
        <a href="#redirect">redirect</a>    // StdIn/Out redireciton
        <a href="#history">history</a>      // command history
        <a href="#rusage">rusage</a>        // resouce usage
        <a href="#encode">encode</a>        // encode / decode
        <a href="#IME">IME</a>      // command line IME
        <a href="#getline">getline</a>      // line editor
        <a href="#scanf">scanf</a>      // string decomposer
        <a href="#interpreter">interpreter</a>  // command interpreter
        <a href="#main">main</a>
</span>
JavaScript part
    <a href="#script-src-view" class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>
    <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
CSS part
    <a href="#style-src-view" class="gsh-link" onclick="jumpto_StyleView();">Source</a>
References
    <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
    <a href="#gsh-reference" class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
Whole parts
    <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Source</a>
    <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Download</a>
    <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Dump</a>

</div>
</details>
*/
//<details id="gsh-gocode">
//<summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
// gsh - Go lang based Shell
// (c) 2020 ITS more Co., Ltd.
// 2020-0807 created by SatoxITS (sato@its-more.jp)

package main // gsh main

// <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
import (
    "fmt"       // <a href="https://golang.org/pkg/fmt/">fmt</a>
    "strings"   // <a href="https://golang.org/pkg/strings/">strings</a>
    "strconv"   // <a href="https://golang.org/pkg/strconv/">strconv</a>
    "sort"      // <a href="https://golang.org/pkg/sort/">sort</a>
    "time"      // <a href="https://golang.org/pkg/time/">time</a>
    "bufio"     // <a href="https://golang.org/pkg/bufio/">bufio</a>
    "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
    "os"        // <a href="https://golang.org/pkg/os/">os</a>
    "syscall"   // <a href="https://golang.org/pkg/syscall/">syscall</a>
    "plugin"    // <a href="https://golang.org/pkg/plugin/">plugin</a>
    "net"       // <a href="https://golang.org/pkg/net/">net</a>
    "net/http"  // <a href="https://golang.org/pkg/net/http/">http</a>
    //"html"    // <a href="https://golang.org/pkg/html/">html</a>
    "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
    "go/types"  // <a href="https://golang.org/pkg/go/types/">types</a>
    "go/token"  // <a href="https://golang.org/pkg/go/token/">token</a>
    "encoding/base64"   // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
    "unicode/utf8"  // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
    //"gshdata" // gshell's logo and source code
    "hash/crc32"    // <a href="https://golang.org/pkg/unicode/hash/crc32/">crc32</a>
    "golang.org/x/net/websocket"
)

// // 2020-0906 added,
// // <a href="https://golang.org/cmd/cgo/">CGo</a>
// #include "poll.h" // <poll.h> // </poll.h> to be closed as HTML tag :-p
// typedef struct { struct pollfd fdv[8]; } pollFdv;
// int pollx(pollFdv *fdv, int nfds, int timeout){
//  return poll(fdv->fdv,nfds,timeout);
// }
import "C"

// // 2020-0906 added,
func CFpollIn1(fp*os.File, timeoutUs int)(ready uintptr){
    var fdv = C.pollFdv{}
    var nfds = 1
    var timeout = timeoutUs/1000

    fdv.fdv[0].fd = C.int(fp.Fd())
    fdv.fdv[0].events = C.POLLIN
    if( 0 < EventRecvFd ){
        fdv.fdv[1].fd = C.int(EventRecvFd)
        fdv.fdv[1].events = C.POLLIN
        nfds += 1
    }
    r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
    if( r <= 0 ){
        return 0
    }
    if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
        //fprintf(stderr,"--De-- got Event\n");
        return uintptr(EventFdOffset + fdv.fdv[1].fd)
    }
    if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
        return uintptr(NormalFdOffset + fdv.fdv[0].fd)
    }
    return 0
}

const (
    NAME = "gsh"
    VERSION = "0.6.2"
    DATE = "2020-10-07"
    AUTHOR = "SatoxITS(^-^)//"
)
var (
    GSH_HOME = ".gsh"   // under home directory
    GSH_PORT = 9999
    MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
    PROMPT = "> "
    LINESIZE = (8*1024)
    PATHSEP = ":"   // should be ";" in Windows
    DIRSEP = "/"    // canbe \ in Windows
)

// -xX logging control
// --A-- all
// --I-- info.
// --D-- debug
// --T-- time and resource usage
// --W-- warning
// --E-- error
// --F-- fatal error
// --Xn- network

// <a name="struct">Structures</a>
type GCommandHistory struct {
    StartAt     time.Time // command line execution started at
    EndAt       time.Time // command line execution ended at
    ResCode     int       // exit code of (external command)
    CmdError    error     // error string
    OutData     *os.File  // output of the command
    FoundFile   []string  // output - result of ufind
    Rusagev     [2]syscall.Rusage // Resource consumption, CPU time or so
    CmdId       int       // maybe with identified with arguments or impact
                      // redireciton commands should not be the CmdId
    WorkDir     string    // working directory at start
    WorkDirX    int       // index in ChdirHistory
    CmdLine     string    // command line
}
type GChdirHistory struct {
    Dir     string
    MovedAt     time.Time
    CmdIndex    int
}
type CmdMode struct {
    BackGround  bool
}
type Event struct {
    when        time.Time
    event       int
    evarg       int64
    CmdIndex    int
}
var CmdIndex int
var Events []Event
type PluginInfo struct {
    Spec        *plugin.Plugin
    Addr        plugin.Symbol
    Name        string // maybe relative
    Path        string // this is in Plugin but hidden
}
type GServer struct {
    host        string
    port        string
}

// <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
const ( // SumType
    SUM_ITEMS   = 0x000001 // items count
    SUM_SIZE    = 0x000002 // data length (simplly added)
    SUM_SIZEHASH    = 0x000004 // data length (hashed sequence)
    SUM_DATEHASH    = 0x000008 // date of data (hashed sequence)
    // also envelope attributes like time stamp can be a part of digest
    // hashed value of sizes or mod-date of files will be useful to detect changes

    SUM_WORDS   = 0x000010 // word count is a kind of digest
    SUM_LINES   = 0x000020 // line count is a kind of digest
    SUM_SUM64   = 0x000040 // simple add of bytes, useful for human too

    SUM_SUM32_BITS  = 0x000100 // the number of true bits
    SUM_SUM32_2BYTE = 0x000200 // 16bits words
    SUM_SUM32_4BYTE = 0x000400 // 32bits words
    SUM_SUM32_8BYTE = 0x000800 // 64bits words

    SUM_SUM16_BSD   = 0x001000 // UNIXsum -sum -bsd
    SUM_SUM16_SYSV  = 0x002000 // UNIXsum -sum -sysv
    SUM_UNIXFILE    = 0x004000
    SUM_CRCIEEE = 0x008000
)
type CheckSum struct {
    Files       int64   // the number of files (or data)
    Size        int64   // content size
    Words       int64   // word count
    Lines       int64   // line count
    SumType     int
    Sum64       uint64
    Crc32Table  crc32.Table
    Crc32Val    uint32
    Sum16       int
```

```
	Ctime       time.Time
	Atime       time.Time
	Mtime       time.Time
	Start       time.Time
	Done        time.Time
	RusgAtStart [2]syscall.Rusage
	RusgAtEnd   [2]syscall.Rusage
}
type ValueStack [][]string
type GshContext struct {
	StartDir    string  // the current directory at the start
	GetLine     string  // gsh-getline command as a input line editor
	ChdirHistory    []GChdirHistory // the 1st entry is wd at the start
	gshPA       syscall.ProcAttr
	CommandHistory  []GCommandHistory
	CmdCurrent  GCommandHistory
	BackGround  bool
	BackGroundJobs  []int
	LastRusage  syscall.Rusage
	GshHomeDir  string
	TerminalId  int
	CmdTrace    bool // should be [map]
	CmdTime     bool // should be [map]
	PluginFuncs []PluginInfo
	iValues     []string
	iDelimiter  string // field sepearater of print out
	iFormat     string // default print format (of integer)
	iValStack   ValueStack
	LastServer  GServer
	RSERV       string // [gsh://]host[:port]
	RWD     string // remote (target, there) working directory
	lastCheckSum    CheckSum
}

func nsleep(ns time.Duration){
	time.Sleep(ns)
}
func usleep(ns time.Duration){
	nsleep(ns*1000)
}
func msleep(ns time.Duration){
	nsleep(ns*1000000)
}
func sleep(ns time.Duration){
	nsleep(ns*1000000000)
}

func strBegins(str, pat string)(bool){
	if len(pat) <= len(str){
		yes := str[0:len(pat)] == pat
		//fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
		return yes
	}
	//fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
	return false
}
func isin(what string, list []string) bool {
	for _, v := range list  {
		if v == what {
			return true
		}
	}
	return false
}
func isinX(what string,list[]string)(int){
	for i,v := range list {
		if v == what {
			return i
		}
	}
	return -1
}

func env(opts []string) {
	env := os.Environ()
	if isin("-s", opts){
		sort.Slice(env, func(i,j int) bool {
			return env[i] < env[j]
		})
	}
	for _, v := range env {
		fmt.Printf("%v\n",v)
	}
}

// - rewriting should be context dependent
// - should postpone until the real point of evaluation
// - should rewrite only known notation of symobl
func scanInt(str string)(val int,leng int){
	leng = -1
	for i,ch := range str {
		if '0' <= ch && ch <= '9' {
			leng = i+1
		}else{
			break
		}
	}
	if 0 < leng {
		ival,_ := strconv.Atoi(str[0:leng])
		return ival,leng
	}else{
		return 0,0
	}
}
func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
	if len(str[i+1:]) == 0 {
		return 0,rstr
	}
	hi := 0
	histlen := len(gshCtx.CommandHistory)
	if str[i+1] == '!' {
		hi = histlen - 1
		leng = 1
	}else{
		hi,leng = scanInt(str[i+1:])
		if leng == 0 {
			return 0,rstr
		}
		if hi < 0 {
			hi = histlen + hi
		}
	}
	if 0 <= hi && hi < histlen {
		var ext byte
```

```
        if 1 < len(str[i+leng:]) {
            ext = str[i+leng:][1]
        }
        //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
        if ext == 'f' {
            leng += 1
            xlist := []string{}
            list := gshCtx.CommandHistory[hi].FoundFile
            for _,v := range list {
                //list[i] = escapeWhiteSP(v)
                xlist = append(xlist,escapeWhiteSP(v))
            }
            //rstr += strings.Join(list," ")
            rstr += strings.Join(xlist," ")
        }else
        if ext == '@' || ext == 'd' {
            // !N@ .. workdir at the start of the command
            leng += 1
            rstr += gshCtx.CommandHistory[hi].WorkDir
        }else{
            rstr += gshCtx.CommandHistory[hi].CmdLine
        }
    }else{
        leng = 0
    }
    return leng,rstr
}
func escapeWhiteSP(str string)(string){
    if len(str) == 0 {
        return "\\z" // empty, to be ignored
    }
    rstr := ""
    for _,ch := range str {
        switch ch {
            case '\\': rstr += "\\\\"
            case ' ': rstr += "\\s"
            case '\t': rstr += "\\t"
            case '\r': rstr += "\\r"
            case '\n': rstr += "\\n"
            default: rstr += string(ch)
        }
    }
    return rstr
}
func unescapeWhiteSP(str string)(string){ // strip original escapes
    rstr := ""
    for i := 0; i < len(str); i++ {
        ch := str[i]
        if ch == '\\' {
            if i+1 < len(str) {
                switch str[i+1] {
                    case 'z':
                        continue;
                }
            }
        }
        rstr += string(ch)
    }
    return rstr
}
func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
    ustrv := []string{}
    for _,v := range strv {
        ustrv = append(ustrv,unescapeWhiteSP(v))
    }
    return ustrv
}

// <a name="comexpansion">str-expansion</a>
// - this should be a macro processor
func strsubst(gshCtx *GshContext,str string,histonly bool) string {
    rbuff := []byte{}
    if false {
        //@@U Unicode should be cared as a character
        return str
    }
    //rstr := ""
    inEsc := 0 // escape characer mode
    for i := 0; i < len(str); i++ {
        //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
        ch := str[i]
        if inEsc == 0 {
            if ch == '!' {
                //leng,xrstr := substHistory(gshCtx,str,i,rstr)
                leng,rs := substHistory(gshCtx,str,i,"")
                if 0 < leng {
    //_,rs := substHistory(gshCtx,str,i,"")
    rbuff = append(rbuff,[]byte(rs)...)
                    i += leng
                    //rstr = xrstr
                    continue
                }
            }
            switch ch {
                case '\\': inEsc = '\\'; continue
                //case '%':  inEsc = '%';  continue
                case '$':
            }
        }
        switch inEsc {
        case '\\':
            switch ch {
                case '\\': ch = '\\'
                case 's': ch = ' '
                case 't': ch = '\t'
                case 'r': ch = '\r'
                case 'n': ch = '\n'
                case 'z': inEsc = 0; continue // empty, to be ignored
            }
            inEsc = 0
        case '%':
            switch {
                case ch == '%': ch = '%'
                case ch == 'T':
                    //rstr = rstr + time.Now().Format(time.Stamp)
    rs := time.Now().Format(time.Stamp)
    rbuff = append(rbuff,[]byte(rs)...)
                    inEsc = 0
                    continue;
                default:
                    // postpone the interpretation
                    //rstr = rstr + "%" + string(ch)
    rbuff = append(rbuff,ch)
                    inEsc = 0
                    continue;
            }
            inEsc = 0
        }
        //rstr = rstr + string(ch)
        rbuff = append(rbuff,ch)
    }
    //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
    return string(rbuff)
    //return rstr
}
func showFileInfo(path string, opts []string) {
    if isin("-l",opts) || isin("-ls",opts) {
        fi, err := os.Stat(path)
        if err != nil {
            fmt.Printf("---------- ((%v))",err)
        }else{
            mod := fi.ModTime()
            date := mod.Format(time.Stamp)
            fmt.Printf("%v %8v %s ",fi.Mode(),fi.Size(),date)
        }
    }
    fmt.Printf("%s",path)
    if isin("-sp",opts) {
        fmt.Printf(" ")
    }else
    if ! isin("-n",opts) {
        fmt.Printf("\n")
    }
}
func userHomeDir()(string,bool){
    /*
    homedir,_ = os.UserHomeDir() // not implemented in older Golang
    */
    homedir,found := os.LookupEnv("HOME")
    //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
    if !found {
        return "/tmp",found
    }
    return homedir,found
}

func toFullpath(path string) (fullpath string) {
    if path[0] == '/' {
        return path
    }
    pathv := strings.Split(path,DIRSEP)
    switch {
    case pathv[0] == ".":
        pathv[0], _ = os.Getwd()
    case pathv[0] == "..": // all ones should be interpreted
        cwd, _ := os.Getwd()
        ppathv := strings.Split(cwd,DIRSEP)
        pathv[0] = strings.Join(ppathv,DIRSEP)
    case pathv[0] == "~":
        pathv[0],_ = userHomeDir()
    default:
        cwd, _ := os.Getwd()
        pathv[0] = cwd + DIRSEP + pathv[0]
    }
    return strings.Join(pathv,DIRSEP)
}

func IsRegFile(path string)(bool){
    fi, err := os.Stat(path)
    if err == nil {
        fm := fi.Mode()
        return fm.IsRegular();
    }
    return false
}

// <a name="encode">Encode / Decode</a>
// <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
func (gshCtx *GshContext)Enc(argv[]string){
    file := os.Stdin
    buff := make([]byte,LINESIZE)
    li := 0
    encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
    for li = 0; ; li++ {
        count, err := file.Read(buff)
        if count <= 0 {
            break
        }
        if err != nil {
            break
        }
        encoder.Write(buff[0:count])
    }
    encoder.Close()
}
func (gshCtx *GshContext)Dec(argv[]string){
    decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
    li := 0
    buff := make([]byte,LINESIZE)
    for li = 0; ; li++ {
        count, err := decoder.Read(buff)
        if count <= 0 {
            break
        }
        if err != nil {
            break
        }
        os.Stdout.Write(buff[0:count])
    }
}
// lnsp [N] [-crlf][-C \\]
func (gshCtx *GshContext)SplitLine(argv[]string){
    strRep := isin("-str",argv) // "..."+
    reader := bufio.NewReaderSize(os.Stdin,64*1024)
    ni := 0
    toi := 0
    for ni = 0; ; ni++ {
        line, err := reader.ReadString('\n')
        if len(line) <= 0 {
            if err != nil {
            fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d (%v)\n",ni,toi,err)
            break
            }
        }
        off := 0
        ilen := len(line)
        remlen := len(line)
        if strRep { os.Stdout.Write([]byte("\"")) }
        for oi := 0; 0 < remlen; oi++ {
            olen := remlen
            addnl := false
            if 72 < olen {
                olen = 72
                addnl = true
            }
            fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d/%d\n",
                toi,ni,oi,off,olen,remlen,ilen)
            toi += 1
            os.Stdout.Write([]byte(line[0:olen]))
            if addnl {
                if strRep {
                    os.Stdout.Write([]byte("\"+\n\""))
                }else{
                    //os.Stdout.Write([]byte("\r\n"))
                    os.Stdout.Write([]byte("\\"))
                    os.Stdout.Write([]byte("\n"))
                }
            }
            line = line[olen:]
            off += olen
            remlen -= olen
        }
        if strRep { os.Stdout.Write([]byte("\"\n")) }
    }
    fmt.Fprintf(os.Stderr,"--I-- lnsp %d to %d\n",ni,toi)
}

// CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
// 1 0000 0100 1100 0001 0001 1101 1011 0111
var CRC32UNIX uint32 = uint32(0x04C11DB7) // Unix cksum
var CRC32IEEE uint32 = uint32(0xEDB88320)
func byteCRC32add(crc uint32,str[]byte,len uint64)(uint32){
    var oi uint64
    for oi = 0; oi < len; oi++ {
        var oct = str[oi]
        for bi := 0; bi < 8; bi++ {
            //fprintf(stderr,"--CRC32 %d %X (%d.%d)\n",crc,oct,oi,bi)
            ovf1 := (crc & 0x80000000) != 0
            ovf2 := (oct & 0x80) != 0
            ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
            oct <<= 1
            crc <<= 1
            if ovf { crc ^= CRC32UNIX }
        }
    }
    //fprintf(stderr,"--CRC32 return %d %d\n",crc,len)
    return crc;
}
func byteCRC32end(crc uint32, len uint64)(uint32){
    var slen = make([]byte,4)
    var li = 0
        for li = 0; li < 4; {
                slen[li] = byte(len)
        li += 1
                len >>= 8
                if( len == 0 ){
                        break
        }
        }
        crc = byteCRC32add(crc,slen,uint64(li))
        crc ^= 0xFFFFFFFF
        return crc
}
func strCRC32(str string,len uint64)(crc uint32){
    crc = byteCRC32add(0,[]byte(str),len)
    crc = byteCRC32end(crc,len)
    //fprintf(stderr,"--CRC32 %d %d\n",crc,len)
    return crc
}
func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
    var slen = make([]byte,4)
    var li = 0
        for li = 0; li < 4; {
                slen[li] = byte(len & 0xFF)
        li += 1
                len >>= 8
                if( len == 0 ){
                        break
        }
        }
    crc = crc32.Update(crc,table,slen)
        crc ^= 0xFFFFFFFF
        return crc
}

func (gsh*GshContext)xCksum(path string,argv[]string, sum*CheckSum)(int64){
    if isin("-type/f",argv) && !IsRegFile(path){
        return 0
    }
    if isin("-type/d",argv) && IsRegFile(path){
        return 0
    }
    file, err := os.OpenFile(path,os.O_RDONLY,0)
    if err != nil {
        fmt.Printf("--E-- cksum %v (%v)\n",path,err)
        return -1
    }
    defer file.Close()
    if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n",path,argv) }

    bi := 0
    var buff = make([]byte,32*1024)
    var total int64 = 0
    var initTime = time.Time{}
    if sum.Start == initTime {
        sum.Start = time.Now()
    }
    for bi = 0; ; bi++ {
        count,err := file.Read(buff)
        if count <= 0 || err != nil {
            break
        }
        if (sum.SumType & SUM_SUM64) != 0 {
            s := sum.Sum64
            for _,c := range buff[0:count] {
                s += uint64(c)
            }
            sum.Sum64 = s
        }
        if (sum.SumType & SUM_UNIXFILE) != 0 {
            sum.Crc32Val = byteCRC32add(sum.Crc32Val,buff,uint64(count))
        }
        if (sum.SumType & SUM_CRCIEEE) != 0 {
            sum.Crc32Val = crc32.Update(sum.Crc32Val,&sum.Crc32Table,buff[0:count])
        }
        // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
        if (sum.SumType & SUM_SUM16_BSD) != 0 {
            s := sum.Sum16
            for _,c := range buff[0:count] {
                s = (s >> 1) + ((s & 1) << 15)
                s += int(c)
                s &= 0xFFFF
                //fmt.Printf("BSDsum: %d[%d] %d\n",sum.Size+int64(i),i,s)
            }
            sum.Sum16 = s
        }
        if (sum.SumType & SUM_SUM16_SYSV) != 0 {
            for bj := 0; bj < count; bj++ {
                sum.Sum16 += int(buff[bj])
            }
        }
        total += int64(count)
    }
    sum.Done = time.Now()
    sum.Files += 1
    sum.Size += total
    if !isin("-s",argv) {
        fmt.Printf("%v ",total)
    }
    return 0
}

// <a name="grep">grep</a>
// "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
// a*,!ab,c, ... sequentioal combination of patterns
// what "LINE" is should be definable
// generic line-by-line processing
// grep [-v]
// cat -n -v
// uniq [-c]
// tail -f
// sed s/x/y/ or awk
// grep with line count like wc
// rewrite contents if specified
func (gsh*GshContext)xGrep(path string,rexpv[]string)(int){
    file, err := os.OpenFile(path,os.O_RDONLY,0)
    if err != nil {
        fmt.Printf("--E-- grep %v (%v)\n",path,err)
        return -1
    }
    defer file.Close()
    if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path,rexpv) }
    //reader := bufio.NewReaderSize(file,LINESIZE)
    reader := bufio.NewReaderSize(file,80)
    li := 0
    found := 0
    for li = 0; ; li++ {
        line, err := reader.ReadString('\n')
        if len(line) <= 0 {
            break
        }
        if 150 < len(line) {
            // maybe binary
            break;
        }
        if err != nil {
            break
        }
        if 0 <= strings.Index(string(line),rexpv[0]) {
            found += 1
            fmt.Printf("%s:%d: %s",path,li,line)
        }
    }
        //fmt.Printf("total %d lines %s\n",li,path)
    //if( 0 < found ){ fmt.Printf("((found %d lines %s))\n",found,path); }
    return found
}

// <a name="finder">Finder</a>
// finding files with it name and contents
// file names are ORed
// show the content with %x fmt list
// ls -R
// tar command by adding output
type fileSum struct {
    Err int64    // access error or so
    Size    int64   // content size
    DupSize int64   // content size from hard links
    Blocks  int64   // number of blocks (of 512 bytes)
    DupBlocks int64 // Blocks pointed from hard links
    HLinks  int64   // hard links
    Words   int64
    Lines   int64
    Files   int64
    Dirs    int64   // the num. of directories
    SymLink int64
    Flats   int64   // the num. of flat files
    MaxDepth    int64
    MaxNamlen   int64   // max. name length
    nextRepo    time.Time
}
func showFusage(dir string,fusage *fileSum){
    bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
    //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0

    fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
        dir,
        fusage.Files,
        fusage.Dirs,
        fusage.SymLink,
        fusage.HLinks,
        float64(fusage.Size)/1000000.0,bsume);
}
const (
    S_IFMT    = 0170000
    S_IFCHR   = 0020000
    S_IFDIR   = 0040000
    S_IFREG   = 0100000
    S_IFLNK   = 0120000
    S_IFSOCK  = 0140000
)
func cumFinfo(fsum *fileSum, path string, staterr error, fstat syscall.Stat_t, argv[]string,verb bool)(*fileSum){
    now := time.Now()
    if time.Second <= now.Sub(fsum.nextRepo) {
        if !fsum.nextRepo.IsZero(){
            tstmp := now.Format(time.Stamp)
```

```go
            showFusage(tstmp,fsum)
        }
        fsum.nextRepo = now.Add(time.Second)
    }
    if staterr != nil {
        fsum.Err += 1
        return fsum
    }
    fsum.Files += 1
    if 1 < fstat.Nlink {
        // must count only once...
        // at least ignore ones in the same directory
        //if finfo.Mode().IsRegular() {
        if (fstat.Mode & S_IFMT) == S_IFREG {
            fsum.HLinks += 1
            fsum.DupBlocks += int64(fstat.Blocks)
            //fmt.Printf("---Dup HardLink %v %s\n",fstat.Nlink,path)
        }
    }
    //fsum.Size += finfo.Size()
    fsum.Size += fstat.Size
    fsum.Blocks += int64(fstat.Blocks)
    //if verb { fmt.Printf("(%8dBlk) %s",fstat.Blocks/2,path) }
    if isin("-ls",argv){
        //if verb { fmt.Printf("%4d %8d ",fstat.Blksize,fstat.Blocks) }
//      fmt.Printf("%d\t",fstat.Blocks/2)
    }
    //if finfo.IsDir()
    if (fstat.Mode & S_IFMT) == S_IFDIR {
        fsum.Dirs += 1
    }
    //if (finfo.Mode() & os.ModeSymlink) != 0
    if (fstat.Mode & S_IFMT) == S_IFLNK {
        //if verb { fmt.Printf("symlink(%v,%s)\n",fstat.Mode,finfo.Name()) }
        //{ fmt.Printf("symlink(%o,%s)\n",fstat.Mode,finfo.Name()) }
        fsum.SymLink += 1
    }
    return fsum
}
func (gsh*GshContext)xxFindEntv(depth int,total *fileSum,dir string, dstat syscall.Stat_t, ei int, entv []string,npatv[]string,argv[]string)(*fileSum){
    nols := isin("-grep",argv)
    // sort entv
    /*
    if isin("-t",argv){
        sort.Slice(filev, func(i,j int) bool {
            return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
        })
    }
    */
        /*
        if isin("-u",argv){
            sort.Slice(filev, func(i,j int) bool {
                return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
            })
        }
        if isin("-U",argv){
            sort.Slice(filev, func(i,j int) bool {
                return 0 < filev[i].CreatTime().Sub(filev[j].CreatTime())
            })
        }
        */
    /*
    if isin("-S",argv){
        sort.Slice(filev, func(i,j int) bool {
            return filev[j].Size() < filev[i].Size()
        })
    }
    */
    for _,filename := range entv {
        for _,npat := range npatv {
            match := true
            if npat == "*" {
                match = true
            }else{
                match, _ = filepath.Match(npat,filename)
            }
            path := dir + DIRSEP + filename
            if !match {
                continue
            }
            var fstat syscall.Stat_t
            staterr := syscall.Lstat(path,&fstat)
            if staterr != nil {
                if !isin("-w",argv){fmt.Printf("ufind: %v\n",staterr) }
                continue;
            }
            if isin("-du",argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
                // should not show size of directory in "-du" mode ...
            }else
            if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
                if isin("-du",argv) {
                    fmt.Printf("%d\t",fstat.Blocks/2)
                }
                showFileInfo(path,argv)
            }
            if true { // && isin("-du",argv)
                total = cumFinfo(total,path,staterr,fstat,argv,false)
            }
            /*
            if isin("-wc",argv) {
            }
            */
            if gsh.lastCheckSum.SumType != 0 {
                gsh.xCksum(path,argv,&gsh.lastCheckSum);
            }
            x := isinX("-grep",argv); // -grep will be convenient like -ls
            if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
                if IsRegFile(path){
                    found := gsh.xGrep(path,argv[x+1:])
                    if 0 < found {
                        foundv := gsh.CmdCurrent.FoundFile
                        if len(foundv) < 10 {
                            gsh.CmdCurrent.FoundFile =
                            append(gsh.CmdCurrent.FoundFile,path)
                        }
                    }
                }
            }
            if !isin("-r0",argv) { // -d 0 in du, -depth n in find
                //total.Depth += 1
                if (fstat.Mode & S_IFMT) == S_IFLNK {
                    continue
                }
                if dstat.Rdev != fstat.Rdev {
                    fmt.Printf("--I-- don't follow differnet device %v(%v) %v(%v)\n",
                        dir,dstat.Rdev,path,fstat.Rdev)
                }
                if (fstat.Mode & S_IFMT) == S_IFDIR {
                    total = gsh.xxFind(depth+1,total,path,npatv,argv)
                }
            }
        }
    }
    return total
}
func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum){
    nols := isin("-grep",argv)
    dirfile,oerr := os.OpenFile(dir,os.O_RDONLY,0)
    if oerr == nil {
        //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
        defer dirfile.Close()
    }else{
    }

    prev := *total
    var dstat syscall.Stat_t
    staterr := syscall.Lstat(dir,&dstat) // should be flstat

    if staterr != nil {
        if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
        return total
    }
        //filev,err := ioutil.ReadDir(dir)
        //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
        /*
        if err != nil {
            if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
            return total
        }
        */
    if depth == 0 {
        total = cumFinfo(total,dir,staterr,dstat,argv,true)
        if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
            showFileInfo(dir,argv)
        }
    }
    // it it is not a directory, just scan it and finish

    for ei := 0; ; ei++ {
        entv,rderr := dirfile.Readdirnames(8*1024)
        if len(entv) == 0 || rderr != nil {
            //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rderr) }
            break
        }
        if 0 < ei {
            fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
        }
        total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
    }
    if isin("-du",argv) {
        // if in "du" mode
        fmt.Printf("%d\t%s\n",(total.Blocks-prev.Blocks)/2,dir)
    }
    return total
}

// {ufind|fu|ls} [Files] [// Names] [-- Expressions]
//  Files is "." by default
//  Names is "*" by default
//  Expressions is "-print" by default for "ufind", or -du for "fu" command
func (gsh*GshContext)xFind(argv[]string){
    if 0 < len(argv) && strBegins(argv[0],"?"){
        showFound(gsh,argv)
        return
    }
    if isin("-cksum",argv) || isin("-sum",argv) {
        gsh.lastCheckSum = CheckSum{}
        if isin("-sum",argv) && isin("-add",argv) {
            gsh.lastCheckSum.SumType |= SUM_SUM64
        }else
        if isin("-sum",argv) && isin("-size",argv) {
            gsh.lastCheckSum.SumType |= SUM_SIZE
        }else
        if isin("-sum",argv) && isin("-bsd",argv) {
            gsh.lastCheckSum.SumType |= SUM_SUM16_BSD
        }else
        if isin("-sum",argv) && isin("-sysv",argv) {
            gsh.lastCheckSum.SumType |= SUM_SUM16_SYSV
        }else
        if isin("-sum",argv) {
            gsh.lastCheckSum.SumType |= SUM_SUM64
        }
        if isin("-unix",argv) {
            gsh.lastCheckSum.SumType |= SUM_UNIXFILE
            gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
        }
        if isin("-ieee",argv){
            gsh.lastCheckSum.SumType |= SUM_CRCIEEE
            gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEE)
        }
        gsh.lastCheckSum.RusgAtStart = Getrusagev()
    }
    var total = fileSum{}
    npats := []string{}
    for _,v := range argv {
        if 0 < len(v) && v[0] != '-' {
            npats = append(npats,v)
        }
        if v == "//" { break }
        if v == "--" { break }
        if v == "-grep" { break }
        if v == "-ls" { break }
    }
    if len(npats) == 0 {
        npats = []string{"*"}
    }
    cwd := "."
    // if to be fullpath ::: cwd, _ := os.Getwd()
    if len(npats) == 0 { npats = []string{"*"} }
    fusage := gsh.xxFind(0,&total,cwd,npats,argv)
    if gsh.lastCheckSum.SumType != 0 {
        var sumi uint64 = 0
        sum := &gsh.lastCheckSum
        if (sum.SumType & SUM_SIZE) != 0 {
            sumi = uint64(sum.Size)
        }
        if (sum.SumType & SUM_SUM64) != 0 {
            sumi = sum.Sum64
        }
        if (sum.SumType & SUM_SUM16_SYSV) != 0 {
            s := uint32(sum.Sum16)
            r := (s & 0xFFFF) + ((s & 0xFFFFFFFF) >> 16)
            s = (r & 0xFFFF) + (r >> 16)
            sum.Crc32Val = uint32(s)
            sumi = uint64(s)
        }
        if (sum.SumType & SUM_SUM16_BSD) != 0 {
            sum.Crc32Val = uint32(sum.Sum16)
            sumi = uint64(sum.Sum16)
        }
        if (sum.SumType & SUM_UNIXFILE) != 0 {
            sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
            sumi = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
        }
        if 1 < sum.Files {
            fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
                sumi,sum.Size,
                abssize(sum.Size),sum.Files,
                abssize(sum.Size/sum.Files))
        }else{
            fmt.Printf("%v %v %v\n",
                sumi,sum.Size,npats[0])
        }
    }
    if !isin("-grep",argv) {
        showFusage("total",fusage)
    }
    if !isin("-s",argv){
        hits := len(gsh.CmdCurrent.FoundFile)
        if 0 < hits {
            fmt.Printf("--I-- %d files hits // can be refered with !%df\n",
                hits,len(gsh.CommandHistory))
        }
    }
    if gsh.lastCheckSum.SumType != 0 {
        if isin("-ru",argv) {
            sum := &gsh.lastCheckSum
            sum.Done = time.Now()
            gsh.lastCheckSum.RusgAtEnd = Getrusagev()
            elps := sum.Done.Sub(sum.Start)
            fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
                sum.Size,abssize(sum.Size),sum.Files,abssize(sum.Size/sum.Files))
            nanos := int64(elps)
            fmt.Printf("--cksum-time: %v/total, %v/file, %.1f files/s, %v\r\n",
                abbtime(nanos),
                abbtime(nanos/sum.Files),
                (float64(sum.Files)*1000000000.0)/float64(nanos),
                abbspeed(sum.Size,nanos))
            diff := RusageSubv(sum.RusgAtEnd,sum.RusgAtStart)
            fmt.Printf("--cksum-rusg: %v\n",sRusagef("",argv,diff))
        }
    }
    return
}

func showFiles(files[]string){
    sp := ""
    for i,file := range files {
        if 0 < i { sp = " " } else { sp = "" }
        fmt.Printf(sp+"%s",escapeWhiteSP(file))
    }
}
func showFound(gshCtx *GshContext, argv[]string){
    for i,v := range gshCtx.CommandHistory {
        if 0 < len(v.FoundFile) {
            fmt.Printf("!%d (%d) ",i,len(v.FoundFile))
            if isin("-ls",argv){
                fmt.Printf("\n")
                for _,file := range v.FoundFile {
                    fmt.Printf("") //sub number?
                    showFileInfo(file,argv)
                }
            }else{
                showFiles(v.FoundFile)
                fmt.Printf("\n")
            }
        }
    }
}

func showMatchFile(filev []os.FileInfo, npat,dir string, argv[]string)(string,bool){
    fname := ""
    found := false
    for _,v := range filev {
        match, _ := filepath.Match(npat,(v.Name()))
        if match {
            fname = v.Name()
            found = true
            //fmt.Printf("[%d] %s\n",i,v.Name())
            showIfExecutable(fname,dir,argv)
        }
    }
    return fname,found
}
func showIfExecutable(name,dir string,argv[]string)(ffullpath string,ffound bool){
    var fullpath string
    if strBegins(name,DIRSEP){
        fullpath = name
    }else{
        fullpath = dir + DIRSEP + name
    }
    fi, err := os.Stat(fullpath)
    if err != nil {
        fullpath = dir + DIRSEP + name + ".go"
        fi, err = os.Stat(fullpath)
    }
    if err == nil {
        fm := fi.Mode()
        if fm.IsRegular() {
          // R_OK=4, W_OK=2, X_OK=1, F_OK=0
          if syscall.Access(fullpath,5) == nil {
            ffullpath = fullpath
            ffound = true
            if ! isin("-s", argv) {
                showFileInfo(fullpath,argv)
            }
          }
        }
    }
    return ffullpath, ffound
}
func which(list string, argv []string) (fullpathv []string, itis bool){
```

```go
    if len(argv) <= 1 {
        fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
        return []string{""}, false
    }
    path := argv[1]
    if strBegins(path,"/") {
        // should check if excecutable?
        _,exOK := showIfExecutable(path,"/",argv)
        fmt.Printf("--D-- %v exOK=%v\n",path,exOK)
        return []string{path},exOK
    }
    pathenv, efound := os.LookupEnv(list)
    if ! efound {
        fmt.Printf("--E-- which: no \"%s\" environment\n",list)
        return []string{""}, false
    }
    showall := isin("-a",argv) || 0 <= strings.Index(path,"*")
    dirv := strings.Split(pathenv,PATHSEP)
    ffound := false
    ffullpath := path
    for _, dir := range dirv {
        if 0 <= strings.Index(path,"*") { // by wild-card
            list,_ := ioutil.ReadDir(dir)
            ffullpath, ffound = showMatchFile(list,path,dir,argv)
        }else{
            ffullpath, ffound = showIfExecutable(path,dir,argv)
        }
        //if ffound && !isin("-a", argv) {
        if ffound && !showall {
            break;
        }
    }
    return []string{ffullpath}, ffound
}

func stripLeadingWSParg(argv[]string)([]string){
    for ; 0 < len(argv); {
        if len(argv[0]) == 0 {
            argv = argv[1:]
        }else{
            break
        }
    }
    return argv
}
func xEval(argv []string, nlend bool){
    argv = stripLeadingWSParg(argv)
    if len(argv) == 0 {
        fmt.Printf("eval [%%format] [Go-expression]\n")
        return
    }
    pfmt := "%v"
    if argv[0][0] == '%' {
        pfmt = argv[0]
        argv = argv[1:]
    }
    if len(argv) == 0 {
        return
    }
    gocode := strings.Join(argv," ");
    //fmt.Printf("eval [%v] [%v]\n",pfmt,gocode)
    fset := token.NewFileSet()
    rval, _ := types.Eval(fset,nil,token.NoPos,gocode)
    fmt.Printf(pfmt,rval.Value)
    if nlend { fmt.Printf("\n") }
}

func getval(name string) (found bool, val int) {
    /* should expand the name here */
    if name == "gsh.pid" {
        return true, os.Getpid()
    }else
    if name == "gsh.ppid" {
        return true, os.Getppid()
    }
    return false, 0
}

func echo(argv []string, nlend bool){
    for ai := 1; ai < len(argv); ai++ {
        if 1 < ai {
            fmt.Printf(" ");
        }
        arg := argv[ai]
        found, val := getval(arg)
        if found {
            fmt.Printf("%d",val)
        }else{
            fmt.Printf("%s",arg)
        }
    }
    if nlend {
        fmt.Printf("\n");
    }
}

func resfile() string {
    return "gsh.tmp"
}
//var resF *File
func resmap() {
    //_ , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
    // https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
    _ , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
    if err != nil {
        fmt.Printf("refF could not open: %s\n",err)
    }else{
        fmt.Printf("refF opened\n")
    }
}

// @@2020-0821
func gshScanArg(str string,strip int)(argv []string){
    var si = 0
    var sb = 0
    var inBracket = 0
    var arg1 = make([]byte,LINESIZE)
    var ax = 0
    debug := false

    for ; si < len(str); si++ {
        if str[si] != ' ' {
            break
        }
    }
    sb = si
    for ; si < len(str); si++ {
        if sb <= si {
            if debug {
                fmt.Printf("--Da- +%d %2d-%2d %s ... %s\n",
                    inBracket,sb,si,arg1[0:ax],str[si:])
            }
        }
        ch := str[si]
        if ch  == '{' {
            inBracket += 1
            if 0 < strip && inBracket <= strip {
                //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
                continue
            }
        }
        if 0 < inBracket {
            if ch == '}' {
                inBracket -= 1
                if 0 < strip && inBracket < strip {
                    //fmt.Printf("stripLEV %d <  %d?\n",inBracket,strip)
                    continue
                }
            }
            arg1[ax] = ch
            ax += 1
            continue
        }
        if str[si] == ' ' {
            argv = append(argv,string(arg1[0:ax]))
            if debug {
                fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
                    -1+len(argv),sb,si,str[sb:si],string(str[si:]))
            }
            sb = si+1
            ax = 0
            continue
        }
        arg1[ax] = ch
        ax += 1
    }
    if sb < si {
        argv = append(argv,string(arg1[0:ax]))
        if debug {
            fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
                -1+len(argv),sb,si,string(arg1[0:ax]),string(str[si:]))
        }
    }
    if debug {
        fmt.Printf("--Da- %d [%s] => [%d]%v\n",strip,str,len(argv),argv)
    }
    return argv
}

// should get stderr (into tmpfile ?) and return
func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
    var pv = []int{-1,-1}
    syscall.Pipe(pv)

    xarg := gshScanArg(name,1)
    name = strings.Join(xarg," ")

    pin = os.NewFile(uintptr(pv[0]),"StdoutOf-{"+name+"}")
    pout = os.NewFile(uintptr(pv[1]),"StdinOf-{"+name+"}")
    fdix := 0
    dir := "?"
    if mode == "r" {
        dir = "<"
        fdix = 1 // read from the stdout of the process
    }else{
        dir = ">"
        fdix = 0 // write to the stdin of the process
    }
    gshPA := gsh.gshPA
    savfd := gshPA.Files[fdix]

    var fd uintptr = 0
    if mode == "r" {
        fd = pout.Fd()
        gshPA.Files[fdix] = pout.Fd()
    }else{
        fd = pin.Fd()
        gshPA.Files[fdix] = pin.Fd()
    }
        // should do this by Goroutine?
        if false {
            fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
            fmt.Printf("--RED1 [%d,%d,%d]->[%d,%d,%d]\n",
                os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
                pin.Fd(),pout.Fd(),pout.Fd())
        }
            savi := os.Stdin
            savo := os.Stdout
            save := os.Stderr
            os.Stdin  = pin
            os.Stdout = pout
            os.Stderr = pout
        gsh.BackGround = true
        gsh.gshelllh(name)
        gsh.BackGround = false
            os.Stdin  = savi
            os.Stdout = savo
            os.Stderr = save

    gshPA.Files[fdix] = savfd
    return pin,pout,false
}

// <a name="ex-commands">External commands</a>
func (gsh*GshContext)excommand(exec bool, argv []string) (notf bool,exit bool) {
    if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }

    gshPA := gsh.gshPA
    fullpathv, itis := which("PATH",[]string{"which",argv[0],"-s"})
    if itis == false {
        return true,false
    }
    fullpath := fullpathv[0]
    argv = unescapeWhiteSPV(argv)
    if 0 < strings.Index(fullpath,".go") {
        nargv := argv // []string{}
        gofullpathv, itis := which("PATH",[]string{"which","go","-s"})
        if itis == false {
            fmt.Printf("--F-- Go not found\n")
            return false,true
        }
        gofullpath := gofullpathv[0]
        nargv = []string{ gofullpath, "run", fullpath }
        fmt.Printf("--I-- %s {%s %s %s}\n",gofullpath,
            nargv[0],nargv[1],nargv[2])
        if exec {
            syscall.Exec(gofullpath,nargv,os.Environ())
        }else{
            pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
            if gsh.BackGround {
                fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%d(%v)\n",pid,len(argv),nargv)
                gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
            }else{
                rusage := syscall.Rusage {}
                syscall.Wait4(pid,nil,0,&rusage)
                gsh.LastRusage = rusage
                gsh.CmdCurrent.Rusagev[1] = rusage
            }
        }
    }else{
        if exec {
            syscall.Exec(fullpath,argv,os.Environ())
        }else{
            pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
            //fmt.Printf("[%d]\n",pid); // '&' to be background
            if gsh.BackGround {
                fmt.Fprintf(stderr,"--Ip- in Background pid[%d]%d(%v)\n",pid,len(argv),argv)
                gsh.BackGroundJobs = append(gsh.BackGroundJobs,pid)
            }else{
                rusage := syscall.Rusage {}
                syscall.Wait4(pid,nil,0,&rusage);
                gsh.LastRusage = rusage
                gsh.CmdCurrent.Rusagev[1] = rusage
            }
        }
    }
    return false,false
}

// <a name="builtin">Builtin Commands</a>
func (gshCtx *GshContext) sleep(argv []string) {
    if len(argv) < 2 {
        fmt.Printf("Sleep 100ms, 100us, 100ns, ...\n")
        return
    }
    duration := argv[1];
    d, err := time.ParseDuration(duration)
    if err != nil {
        d, err = time.ParseDuration(duration+"s")
        if err != nil {
            fmt.Printf("duration ? %s (%s)\n",duration,err)
            return
        }
    }
    //fmt.Printf("Sleep %v\n",duration)
    time.Sleep(d)
    if 0 < len(argv[2:]) {
        gshCtx.gshellv(argv[2:])
    }
}
func (gshCtx *GshContext)repeat(argv []string) {
    if len(argv) < 2 {
        return
    }
    start0 := time.Now()
    for ri,_ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
        if 0 < len(argv[2:]) {
            //start := time.Now()
            gshCtx.gshellv(argv[2:])
            end := time.Now()
            elps := end.Sub(start0);
            if( 1000000000 < elps ){
                fmt.Printf("(repeat#%d %v)\n",ri,elps);
            }
        }
    }
}

func (gshCtx *GshContext)gen(argv []string) {
    gshPA := gshCtx.gshPA
    if len(argv) < 2 {
        fmt.Printf("Usage: %s N\n",argv[0])
        return
    }
    // should br repeated by "repeat" command
    count, _ := strconv.Atoi(argv[1])
    fd := gshPA.Files[1] // Stdout
    file := os.NewFile(fd,"internalStdOut")
    fmt.Printf("--I-- Gen. Count=%d to [%d]\n",count,file.Fd())
    //buf := []byte{}
    outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
    for gi := 0; gi < count; gi++ {
        file.WriteString(outdata)
    }
    //file.WriteString("\n")
    fmt.Printf("\n(%d B)\n",count*len(outdata));
    //file.Close()
}

// <a name="rexec">Remote Execution</a> // 2020-0820
func Elapsed(from time.Time)(string){
    elps := time.Now().Sub(from)
    if 1000000000 < elps {
        return fmt.Sprintf("[%5d.%02ds]",elps/1000000000,(elps%1000000000)/10000000)
    }else
    if 1000000 < elps {
        return fmt.Sprintf("[%3d.%03dms]",elps/1000000,(elps%1000000)/1000)
    }else{
        return fmt.Sprintf("[%3d.%03dus]",elps/1000,(elps%1000))
    }
}
func abbtime(nanos int64)(string){
    if 1000000000 < nanos {
        return fmt.Sprintf("%d.%02ds",nanos/1000000000,(nanos%1000000000)/10000000)
    }else
    if 1000000 < nanos {
        return fmt.Sprintf("%d.%03dms",nanos/1000000,(nanos%1000000)/1000)
    }else{
        return fmt.Sprintf("%d.%03dus",nanos/1000,(nanos%1000))
    }
}
func abssize(size int64)(string){
```

```go
    fsize := float64(size)
    if 1024*1024*1024 < size {
        return fmt.Sprintf("%.2fGiB",fsize/(1024*1024*1024))
    }else
    if 1024*1024 < size {
        return fmt.Sprintf("%.3fMiB",fsize/(1024*1024))
    }else{
        return fmt.Sprintf("%.3fKiB",fsize/1024)
    }
}
func absize(size int64)(string){
    fsize := float64(size)
    if 1024*1024*1024 < size {
        return fmt.Sprintf("%8.2fGiB",fsize/(1024*1024*1024))
    }else
    if 1024*1024 < size {
        return fmt.Sprintf("%8.3fMiB",fsize/(1024*1024))
    }else{
        return fmt.Sprintf("%8.3fKiB",fsize/1024)
    }
}
func abbspeed(totalB int64,ns int64)(string){
    MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
    if 1000 <= MBs {
        return fmt.Sprintf("%6.3fGB/s",MBs/1000)
    }
    if 1 <= MBs {
        return fmt.Sprintf("%6.3fMB/s",MBs)
    }else{
        return fmt.Sprintf("%6.3fKB/s",MBs*1000)
    }
}
func abspeed(totalB int64,ns time.Duration)(string){
    MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
    if 1000 <= MBs {
        return fmt.Sprintf("%6.3fGBps",MBs/1000)
    }
    if 1 <= MBs {
        return fmt.Sprintf("%6.3fMBps",MBs)
    }else{
        return fmt.Sprintf("%6.3fKBps",MBs*1000)
    }
}
func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
    Start := time.Now()
    buff := make([]byte,bsiz)
    var total int64 = 0
    var rem int64 = size
    nio := 0
    Prev := time.Now()
    var PrevSize int64 = 0

    fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) START\n",
        what,absize(total),size,nio)

    for i:= 0; ; i++ {
        var len = bsiz
        if int(rem) < len {
            len = int(rem)
        }
        Now := time.Now()
        Elps := Now.Sub(Prev);
        if 1000000000 < Now.Sub(Prev) {
            fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %s\n",
                what,absize(total),size,nio,
                abspeed((total-PrevSize),Elps))
            Prev = Now;
            PrevSize = total
        }
        rlen := len
        if in != nil {
            // should watch the disconnection of out
            rcc,err := in.Read(buff[0:rlen])
            if err != nil {
                fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
                    what,rcc,err,in.Name())
                break
            }
            rlen = rcc
            if string(buff[0:10]) == "((SoftEOF " {
                var ecc int64 = 0
                fmt.Sscanf(string(buff),"((SoftEOF %v",&ecc)
                fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
                    what,ecc,total)
                if ecc == total {
                    break
                }
            }
        }

        wlen := rlen
        if out != nil {
            wcc,err := out.Write(buff[0:rlen])
            if err != nil {
                fmt.Printf(Elapsed(Start)+"-En-- X: %s write(%v,%v)>%v\n",
                    what,wcc,err,out.Name())
                break
            }
            wlen = wcc
        }
        if wlen < rlen {
            fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
                what,wlen,rlen)
            break;
        }

        nio += 1
        total += int64(rlen)
        rem -= int64(rlen)
        if rem <= 0 {
            break
        }
    }
    Done := time.Now()
    Elps := float64(Done.Sub(Start))/1000000000 //Seconds
    TotalMB := float64(total)/1000000 //MB
    MBps := TotalMB / Elps
    fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v/%v) %v %.3fMB/s\n",
        what,total,size,nio,absize(total),MBps)
    return total
}
func tcpPush(clnt *os.File){
    // shrink socket buffer and recover
    usleep(100);
```

```
}
func (gsh*GshContext)RexecServer(argv[]string){
	debug := true
	Start0 := time.Now()
	Start := Start0
//	if local == ":" { local = "0.0.0.0:9999" }
	local := "0.0.0.0:9999"

	if 0 < len(argv) {
		if argv[0] == "-s" {
			debug = false
			argv = argv[1:]
		}
	}
	if 0 < len(argv) {
		argv = argv[1:]
	}
	port, err := net.ResolveTCPAddr("tcp",local);
	if err != nil {
		fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
		return
	}
	fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
	sconn, err := net.ListenTCP("tcp", port)
	if err != nil {
		fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
		return
	}

	reqbuf := make([]byte,LINESIZE)
	res := ""
	for {
		fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
		aconn, err := sconn.AcceptTCP()
		Start = time.Now()
		if err != nil {
			fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
			return
		}
		clnt, _ := aconn.File()
		fd := clnt.Fd()
		ar := aconn.RemoteAddr()
		if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
			local,fd,ar) }
		res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
		fmt.Fprintf(clnt,"%s",res)
		if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
		count, err := clnt.Read(reqbuf)
		if err != nil {
			fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
				count,err,string(reqbuf))
		}
		req := string(reqbuf[:count])
		if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
		reqv := strings.Split(string(req),"\r")
		cmdv := gshScanArg(reqv[0],0)
		//cmdv := strings.Split(reqv[0]," ")
		switch cmdv[0] {
			case "HELO":
				res = fmt.Sprintf("250 %v",req)
			case "GET":
				// download {remotefile|-zN} [localfile]
				var dsize int64 = 32*1024*1024
				var bsize int = 64*1024
				var fname string = ""
				var in *os.File = nil
				var pseudoEOF = false
				if 1 < len(cmdv) {
					fname = cmdv[1]
					if strBegins(fname,"-z") {
						fmt.Sscanf(fname[2:],"%d",&dsize)
					}else
					if strBegins(fname,"{") {
						xin,xout,err := gsh.Popen(fname,"r")
						if err {
						}else{
							xout.Close()
							defer xin.Close()
							in = xin
							dsize = MaxStreamSize
							pseudoEOF = true
						}
					}else{
						xin,err := os.Open(fname)
						if err != nil {
							fmt.Printf("--En- GET (%v)\n",err)
						}else{
							defer xin.Close()
							in = xin
							fi,_ := xin.Stat()
							dsize = fi.Size()
						}
					}
				}
				//fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dsize,bsize)
				res = fmt.Sprintf("200 %v\r\n",dsize)
				fmt.Fprintf(clnt,"%v",res)
				tcpPush(clnt); // should be separated as line in receiver
				fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
				wcount := fileRelay("SendGET",in,clnt,dsize,bsize)
				if pseudoEOF {
					in.Close() // pipe from the command
					// show end of stream data (its size) by OOB?
					SoftEOF := fmt.Sprintf("((SoftEOF %v))",wcount)
					fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)

					tcpPush(clnt); // to let SoftEOF data apper at the top of recevied data
					fmt.Fprintf(clnt,"%v\r\n",SoftEOF)
					tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
						// with client generated random?
					//fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
				}
				res = fmt.Sprintf("200 GET done\r\n")
			case "PUT":
				// upload {srcfile|-zN} [dstfile]
				var dsize int64 = 32*1024*1024
				var bsize int = 64*1024
				var fname string = ""
				var out *os.File = nil
				if 1 < len(cmdv) { // localfile
					fmt.Sscanf(cmdv[1],"%d",&dsize)
				}
				if 2 < len(cmdv) {
					fname = cmdv[2]
```

```
                    if fname == "-" {
                        // nul dev
                    }else
                    if strBegins(fname,"{") {
                        xin,xout,err := gsh.Popen(fname,"w")
                        if err {
                        }else{
                            xin.Close()
                            defer xout.Close()
                            out = xout
                        }
                    }else{
                    // should write to temporary file
                    // should suppress ^C on tty
            xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
            //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n",fname,xout,err)
                        if err != nil {
                            fmt.Printf("--En- PUT (%v)\n",err)
                        }else{
                            out = xout
                        }
                    }
                fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
                    fname,local,err)
                }
                fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
                fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n",dsize)
                fmt.Fprintf(clnt,"200 %v OK\r\n",dsize)
                fileRelay("RecvPUT",clnt,out,dsize,bsize)
                res = fmt.Sprintf("200 PUT done\r\n")
            default:
                res = fmt.Sprintf("400 What? %v",req)
        }
        swcc,serr := clnt.Write([]byte(res))
        if serr != nil {
            fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v",swcc,serr,res)
        }else{
            fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
        }
        aconn.Close();
        clnt.Close();
    }
    sconn.Close();
}
func (gsh*GshContext)RexecClient(argv[]string)(int,string){
    debug := true
    Start := time.Now()
    if len(argv) == 1 {
        return -1,"EmptyARG"
    }
    argv = argv[1:]
    if argv[0] == "-serv" {
        gsh.RexecServer(argv[1:])
        return 0,"Server"
    }
    remote := "0.0.0.0:9999"
    if argv[0][0] == '@' {
        remote = argv[0][1:]
        argv = argv[1:]
    }
    if argv[0] == "-s" {
        debug = false
        argv = argv[1:]
    }
    dport, err := net.ResolveTCPAddr("tcp",remote);
    if err != nil {
        fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n",remote,err)
        return -1,"AddressError"
    }
    fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n",remote)
    serv, err := net.DialTCP("tcp",nil,dport)
    if err != nil {
        fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n",remote,err)
        return -1,"CannotConnect"
    }
    if debug {
        al := serv.LocalAddr()
        fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n",remote,al)
    }

    req := ""
    res := make([]byte,LINESIZE)
    count,err := serv.Read(res)
    if err != nil {
        fmt.Printf("--En- S: (%3d,%v) %v",count,err,string(res))
    }
    if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res)) }

    if argv[0] == "GET" {
        savPA := gsh.gshPA
        var bsize int = 64*1024
        req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
        fmt.Printf(Elapsed(Start)+"--In- C: %v",req)
        fmt.Fprintf(serv,req)
        count,err = serv.Read(res)
        if err != nil {
        }else{
            var dsize int64 = 0
            var out *os.File = nil
            var out_tobeclosed *os.File = nil
            var fname string = ""
            var rcode int = 0
            var pid int = -1
            fmt.Sscanf(string(res),"%d %d",&rcode,&dsize)
            fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count]))
            if 3 <= len(argv) {
                fname = argv[2]
                if strBegins(fname,"{") {
                    xin,xout,err := gsh.Popen(fname,"w")
                    if err {
                    }else{
                        xin.Close()
                        defer xout.Close()
                        out = xout
                        out_tobeclosed = xout
                        pid = 0 // should be its pid
                    }
                }else{
                    // should write to temporary file
                    // should suppress ^C on tty
                    xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
                    if err != nil {
                        fmt.Print("--En- %v\n",err)
                    }
```

```
					out = xout
					//fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
				}
			}
			in,_ := serv.File()
			fileRelay("RecvGET",in,out,dsize,bsize)
			if 0 <= pid {
				gsh.gshPA = savPA // recovery of Fd(), and more?
				fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
				out_tobeclosed.Close()
				//syscall.Wait4(pid,nil,0,nil) //@@
			}
		}
	}else
	if argv[0] == "PUT" {
		remote, _ := serv.File()
		var local *os.File = nil
		var dsize int64 = 32*1024*1024
		var bsize int = 64*1024
		var ofile string = "-"
		//fmt.Printf("--I-- Rex %v\n",argv)
		if 1 < len(argv) {
			fname := argv[1]
			if strBegins(fname,"-z") {
				fmt.Sscanf(fname[2:],"%d",&dsize)
			}else
			if strBegins(fname,"{") {
				xin,xout,err := gsh.Popen(fname,"r")
				if err {
				}else{
					xout.Close()
					defer xin.Close()
					//in = xin
					local = xin
					fmt.Printf("--In- [%d] < Upload output of %v\n",
						local.Fd(),fname)
					ofile = "-from."+fname
					dsize = MaxStreamSize
				}
			}else{
				xlocal,err := os.Open(fname)
				if err != nil {
					fmt.Printf("--En- (%s)\n",err)
					local = nil
				}else{
					local = xlocal
					fi,_ := local.Stat()
					dsize = fi.Size()
					defer local.Close()
					//fmt.Printf("--I-- Rex in(%v / %v)\n",ofile,dsize)
				}
				ofile = fname
				fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
					fname,dsize,local,err)
			}
		}
		if 2 < len(argv) && argv[2] != "" {
			ofile = argv[2]
			//fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
		}
		//fmt.Printf(Elapsed(Start)+"--I-- Rex out(%v)\n",ofile)
		fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
		req = fmt.Sprintf("PUT %v %v \r\n",dsize,ofile)
		if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
		fmt.Fprintf(serv,"%v",req)
		count,err = serv.Read(res)
		if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
		fileRelay("SendPUT",local,remote,dsize,bsize)
	}else{
		req = fmt.Sprintf("%v\r\n",strings.Join(argv," "))
		if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
		fmt.Fprintf(serv,"%v",req)
		//fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
	}
	//fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
	count,err = serv.Read(res)
	ress := ""
	if count == 0 {
		ress = "(nil)\r\n"
	}else{
		ress = string(res[:count])
	}
	if err != nil {
		fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,ress)
	}else{
		fmt.Printf(Elapsed(Start)+"--In- S: %v",ress)
	}
	serv.Close()
	//conn.Close()

	var stat string
	var rcode int
	fmt.Sscanf(ress,"%d %s",&rcode,&stat)
	//fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
	return rcode,ress
}

// <a name="remote-sh">Remote Shell</a>
// gcp file [...] { [host]:[port:][dir] | dir } // -p | -no-p
func (gsh*GshContext)FileCopy(argv[]string){
	var host = ""
	var port = ""
	var upload = false
	var download = false
	var xargv = []string{"rex-gcp"}
	var srcv = []string{}
	var dstv = []string{}
	argv = argv[1:]

	for _,v := range argv {
		/*
		if v[0] == '-' { // might be a pseudo file (generated date)
			continue
		}
		*/
		obj := strings.Split(v,":")
		//fmt.Printf("%d %v %v\n",len(obj),v,obj)
		if 1 < len(obj) {
			host = obj[0]
			file := ""
			if 0 < len(host) {
				gsh.LastServer.host = host
			}else{
				host = gsh.LastServer.host
```

```
                port = gsh.LastServer.port
            }
            if 2 < len(obj) {
                port = obj[1]
                if 0 < len(port) {
                    gsh.LastServer.port = port
                }else{
                    port = gsh.LastServer.port
                }
                file = obj[2]
            }else{
                file = obj[1]
            }
            if len(srcv) == 0 {
                download = true
                srcv = append(srcv,file)
                continue
            }
            upload = true
            dstv = append(dstv,file)
            continue
        }
        /*
        idx := strings.Index(v,":")
        if 0 <= idx {
            remote = v[0:idx]
            if len(srcv) == 0 {
                download = true
                srcv = append(srcv,v[idx+1:])
                continue
            }
            upload = true
            dstv = append(dstv,v[idx+1:])
            continue
        }
        */
        if download {
            dstv = append(dstv,v)
        }else{
            srcv = append(srcv,v)
        }
    }
    hostport := "@" + host + ":" + port
    if upload {
        if host != "" { xargv = append(xargv,hostport) }
        xargv = append(xargv,"PUT")
        xargv = append(xargv,srcv[0:]...)
        xargv = append(xargv,dstv[0:]...)
    //fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)
    fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v\n",hostport,dstv,srcv)
        gsh.RexecClient(xargv)
    }else
    if download {
        if host != "" { xargv = append(xargv,hostport) }
        xargv = append(xargv,"GET")
        xargv = append(xargv,srcv[0:]...)
        xargv = append(xargv,dstv[0:]...)
    //fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v // %v\n",hostport,srcv,dstv,xargv)
    fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v\n",hostport,srcv,dstv)
        gsh.RexecClient(xargv)
    }else{
    }
}

// target
func (gsh*GshContext)Trelpath(rloc string)(string){
    cwd, _ := os.Getwd()
    os.Chdir(gsh.RWD)
    os.Chdir(rloc)
    twd, _ := os.Getwd()
    os.Chdir(cwd)

    tpath := twd + "/" + rloc
    return tpath
}
// join to rmote GShell - [user@]host[:port] or cd host:[port]:path
func (gsh*GshContext)Rjoin(argv[]string){
    if len(argv) <= 1 {
        fmt.Printf("--I-- current server = %v\n",gsh.RSERV)
        return
    }
    serv := argv[1]
    servv := strings.Split(serv,":")
    if 1 <= len(servv) {
        if servv[0] == "lo" {
            servv[0] = "localhost"
        }
    }
    switch len(servv) {
        case 1:
            //if strings.Index(serv,":") < 0 {
            serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
            //}
        case 2: // host:port
            serv = strings.Join(servv,":")
    }
    xargv := []string{"rex-join","@"+serv,"HELO"}
    rcode,stat := gsh.RexecClient(xargv)
    if (rcode / 100) == 2 {
        fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
        gsh.RSERV = serv
    }else{
        fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
    }
}
func (gsh*GshContext)Rexec(argv[]string){
    if len(argv) <= 1 {
        fmt.Printf("--I-- rexec command [ | {file || {command} ]\n",gsh.RSERV)
        return
    }

    /*
    nargv := gshScanArg(strings.Join(argv," "),0)
    fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
    if nargv[1][0] != '{' {
        nargv[1] = "{" + nargv[1] + "}"
        fmt.Printf("--D-- nargc=%d [%v]\n",len(nargv),nargv)
    }
    argv = nargv
    */
    nargv := []string{}
    nargv = append(nargv,"{"+strings.Join(argv[1:]," ")+"}")
    fmt.Printf("--D-- nargc=%d %v\n",len(nargv),nargv)
    argv = nargv
```

```

	xargv := []string{"rex-exec","@"+gsh.RSERV,"GET"}
	xargv = append(xargv,argv...)
	xargv = append(xargv,"/dev/tty")
	rcode,stat := gsh.RexecClient(xargv)
	if (rcode / 100) == 2 {
		fmt.Printf("--I-- OK Rexec (%v) %v\n",rcode,stat)
	}else{
		fmt.Printf("--I-- NG Rexec (%v) %v\n",rcode,stat)
	}
}
func (gsh*GshContext)Rchdir(argv[]string){
	if len(argv) <= 1 {
		return
	}
	cwd, _ := os.Getwd()
	os.Chdir(gsh.RWD)
	os.Chdir(argv[1])
	twd, _ := os.Getwd()
	gsh.RWD = twd
	fmt.Printf("--I-- JWD=%v\n",twd)
	os.Chdir(cwd)
}
func (gsh*GshContext)Rpwd(argv[]string){
	fmt.Printf("%v\n",gsh.RWD)
}
func (gsh*GshContext)Rls(argv[]string){
	cwd, _ := os.Getwd()
	os.Chdir(gsh.RWD)
	argv[0] = "-ls"
	gsh.xFind(argv)
	os.Chdir(cwd)
}
func (gsh*GshContext)Rput(argv[]string){
	var local string = ""
	var remote string = ""
	if 1 < len(argv) {
		local = argv[1]
		remote = local // base name
	}
	if 2 < len(argv) {
		remote = argv[2]
	}
	fmt.Printf("--I-- jput from=%v to=%v\n",local,gsh.Trelpath(remote))
}
func (gsh*GshContext)Rget(argv[]string){
	var remote string = ""
	var local string = ""
	if 1 < len(argv) {
		remote = argv[1]
		local = remote // base name
	}
	if 2 < len(argv) {
		local = argv[2]
	}
	fmt.Printf("--I-- jget from=%v to=%v\n",gsh.Trelpath(remote),local)
}

// <a name="network">network</a>
// -s, -si, -so // bi-directional, source, sync (maybe socket)
func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
	gshPA := gshCtx.gshPA
	if len(argv) < 2 {
		fmt.Printf("Usage: -s [host]:[port[.udp]]\n")
		return
	}
	remote := argv[1]
	if remote == ":" { remote = "0.0.0.0:9999" }

	if inTCP { // TCP
		dport, err := net.ResolveTCPAddr("tcp",remote);
		if err != nil {
			fmt.Printf("Address error: %s (%s)\n",remote,err)
			return
		}
		conn, err := net.DialTCP("tcp",nil,dport)
		if err != nil {
			fmt.Printf("Connection error: %s (%s)\n",remote,err)
			return
		}
		file, _ := conn.File();
		fd := file.Fd()
		fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)

		savfd := gshPA.Files[1]
		gshPA.Files[1] = fd;
		gshCtx.gshellv(argv[2:])
		gshPA.Files[1] = savfd
		file.Close()
		conn.Close()
	}else{
		//dport, err := net.ResolveUDPAddr("udp4",remote);
		dport, err := net.ResolveUDPAddr("udp",remote);
		if err != nil {
			fmt.Printf("Address error: %s (%s)\n",remote,err)
			return
		}
		//conn, err := net.DialUDP("udp4",nil,dport)
		conn, err := net.DialUDP("udp",nil,dport)
		if err != nil {
			fmt.Printf("Connection error: %s (%s)\n",remote,err)
			return
		}
		file, _ := conn.File();
		fd := file.Fd()

		ar := conn.RemoteAddr()
		//al := conn.LocalAddr()
		fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
			remote,ar.String(),fd)

		savfd := gshPA.Files[1]
		gshPA.Files[1] = fd;
		gshCtx.gshellv(argv[2:])
		gshPA.Files[1] = savfd
		file.Close()
		conn.Close()
	}
}
func (gshCtx*GshContext)saccept(inTCP bool, argv []string) {
	gshPA := gshCtx.gshPA
	if len(argv) < 2 {
		fmt.Printf("Usage: -ac [host]:[port[.udp]]\n")
		return
```

```
    }
    local := argv[1]
    if local == ":" { local = "0.0.0.0:9999" }
    if inTCP { // TCP
        port, err := net.ResolveTCPAddr("tcp",local);
        if err != nil {
            fmt.Printf("Address error: %s (%s)\n",local,err)
            return
        }
        //fmt.Printf("Listen at %s...\n",local);
        sconn, err := net.ListenTCP("tcp", port)
        if err != nil {
            fmt.Printf("Listen error: %s (%s)\n",local,err)
            return
        }
        //fmt.Printf("Accepting at %s...\n",local);
        aconn, err := sconn.AcceptTCP()
        if err != nil {
            fmt.Printf("Accept error: %s (%s)\n",local,err)
            return
        }
        file, _ := aconn.File()
        fd := file.Fd()
        fmt.Printf("Accepted TCP at %s [%d]\n",local,fd)

        savfd := gshPA.Files[0]
        gshPA.Files[0] = fd;
        gshCtx.gshellv(argv[2:])
        gshPA.Files[0] = savfd

        sconn.Close();
        aconn.Close();
        file.Close();
    }else{
        //port, err := net.ResolveUDPAddr("udp4",local);
        port, err := net.ResolveUDPAddr("udp",local);
        if err != nil {
            fmt.Printf("Address error: %s (%s)\n",local,err)
            return
        }
        fmt.Printf("Listen UDP at %s...\n",local);
        //uconn, err := net.ListenUDP("udp4", port)
        uconn, err := net.ListenUDP("udp", port)
        if err != nil {
            fmt.Printf("Listen error: %s (%s)\n",local,err)
            return
        }
        file, _ := uconn.File()
        fd := file.Fd()
        ar := uconn.RemoteAddr()
        remote := ""
        if ar != nil { remote = ar.String() }
        if remote == "" { remote = "?" }

        // not yet received
        //fmt.Printf("Accepted at %s [%d] <- %s\n",local,fd,"")

        savfd := gshPA.Files[0]
        gshPA.Files[0] = fd;
        savenv := gshPA.Env
        gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
        gshCtx.gshellv(argv[2:])
        gshPA.Env = savenv
        gshPA.Files[0] = savfd

        uconn.Close();
        file.Close();
    }
}

// empty line command
func (gshCtx*GshContext)xPwd(argv[]string){
    // execute context command, pwd + date
    // context notation, representation scheme, to be resumed at re-login
    cwd, _ := os.Getwd()
    switch {
    case isin("-a",argv):
        gshCtx.ShowChdirHistory(argv)
    case isin("-ls",argv):
        showFileInfo(cwd,argv)
    default:
        fmt.Printf("%s\n",cwd)
    case isin("-v",argv): // obsolete emtpy command
        t := time.Now()
        date := t.Format(time.UnixDate)
        exe, _ := os.Executable()
        host, _ := os.Hostname()
        fmt.Printf("{PWD=\"%s\"",cwd)
        fmt.Printf(" HOST=\"%s\"",host)
        fmt.Printf(" DATE=\"%s\"",date)
        fmt.Printf(" TIME=\"%s\"",t.String())
        fmt.Printf(" PID=\"%d\"",os.Getpid())
        fmt.Printf(" EXE=\"%s\"",exe)
        fmt.Printf("}\n")
    }
}

// <a name="history">History</a>
// these should be browsed and edited by HTTP browser
// show the time of command with -t and direcotry with -ls
// openfile-history, sort by -a -m -c
// sort by elapsed time by -t -s
// search by "more" like interface
// edit history
// sort history, and wc or uniq
// CPU and other resource consumptions
// limit showing range (by time or so)
// export / import history
func (gshCtx *GshContext)xHistory(argv []string){
    atWorkDirX := -1
    if 1 < len(argv) && strBegins(argv[1],"@") {
        atWorkDirX,_ = strconv.Atoi(argv[1][1:])
    }
    //fmt.Printf("--D-- showHistory(%v)\n",argv)
    for i, v := range gshCtx.CommandHistory {
        // exclude commands not to be listed by default
        // internal commands may be suppressed by default
        if v.CmdLine == "" && !isin("-a",argv) {
            continue;
        }
        if 0 <= atWorkDirX {
            if v.WorkDirX != atWorkDirX {
                continue
            }
```

```
		}
		if !isin("-n",argv){ // like "fc"
			fmt.Printf("!%-2d ",i)
		}
		if isin("-v",argv){
			fmt.Println(v) // should be with it date
		}else{
			if isin("-l",argv) || isin("-l0",argv) {
				elps := v.EndAt.Sub(v.StartAt);
				start := v.StartAt.Format(time.Stamp)
				fmt.Printf("@%d ",v.WorkDirX)
				fmt.Printf("[%v] %11v/t ",start,elps)
			}
			if isin("-l",argv) && !isin("-l0",argv){
				fmt.Printf("%v",Rusagef("%t %u\t// %s",argv,v.Rusagev))
			}
			if isin("-at",argv) { // isin("-ls",argv){
				dhi := v.WorkDirX // workdir history index
				fmt.Printf("@%d %s\t",dhi,v.WorkDir)
				// show the FileInfo of the output command??
			}
			fmt.Printf("%s",v.CmdLine)
			fmt.Printf("\n")
		}
	}
}
// !n - history index
func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
	if gline[0] == '!' {
		hix, err := strconv.Atoi(gline[1:])
		if err != nil {
			fmt.Printf("--E-- (%s : range)\n",hix)
			return "", false, true
		}
		if hix < 0 || len(gshCtx.CommandHistory) <= hix {
			fmt.Printf("--E-- (%d : out of range)\n",hix)
			return "", false, true
		}
		return gshCtx.CommandHistory[hix].CmdLine, false, false
	}
	// search
	//for i, v := range gshCtx.CommandHistory {
	//}
	return gline, false, false
}
func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
	if 0 <= hix && hix < len(gsh.CommandHistory) {
		return gsh.CommandHistory[hix].CmdLine,true
	}
	return "",false
}

// temporary adding to PATH environment
// cd name -lib for LD_LIBRARY_PATH
// chdir with directory history (date + full-path)
// -s for sort option (by visit date or so)
func (gsh*GshContext)ShowChdirHistory1(i int,v GChdirHistory, argv []string){
	fmt.Printf("!%-2d ",v.CmdIndex) // the first command at this WorkDir
	fmt.Printf("@%d ",i)
	fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
	showFileInfo(v.Dir,argv)
}
func (gsh*GshContext)ShowChdirHistory(argv []string){
	for i, v := range gsh.ChdirHistory {
		gsh.ShowChdirHistory1(i,v,argv)
	}
}
func skipOpts(argv[]string)(int){
	for i,v := range argv {
		if strBegins(v,"-") {
		}else{
			return i
		}
	}
	return -1
}
func (gshCtx*GshContext)xChdir(argv []string){
	cdhist := gshCtx.ChdirHistory
	if isin("?",argv ) || isin("-t",argv) || isin("-a",argv) {
		gshCtx.ShowChdirHistory(argv)
		return
	}
	pwd, _ := os.Getwd()
	dir := ""
	if len(argv) <= 1 {
		dir = toFullpath("~")
	}else{
		i := skipOpts(argv[1:])
		if i < 0 {
			dir = toFullpath("~")
		}else{
			dir = argv[1+i]
		}
	}
	if strBegins(dir,"@") {
		if dir == "@0" { // obsolete
			dir = gshCtx.StartDir
		}else
		if dir == "@!" {
			index := len(cdhist) - 1
			if 0 < index { index -= 1 }
			dir = cdhist[index].Dir
		}else{
			index, err := strconv.Atoi(dir[1:])
			if err != nil {
				fmt.Printf("--E-- xChdir(%v)\n",err)
				dir = "?"
			}else
			if len(gshCtx.ChdirHistory) <= index {
				fmt.Printf("--E-- xChdir(history range error)\n")
				dir = "?"
			}else{
				dir = cdhist[index].Dir
			}
		}
	}
	if dir != "?" {
		err := os.Chdir(dir)
		if err != nil {
			fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
		}else{
			cwd, _ := os.Getwd()
			if cwd != pwd {
				hist1 := GChdirHistory { }
				hist1.Dir = cwd
				hist1.MovedAt = time.Now()
				hist1.CmdIndex = len(gshCtx.CommandHistory)+1
				gshCtx.ChdirHistory = append(cdhist,hist1)
				if !isin("-s",argv){
					//cwd, _ := os.Getwd()
					//fmt.Printf("%s\n",cwd)
					ix := len(gshCtx.ChdirHistory)-1
					gshCtx.ShowChdirHistory1(ix,hist1,argv)
				}
			}
		}
	}
	if isin("-ls",argv){
		cwd, _ := os.Getwd()
		showFileInfo(cwd,argv);
	}
}
func TimeValSub(tv1 *syscall.Timeval, tv2 *syscall.Timeval){
	*tv1 = syscall.NsecToTimeval(tv1.Nano() - tv2.Nano())
}
func RusageSubv(ru1, ru2 [2]syscall.Rusage)([2]syscall.Rusage){
	TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
	TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
	TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
	TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
	return ru1
}
func TimeValAdd(tv1 syscall.Timeval, tv2 syscall.Timeval)(syscall.Timeval){
	tvs := syscall.NsecToTimeval(tv1.Nano() + tv2.Nano())
	return tvs
}
/*
func RusageAddv(ru1, ru2 [2]syscall.Rusage)([2]syscall.Rusage){
	TimeValAdd(ru1[0].Utime,ru2[0].Utime)
	TimeValAdd(ru1[0].Stime,ru2[0].Stime)
	TimeValAdd(ru1[1].Utime,ru2[1].Utime)
	TimeValAdd(ru1[1].Stime,ru2[1].Stime)
	return ru1
}
*/

// <a name="rusage">Resource Usage</a>
func sRusagef(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
	// ru[0] self , ru[1] children
	ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
	st := TimeValAdd(ru[0].Stime,ru[1].Stime)
	uu := (ut.Sec*1000000 + int64(ut.Usec)) * 1000
	su := (st.Sec*1000000 + int64(st.Usec)) * 1000
	tu := uu + su
	ret := fmt.Sprintf("%v/sum",abbtime(tu))
	ret += fmt.Sprintf(", %v/usr",abbtime(uu))
	ret += fmt.Sprintf(", %v/sys",abbtime(su))
	return ret
}
func Rusagef(fmtspec string, argv []string, ru [2]syscall.Rusage)(string){
	ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
	st := TimeValAdd(ru[0].Stime,ru[1].Stime)
	fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
	fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
	return ""
}
func Getrusagev()([2]syscall.Rusage){
	var ruv = [2]syscall.Rusage{}
	syscall.Getrusage(syscall.RUSAGE_SELF,&ruv[0])
	syscall.Getrusage(syscall.RUSAGE_CHILDREN,&ruv[1])
	return ruv
}
func showRusage(what string,argv []string, ru *syscall.Rusage){
	fmt.Printf("%s: ",what);
	fmt.Printf("Usr=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
	fmt.Printf(" Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
	fmt.Printf(" Rss=%vB",ru.Maxrss)
	if isin("-l",argv) {
		fmt.Printf(" MinFlt=%v",ru.Minflt)
		fmt.Printf(" MajFlt=%v",ru.Majflt)
		fmt.Printf(" IxRSS=%vB",ru.Ixrss)
		fmt.Printf(" IdRSS=%vB",ru.Idrss)
		fmt.Printf(" Nswap=%vB",ru.Nswap)
	fmt.Printf(" Read=%v",ru.Inblock)
	fmt.Printf(" Write=%v",ru.Oublock)
	}
	fmt.Printf(" Snd=%v",ru.Msgsnd)
	fmt.Printf(" Rcv=%v",ru.Msgrcv)
	//if isin("-l",argv) {
		fmt.Printf(" Sig=%v",ru.Nsignals)
	//}
	fmt.Printf("\n");
}
func (gshCtx *GshContext)xTime(argv[]string)(bool){
	if 2 <= len(argv){
		gshCtx.LastRusage = syscall.Rusage{}
		rusagev1 := Getrusagev()
		fin := gshCtx.gshellv(argv[1:])
		rusagev2 := Getrusagev()
		showRusage(argv[1],argv,&gshCtx.LastRusage)
		rusagev := RusageSubv(rusagev2,rusagev1)
		showRusage("self",argv,&rusagev[0])
		showRusage("chld",argv,&rusagev[1])
		return fin
	}else{
		rusage:= syscall.Rusage {}
		syscall.Getrusage(syscall.RUSAGE_SELF,&rusage)
		showRusage("self",argv, &rusage)
		syscall.Getrusage(syscall.RUSAGE_CHILDREN,&rusage)
		showRusage("chld",argv, &rusage)
		return false
	}
}
func (gshCtx *GshContext)xJobs(argv[]string){
	fmt.Printf("%d Jobs\n",len(gshCtx.BackGroundJobs))
	for ji, pid := range gshCtx.BackGroundJobs {
		//wstat := syscall.WaitStatus {0}
		rusage := syscall.Rusage {}
		//wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
		wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
		if err != nil {
			fmt.Printf("--E-- %%%d [%d] (%v)\n",ji,pid,err)
		}else{
			fmt.Printf("%%%d[%d](%d)\n",ji,pid,wpid)
			showRusage("chld",argv,&rusage)
		}
	}
}
```

```
func (gsh*GshContext)inBackground(argv[]string)(bool){
	if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n",argv) }
	gsh.BackGround = true // set background option
	xfin := false
	xfin = gsh.gshellv(argv)
	gsh.BackGround = false
	return xfin
}
// -o file without command means just opening it and refer by #N
// should be listed by "files" comnmand
func (gshCtx*GshContext)xOpen(argv[]string){
	var pv = []int{-1,-1}
	err := syscall.Pipe(pv)
	fmt.Printf("--I-- pipe()=[#%d,#%d](%v)\n",pv[0],pv[1],err)
}
func (gshCtx*GshContext)fromPipe(argv[]string){
}
func (gshCtx*GshContext)xClose(argv[]string){
}

// <a name="redirect">redirect</a>
func (gshCtx*GshContext)redirect(argv[]string)(bool){
	if len(argv) < 2 {
		return false
	}

	cmd := argv[0]
	fname := argv[1]
	var file *os.File = nil

	fdix := 0
	mode := os.O_RDONLY

	switch {
	case cmd == "-i" || cmd == "<":
		fdix = 0
		mode = os.O_RDONLY
	case cmd == "-o" || cmd == ">":
		fdix = 1
		mode = os.O_RDWR | os.O_CREATE
	case cmd == "-a" || cmd == ">>":
		fdix = 1
		mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
	}
	if fname[0] == '#' {
		fd, err := strconv.Atoi(fname[1:])
		if err != nil {
			fmt.Printf("--E-- (%v)\n",err)
			return false
		}
		file = os.NewFile(uintptr(fd),"MaybePipe")
	}else{
		xfile, err := os.OpenFile(argv[1], mode, 0600)
		if err != nil {
			fmt.Printf("--E-- (%s)\n",err)
			return false
		}
		file = xfile
	}
	gshPA := gshCtx.gshPA
	savfd := gshPA.Files[fdix]
	gshPA.Files[fdix] = file.Fd()
	fmt.Printf("--I-- Opened [%d] %s\n",file.Fd(),argv[1])
	gshCtx.gshellv(argv[2:])
	gshPA.Files[fdix] = savfd

	return false
}

//fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
func httpHandler(res http.ResponseWriter, req *http.Request){
	path := req.URL.Path
	fmt.Printf("--I-- Got HTTP Request(%s)\n",path)
	{
		gshCtxBuf, _ :=  setupGshContext()
		gshCtx := &gshCtxBuf
		fmt.Printf("--I-- %s\n",path[1:])
		gshCtx.tgshelll(path[1:])
	}
	fmt.Fprintf(res, "Hello(^-^)//\n%s\n",path)
}
func (gshCtx *GshContext) httpServer(argv []string){
	http.HandleFunc("/", httpHandler)
	accport := "localhost:9999"
	fmt.Printf("--I-- HTTP Server Start at [%s]\n",accport)
	http.ListenAndServe(accport,nil)
}
func (gshCtx *GshContext)xGo(argv[]string){
	go gshCtx.gshellv(argv[1:]);
}
func (gshCtx *GshContext) xPs(argv[]string)(){
}

// <a name="plugin">Plugin</a>
// plugin [-ls [names]] to list plugins
// Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
func (gshCtx *GshContext) whichPlugin(name string,argv[]string)(pi *PluginInfo){
	pi = nil
	for _,p := range gshCtx.PluginFuncs {
		if p.Name == name && pi == nil {
			pi = &p
		}
		if !isin("-s",argv){
			//fmt.Printf("%v %v ",i,p)
			if isin("-ls",argv){
				showFileInfo(p.Path,argv)
			}else{
				fmt.Printf("%s\n",p.Name)
			}
		}
	}
	return pi
}
func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
	if len(argv) == 0 || argv[0] == "-ls" {
		gshCtx.whichPlugin("",argv)
		return  nil
	}
	name := argv[0]
	Pin := gshCtx.whichPlugin(name,[]string{"-s"})
	if Pin != nil {
		os.Args = argv // should be recovered?
		Pin.Addr.(func())()
		return nil
```

```
    }
    sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)

    p, err := plugin.Open(sofile)
    if err != nil {
        fmt.Printf("--E-- plugin.Open(%s)(%v)\n",sofile,err)
        return err
    }
    fname := "Main"
    f, err := p.Lookup(fname)
    if( err != nil ){
        fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n",fname,err)
        return err
    }
    pin := PluginInfo {p,f,name,sofile}
    gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
    fmt.Printf("--I-- added (%d)\n",len(gshCtx.PluginFuncs))

    //fmt.Printf("--I-- first call(%s:%s)%v\n",sofile,fname,argv)
    os.Args = argv
    f.(func())()
    return err
}
func (gshCtx*GshContext)Args(argv[]string){
    for i,v := range os.Args {
        fmt.Printf("[%v] %v\n",i,v)
    }
}
func (gshCtx *GshContext) showVersion(argv[]string){
    if isin("-l",argv) {
        fmt.Printf("%v/%v (%v)",NAME,VERSION,DATE);
    }else{
        fmt.Printf("%v",VERSION);
    }
    if isin("-a",argv) {
        fmt.Printf(" %s",AUTHOR)
    }
    if !isin("-n",argv) {
        fmt.Printf("\n")
    }
}

// <a name="scanf">Scanf</a> // string decomposer
// scanf [format] [input]
func scanv(sstr string)(strv[]string){
    strv = strings.Split(sstr," ")
    return strv
}
func scanUntil(src,end string)(rstr string,leng int){
    idx := strings.Index(src,end)
    if 0 <= idx {
        rstr = src[0:idx]
        return rstr,idx+len(end)
    }
    return src,0
}

// -bn -- display base-name part only // can be in some %fmt, for sed rewriting
func (gsh*GshContext)printVal(fmts string, vstr string, optv[]string){
    //vint,err := strconv.Atoi(vstr)
    var ival int64 = 0
    n := 0
    err := error(nil)
    if strBegins(vstr,"_") {
        vx,_ := strconv.Atoi(vstr[1:])
        if vx < len(gsh.iValues) {
            vstr = gsh.iValues[vx]
        }else{
        }
    }
    // should use Eval()
    if strBegins(vstr,"0x") {
        n,err = fmt.Sscanf(vstr[2:],"%x",&ival)
    }else{
        n,err = fmt.Sscanf(vstr,"%d",&ival)
//fmt.Printf("--D-- n=%d err=(%v) {%s}=%v\n",n,err,vstr, ival)
    }
    if n == 1 && err == nil {
        //fmt.Printf("--D-- formatn(%v) ival(%v)\n",fmts,ival)
        fmt.Printf("%"+fmts,ival)
    }else{
        if isin("-bn",optv){
            fmt.Printf("%"+fmts,filepath.Base(vstr))
        }else{
            fmt.Printf("%"+fmts,vstr)
        }
    }
}
func (gsh*GshContext)printfv(fmts,div string,argv[]string,optv[]string,list[]string){
    //fmt.Printf("{%d}",len(list))
    //curfmt := "v"
    outlen := 0
    curfmt := gsh.iFormat

    if 0 < len(fmts) {
        for xi := 0; xi < len(fmts); xi++ {
            fch := fmts[xi]
            if fch == '%' {
                if xi+1 < len(fmts) {
                    curfmt = string(fmts[xi+1])
 gsh.iFormat = curfmt
                    xi += 1
    if xi+1 < len(fmts) && fmts[xi+1] == '(' {
        vals,leng := scanUntil(fmts[xi+2:],")")
        //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
        gsh.printVal(curfmt,vals,optv)
        xi += 2+leng-1
        outlen += 1
    }
                    continue
                }
            }
            if fch == '_' {
                hi,leng := scanInt(fmts[xi+1:])
                if 0 < leng {
                    if hi < len(gsh.iValues) {
                        gsh.printVal(curfmt,gsh.iValues[hi],optv)
                        outlen += 1 // should be the real length
                    }else{
                        fmt.Printf("((out-range))")
                    }
                    xi += leng
                    continue;
                }
            }
            fmt.Printf("%c",fch)
            outlen += 1
        }
    }else{
        //fmt.Printf("--D-- print {%s}\n")
        for i,v := range list {
            if 0 < i {
                fmt.Printf(div)
            }
            gsh.printVal(curfmt,v,optv)
            outlen += 1
        }
    }
    if 0 < outlen {
        fmt.Printf("\n")
    }
}
func (gsh*GshContext)Scanv(argv[]string){
    //fmt.Printf("--D-- Scanv(%v)\n",argv)
    if len(argv) == 1 {
        return
    }
    argv = argv[1:]
    fmts := ""
    if strBegins(argv[0],"-F") {
        fmts = argv[0]
        gsh.iDelimiter = fmts
        argv = argv[1:]
    }
    input := strings.Join(argv," ")
    if fmts == "" { // simple decomposition
        v := scanv(input)
        gsh.iValues = v
        //fmt.Printf("%v\n",strings.Join(v,","))
    }else{
        v := make([]string,8)
        n,err := fmt.Sscanf(input,fmts,&v[0],&v[1],&v[2],&v[3])
        fmt.Printf("--D-- Scanf ->(%v) n=%d err=(%v)\n",v,n,err)
        gsh.iValues = v
    }
}
func (gsh*GshContext)Printv(argv[]string){
    if false { //@@U
        fmt.Printf("%v\n",strings.Join(argv[1:]," "))
        return
    }
    //fmt.Printf("--D-- Printv(%v)\n",argv)
    //fmt.Printf("%v\n",strings.Join(gsh.iValues,","))
    div := gsh.iDelimiter
    fmts := ""
    argv = argv[1:]
    if 0 < len(argv) {
        if strBegins(argv[0],"-F") {
            div = argv[0][2:]
            argv = argv[1:]
        }
    }

    optv := []string{}
    for _,v := range argv {
        if strBegins(v,"-"){
            optv = append(optv,v)
            argv = argv[1:]
        }else{
            break;
        }
    }
    if 0 < len(argv) {
        fmts = strings.Join(argv," ")
    }
    gsh.printfv(fmts,div,argv,optv,gsh.iValues)
}
func (gsh*GshContext)Basename(argv[]string){
    for i,v := range gsh.iValues {
        gsh.iValues[i] = filepath.Base(v)
    }
}
func (gsh*GshContext)Sortv(argv[]string){
    sv := gsh.iValues
    sort.Slice(sv , func(i,j int) bool {
        return sv[i] < sv[j]
    })
}
func (gsh*GshContext)Shiftv(argv[]string){
    vi := len(gsh.iValues)
    if 0 < vi {
        if isin("-r",argv) {
            top := gsh.iValues[0]
            gsh.iValues = append(gsh.iValues[1:],top)
        }else{
            gsh.iValues = gsh.iValues[1:]
        }
    }
}

func (gsh*GshContext)Enq(argv[]string){
}
func (gsh*GshContext)Deq(argv[]string){
}
func (gsh*GshContext)Push(argv[]string){
    gsh.iValStack = append(gsh.iValStack,argv[1:])
    fmt.Printf("depth=%d\n",len(gsh.iValStack))
}
func (gsh*GshContext)Dump(argv[]string){
    for i,v := range gsh.iValStack {
        fmt.Printf("%d %v\n",i,v)
    }
}
func (gsh*GshContext)Pop(argv[]string){
    depth := len(gsh.iValStack)
    if 0 < depth {
        v := gsh.iValStack[depth-1]
        if isin("-cat",argv){
            gsh.iValues = append(gsh.iValues,v...)
        }else{
            gsh.iValues = v
        }
        gsh.iValStack = gsh.iValStack[0:depth-1]
        fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
    }else{
        fmt.Printf("depth=%d\n",depth)
    }
}
```

```

// <a name="interpreter">Command Interpreter</a>
func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
	fin = false

	if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv((%d))\n",len(argv)) }
	if len(argv) <= 0 {
		return false
	}
	xargv := []string{}
	for ai := 0; ai < len(argv); ai++ {
		xargv = append(xargv,strsubst(gshCtx,argv[ai],false))
	}
	argv = xargv
	if false {
		for ai := 0; ai < len(argv); ai++ {
			fmt.Printf("[%d] %s [%d]%T\n",
				ai,argv[ai],len(argv[ai]),argv[ai])
		}
	}
	cmd := argv[0]
	if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)%v\n",len(argv),argv) }
	switch { // https://tour.golang.org/flowcontrol/11
	case cmd == "":
		gshCtx.xPwd([]string{}); // emtpy command
	case cmd == "-x":
		gshCtx.CmdTrace = ! gshCtx.CmdTrace
	case cmd == "-xt":
		gshCtx.CmdTime = ! gshCtx.CmdTime
	case cmd == "-ot":
		gshCtx.sconnect(true, argv)
	case cmd == "-ou":
		gshCtx.sconnect(false, argv)
	case cmd == "-it":
		gshCtx.saccept(true , argv)
	case cmd == "-iu":
		gshCtx.saccept(false, argv)
	case cmd == "-i" || cmd == "<" || cmd == "-o" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
		gshCtx.redirect(argv)
	case cmd == "|":
		gshCtx.fromPipe(argv)
	case cmd == "args":
		gshCtx.Args(argv)
	case cmd == "bg" || cmd == "-bg":
		rfin := gshCtx.inBackground(argv[1:])
		return rfin
	case cmd == "-bn":
		gshCtx.Basename(argv)
	case cmd == "call":
		_,_ = gshCtx.excommand(false,argv[1:])
	case cmd == "cd" || cmd == "chdir":
		gshCtx.xChdir(argv);
	case cmd == "-cksum":
		gshCtx.xFind(argv)
	case cmd == "-sum":
		gshCtx.xFind(argv)
	case cmd == "-sumtest":
		str := ""
		if 1 < len(argv) { str = argv[1] }
		crc := strCRC32(str,uint64(len(str)))
		fprintf(stderr,"%v %v\n",crc,len(str))
	case cmd == "close":
		gshCtx.xClose(argv)
	case cmd == "gcp":
		gshCtx.FileCopy(argv)
	case cmd == "dec" || cmd == "decode":
		gshCtx.Dec(argv)
	case cmd == "#define":
	case cmd == "dic" || cmd == "d":
		xDic(argv)
	case cmd == "dump":
		gshCtx.Dump(argv)
	case cmd == "echo" || cmd == "e":
		echo(argv,true)
	case cmd == "enc" || cmd == "encode":
		gshCtx.Enc(argv)
	case cmd == "env":
		env(argv)
	case cmd == "eval":
		xEval(argv[1:],true)
	case cmd == "ev" || cmd == "events":
		dumpEvents(argv)
	case cmd == "exec":
		_,_ = gshCtx.excommand(true,argv[1:])
		// should not return here
	case cmd == "exit" || cmd == "quit":
		// write Result code EXIT to 3>
		return true
	case cmd == "fdls":
		// dump the attributes of fds (of other process)
	case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
		gshCtx.xFind(argv[1:])
	case cmd == "fu":
		gshCtx.xFind(argv[1:])
	case cmd == "fork":
		// mainly for a server
	case cmd == "-gen":
		gshCtx.gen(argv)
	case cmd == "-go":
		gshCtx.xGo(argv)
	case cmd == "-grep":
		gshCtx.xFind(argv)
	case cmd == "gdeq":
		gshCtx.Deq(argv)
	case cmd == "genq":
		gshCtx.Enq(argv)
	case cmd == "gpop":
		gshCtx.Pop(argv)
	case cmd == "gpush":
		gshCtx.Push(argv)
	case cmd == "history" || cmd == "hi": // hi should be alias
		gshCtx.xHistory(argv)
	case cmd == "jobs":
		gshCtx.xJobs(argv)
	case cmd == "lnsp" || cmd == "nlsp":
		gshCtx.SplitLine(argv)
	case cmd == "-ls":
		gshCtx.xFind(argv)
	case cmd == "nop":
		// do nothing
	case cmd == "pipe":
		gshCtx.xOpen(argv)
	case cmd == "plug" || cmd == "plugin" || cmd == "pin":
		gshCtx.xPlugin(argv[1:])
	case cmd == "print" || cmd == "-pr":
		// output internal slice // also sprintf should be
		gshCtx.Printv(argv)
	case cmd == "ps":
		gshCtx.xPs(argv)
	case cmd == "pstitle":
		// to be gsh.title
	case cmd == "rexecd" || cmd == "rexd":
		gshCtx.RexecServer(argv)
	case cmd == "rexec" || cmd == "rex":
		gshCtx.RexecClient(argv)
	case cmd == "repeat" || cmd == "rep": // repeat cond command
		gshCtx.repeat(argv)
	case cmd == "replay":
		gshCtx.xReplay(argv)
	case cmd == "scan":
		// scan input (or so in fscanf) to internal slice (like Files or map)
		gshCtx.Scanv(argv)
	case cmd == "set":
		// set name ...
	case cmd == "serv":
		gshCtx.httpServer(argv)
	case cmd == "shift":
		gshCtx.Shiftv(argv)
	case cmd == "sleep":
		gshCtx.sleep(argv)
	case cmd == "-sort":
		gshCtx.Sortv(argv)

	case cmd == "j" || cmd == "join":
		gshCtx.Rjoin(argv)
	case cmd == "a" || cmd == "alpa":
		gshCtx.Rexec(argv)
	case cmd == "jcd" || cmd == "jchdir":
		gshCtx.Rchdir(argv)
	case cmd == "jget":
		gshCtx.Rget(argv)
	case cmd == "jls":
		gshCtx.Rls(argv)
	case cmd == "jput":
		gshCtx.Rput(argv)
	case cmd == "jpwd":
		gshCtx.Rpwd(argv)

	case cmd == "time":
		fin = gshCtx.xTime(argv)
	case cmd == "ungets":
		if 1 < len(argv) {
			ungets(argv[1]+"\n")
		}else{
		}
	case cmd == "pwd":
		gshCtx.xPwd(argv);
	case cmd == "ver" || cmd == "-ver" || cmd == "version":
		gshCtx.showVersion(argv)
	case cmd == "where":
		// data file or so?
	case cmd == "which":
		which("PATH",argv);
	case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
		go gj_server(argv[1:]);
	case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
		go gj_server(argv[1:]);
	case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
		go gj_client(argv[1:]);
	case cmd == "gj":
		jsend(argv);
	case cmd == "jsend":
		jsend(argv);
	default:
		if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
			gshCtx.xPlugin(argv)
		}else{
			notfound,_ := gshCtx.excommand(false,argv)
			if notfound {
				fmt.Printf("--E-- command not found (%v)\n",cmd)
			}
		}
	}
	return fin
}

func (gsh*GshContext)gshelll(gline string) (rfin bool) {
	argv := strings.Split(string(gline)," ")
	fin := gsh.gshellv(argv)
	return fin
}
func (gsh*GshContext)tgshelll(gline string)(xfin bool){
	start := time.Now()
	fin := gsh.gshelll(gline)
	end := time.Now()
	elps := end.Sub(start);
	if gsh.CmdTime {
		fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + "(%d.%09ds)\n",
			elps/1000000000,elps%1000000000)
	}
	return fin
}
func Ttyid() (int) {
	fi, err := os.Stdin.Stat()
	if err != nil {
		return 0;
	}
	//fmt.Printf("Stdin: %v Dev=%d\n",
	//	fi.Mode(),fi.Mode()&os.ModeDevice)
	if (fi.Mode() & os.ModeDevice) != 0 {
		stat := syscall.Stat_t{};
		err := syscall.Fstat(0,&stat)
		if err != nil {
			//fmt.Printf("--I-- Stdin: (%v)\n",err)
		}else{
			//fmt.Printf("--I-- Stdin: rdev=%d %d\n",
			//	stat.Rdev&0xFF,stat.Rdev);
			//fmt.Printf("--I-- Stdin: tty%d\n",stat.Rdev&0xFF);
			return int(stat.Rdev & 0xFF)
		}
	}
	return 0
}
func (gshCtx *GshContext) ttyfile() string {
	//fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
	ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
		 fmt.Sprintf("%02d",gshCtx.TerminalId)
		 //strconv.Itoa(gshCtx.TerminalId)
```

```
    //fmt.Printf("--I-- ttyfile=%s\n",ttyfile)
    return ttyfile
}
func (gshCtx *GshContext) ttyline()(*os.File){
    file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
    if err != nil {
        fmt.Printf("--F-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
        return file;
    }
    return file
}
func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
    if( skipping ){
        reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
        line, _, _ := reader.ReadLine()
        return string(line)
    }else
    if true {
        return xgetline(hix,prevline,gshCtx)
    }
    /*
    else
    if( with_exgetline && gshCtx.GetLine != "" ){
        //var xhix int64 = int64(hix); // cast
        newenv := os.Environ()
        newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )

        tty := gshCtx.ttyline()
        tty.WriteString(prevline)
        Pa := os.ProcAttr {
            "", // start dir
            newenv, //os.Environ(),
            []*os.File{os.Stdin,os.Stdout,os.Stderr,tty},
            nil,
        }
//fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.GetLine)
proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
        if err != nil {
            fmt.Printf("--F-- getline process error (%v)\n",err)
            // for ; ; { }
            return "exit (getline program failed)"
        }
        //stat, err := proc.Wait()
        proc.Wait()
        buff := make([]byte,LINESIZE)
        count, err := tty.Read(buff)
        //_, err = tty.Read(buff)
        //fmt.Printf("--D-- getline (%d)\n",count)
        if err != nil {
            if ! (count == 0) { // && err.String() == "EOF" ) {
                fmt.Printf("--E-- getline error (%s)\n",err)
            }
        }else{
            //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
        }
        tty.Close()
        gline := string(buff[0:count])
        return gline
    }else
    */
    {
        // if isatty {
            fmt.Printf("!%d",hix)
            fmt.Print(PROMPT)
        // }
        reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
        line, _, _ := reader.ReadLine()
        return string(line)
    }
}

//== begin ======================================================== getline
/*
 * getline.c
 * 2020-0819 extracted from dog.c
 * getline.go
 * 2020-0822 ported to Go
 */
/*
package main // getline main
import (
    "fmt"       // <a href="https://golang.org/pkg/fmt/">fmt</a>
    "strings"   // <a href="https://golang.org/pkg/strings/">strings</a>
    "os"        // <a href="https://golang.org/pkg/os/">os</a>
    "syscall"   // <a href="https://golang.org/pkg/syscall/">syscall</a>
    //"bytes"       // <a href="https://golang.org/pkg/os/">os</a>
    //"os/exec" // <a href="https://golang.org/pkg/os/">os</a>
)
*/

// C language compatibility functions
var errno = 0
var stdin  *os.File = os.Stdin
var stdout *os.File = os.Stdout
var stderr *os.File = os.Stderr
var EOF = -1
var NULL = 0
type FILE os.File
type StrBuff []byte
var NULL_FP *os.File = nil
var NULLSP = 0
//var LINESIZE = 1024

func system(cmdstr string)(int){
    PA := syscall.ProcAttr {
        "", // the starting directory
        os.Environ(),
        []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
        nil,
    }
    argv := strings.Split(cmdstr," ")
    pid,err := syscall.ForkExec(argv[0],argv,&PA)
    if( err != nil ){
        fmt.Printf("--E-- syscall(%v) err(%v)\n",cmdstr,err)
    }
    syscall.Wait4(pid,nil,0,nil)

    /*
    argv := strings.Split(cmdstr," ")
    fmt.Fprintf(os.Stderr,"--I-- system(%v)\n",argv)
    //cmd := exec.Command(argv[0:]...)
    cmd := exec.Command(argv[0],argv[1],argv[2])
    cmd.Stdin = strings.NewReader("output of system")
    var out bytes.Buffer
    cmd.Stdout = &out
    var serr bytes.Buffer
    cmd.Stderr = &serr
    err := cmd.Run()
    if err != nil {
        fmt.Fprintf(os.Stderr,"--E-- system(%v)err(%v)\n",argv,err)
        fmt.Printf("ERR:%s\n",serr.String())
    }else{
        fmt.Printf("%s",out.String())
    }
    */
    return 0
}
func atoi(str string)(ret int){
    ret,err := fmt.Sscanf(str,"%d",ret)
    if err == nil {
        return ret
    }else{
        // should set errno
        return 0
    }
}
func getenv(name string)(string){
    val,got := os.LookupEnv(name)
    if got {
        return val
    }else{
        return "?"
    }
}
func strcpy(dst StrBuff, src string){
    var i int
    srcb := []byte(src)
    for i = 0; i < len(src) && srcb[i] != 0; i++ {
        dst[i] = srcb[i]
    }
    dst[i] = 0
}
func xstrcpy(dst StrBuff, src StrBuff){
    dst = src
}
func strcat(dst StrBuff, src StrBuff){
    dst = append(dst,src...)
}
func strdup(str StrBuff)(string){
    return string(str[0:strlen(str)])
}
func sstrlen(str string)(int){
    return len(str)
}
func strlen(str StrBuff)(int){
    var i int
    for i = 0; i < len(str) && str[i] != 0; i++ {
    }
    return i
}
func sizeof(data StrBuff)(int){
    return len(data)
}
func isatty(fd int)(ret int){
    return 1
}

func fopen(file string,mode string)(fp*os.File){
    if mode == "r" {
        fp,err := os.Open(file)
        if( err != nil ){
            fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
            return NULL_FP;
        }
        return fp;
    }else{
        fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
        if( err != nil ){
            return NULL_FP;
        }
        return fp;
    }
}
func fclose(fp*os.File){
    fp.Close()
}
func fflush(fp *os.File)(int){
    return 0
}
func fgetc(fp*os.File)(int){
    var buf [1]byte
    _,err := fp.Read(buf[0:1])
    if( err != nil ){
        return EOF;
    }else{
        return int(buf[0])
    }
}
func sfgets(str*string, size int, fp*os.File)(int){
    buf := make(StrBuff,size)
    var ch int
    var i int
    for i = 0; i < len(buf)-1; i++ {
        ch = fgetc(fp)
        //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
        if( ch == EOF ){
            break;
        }
        buf[i] = byte(ch);
        if( ch == '\n' ){
            break;
        }
    }
    buf[i] = 0
    //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
    return i
}
func fgets(buf StrBuff, size int, fp*os.File)(int){
    var ch int
    var i int
    for i = 0; i < len(buf)-1; i++ {
        ch = fgetc(fp)
        //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
        if( ch == EOF ){
            break;
        }
        buf[i] = byte(ch);
        if( ch == '\n' ){
            break;
        }
    }
    buf[i] = 0
    //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
    return i
}
func fputc(ch int , fp*os.File)(int){
    var buf [1]byte
    buf[0] = byte(ch)
    fp.Write(buf[0:1])
    return 0
}
func fputs(buf StrBuff, fp*os.File)(int){
    fp.Write(buf)
    return 0
}
func xfputss(str string, fp*os.File)(int){
    return fputs([]byte(str),fp)
}
func sscanf(str StrBuff,fmts string, params ...interface{})(int){
    fmt.Sscanf(string(str[0:strlen(str)]),fmts,params...)
    return 0
}
func fprintf(fp*os.File,fmts string, params ...interface{})(int){
    fmt.Fprintf(fp,fmts,params...)
    return 0
}

// <a name="IME">Command Line IME</a>
//---------------------------------------------------------------------- MyIME
var MyIMEVER = "MyIME/0.0.2";
type RomKana struct {
    dic string  // dictionaly ID
    pat string  // input pattern
    out string  // output pattern
    hit int64   // count of hit and used
}
var dicents = 0
var romkana [1024]RomKana
var Romkan []RomKana

func isinDic(str string)(int){
    for i,v := range Romkan {
        if v.pat == str {
            return i
        }
    }
    return -1
}
const (
    DIC_COM_LOAD = "im"
    DIC_COM_DUMP = "s"
    DIC_COM_LIST = "ls"
    DIC_COM_ENA  = "en"
    DIC_COM_DIS  = "di"
)
func helpDic(argv []string){
    out := stderr
    cmd := ""
    if 0 < len(argv) { cmd = argv[0] }
    fprintf(out,"--- %v Usage\n",cmd)
    fprintf(out,"... Commands\n")
    fprintf(out,"...   %v %-3v [dicName] [dicURL ] -- Import dictionary\n",cmd,DIC_COM_LOAD)
    fprintf(out,"...   %v %-3v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
    fprintf(out,"...   %v %-3v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
    fprintf(out,"...   %v %-3v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
    fprintf(out,"...   %v %-3v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
    fprintf(out,"... Keys ... %v\n","ESC can be used for '\\'")
    fprintf(out,"...   \\c -- Reverse the case of the last character\n",)
    fprintf(out,"...   \\i -- Replace input with translated text\n",)
    fprintf(out,"...   \\j -- On/Off translation mode\n",)
    fprintf(out,"...   \\l -- Force Lower Case\n",)
    fprintf(out,"...   \\u -- Force Upper Case (software CapsLock)\n",)
    fprintf(out,"...   \\v -- Show translation actions\n",)
    fprintf(out,"...   \\x -- Replace the last input character with it Hexa-Decimal\n",)
}
func xDic(argv[]string){
    if len(argv) <= 1 {
        helpDic(argv)
        return
    }
    argv = argv[1:]
    var debug = false
    var info = false
    var silent = false
    var dump = false
    var builtin = false
    cmd := argv[0]
    argv = argv[1:]
    opt := ""
    arg := ""

    if 0 < len(argv) {
        arg1 := argv[0]
        if arg1[0] == '-' {
            switch arg1 {
                default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
                    return
                case "-b": builtin = true
                case "-d": debug = true
                case "-s": silent = true
                case "-v": info = true
            }
            opt = arg1
            argv = argv[1:]
        }
    }

    dicName := ""
    dicURL := ""
    if 0 < len(argv) {
        arg = argv[0]
        dicName = arg
        argv = argv[1:]
    }
    if 0 < len(argv) {
        dicURL = argv[0]
        argv = argv[1:]
    }
    if false {
        fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
    }
    if cmd == DIC_COM_LOAD {
```

```
		//dicType := ""
		dicBody := ""
		if !builtin && dicName != "" && dicURL == "" {
			f,err := os.Open(dicName)
			if err == nil {
				dicURL = dicName
			}else{
				f,err = os.Open(dicName+".html")
				if err == nil {
					dicURL = dicName+".html"
				}else{
					f,err = os.Open("gshdic-"+dicName+".html")
					if err == nil {
						dicURL = "gshdic-"+dicName+".html"
					}
				}
			}
			if err == nil {
				var buf = make([]byte,128*1024)
				count,err := f.Read(buf)
				f.Close()
				if info {
					fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
				}
				dicBody = string(buf[0:count])
			}
		}
		if dicBody == "" {
			switch arg {
				default:
					dicName = "WorldDic"
					dicURL = WorldDic
					if info {
					fprintf(stderr,"--Id-- default dictionary \"%v\"\n",
						dicName);
					}
				case "wnn":
					dicName = "WnnDic"
					dicURL = WnnDic
				case "sumomo":
					dicName = "SumomoDic"
					dicURL = SumomoDic
				case "sijimi":
					dicName = "SijimiDic"
					dicURL = SijimiDic
				case "jkl":
					dicName = "JKLJaDic"
					dicURL = JA_JKLDic
			}
			if debug {
				fprintf(stderr,"--Id-- %v URL=%v\n\n",dicName,dicURL);
			}
			dicv := strings.Split(dicURL,",")
			if debug {
				fprintf(stderr,"--Id-- %v encoded data...\n",dicName)
				fprintf(stderr,"Type: %v\n",dicv[0])
				fprintf(stderr,"Body: %v\n",dicv[1])
				fprintf(stderr,"\n")
			}
			body,_ := base64.StdEncoding.DecodeString(dicv[1])
			dicBody = string(body)
		}
		if info {
			fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
			fmt.Printf("%s\n",dicBody)
		}
		if debug {
			fprintf(stderr,"--Id-- dicName %v text...\n",dicName)
			fprintf(stderr,"%v\n",string(dicBody))
		}
		entv := strings.Split(dicBody,"\n");
		if info {
			fprintf(stderr,"--Id-- %v scan...\n",dicName);
		}
		var added int = 0
		var dup int = 0
		for i,v := range entv {
			var pat string
			var out string
			fmt.Sscanf(v,"%s %s",&pat,&out)
			if len(pat) <= 0 {
			}else{
				if 0 <= isinDic(pat) {
					dup += 1
					continue
				}
				romkana[dicents] = RomKana{dicName,pat,out,0}
				dicents += 1
				added += 1
				Romkan = append(Romkan,RomKana{dicName,pat,out,0})
				if debug {
					fmt.Printf("[%3v]:[%2v]%-8v [%2v]%v\n",
						i,len(pat),pat,len(out),out)
				}
			}
		}
		if !silent {
			url := dicURL
			if strBegins(url,"data:") {
				url = "builtin"
			}
			fprintf(stderr,"--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
				dicName,added,dup,len(Romkan),url);
		}
		// should sort by pattern length for conclete match, for performance
		if debug {
			arg = "" // search pattern
			dump = true
		}
	}
	if cmd == DIC_COM_DUMP || dump {
		fprintf(stderr,"--Id-- %v dump... %v entries:\n",dicName,len(Romkan));
		var match = 0
		for i := 0; i < len(Romkan); i++ {
			dic := Romkan[i].dic
			pat := Romkan[i].pat
			out := Romkan[i].out
			if arg == "" || 0 <= strings.Index(pat,arg)||0 <= strings.Index(out,arg) {
				fmt.Printf("\\\\%v\t%v [%2v]%-8v [%2v]%v\n",
					i,dic,len(pat),pat,len(out),out)
				match += 1
			}
		}
		fprintf(stderr,"--Id-- %v matched %v / %v entries:\n",arg,match,len(Romkan));
```

```
    }
}
func loadDefaultDic(dic int){
    if( 0 < len(Romkan) ){
        return
    }
    //fprintf(stderr,"\r\n")
    xDic([]string{"dic",DIC_COM_LOAD});

    var info = false
    if info {
        fprintf(stderr,"--Id-- Conguraturations!! WorldDic is now activated.\r\n")
        fprintf(stderr,"--Id-- enter \"dic\" command for help.\r\n")
    }
}
func readDic()(int){
    /*
    var rk *os.File;
    var dic = "MyIME-dic.txt";
    //rk = fopen("romkana.txt","r");
    //rk = fopen("JK-JA-morse-dic.txt","r");
    rk = fopen(dic,"r");
    if( rk == NULL_FP ){
        if( true ){
            fprintf(stderr,"--%s-- Could not load %s\n",MyIMEVER,dic);
        }
        return -1;
    }
    if( true ){
        var di int;
        var line = make(StrBuff,1024);
        var pat string
        var out string
        for di = 0; di < 1024; di++ {
            if( fgets(line,sizeof(line),rk) == NULLSP ){
                break;
            }
            fmt.Sscanf(string(line[0:strlen(line)]),"%s %s",&pat,&out);
            //sscanf(line,"%s %[^\r\n]",&pat,&out);
            romkana[di].pat = pat;
            romkana[di].out = out;
            //fprintf(stderr,"--Dd- %-10s %s\n",pat,out)
        }
        dicents += di
        if( false ){
            fprintf(stderr,"--%s-- loaded romkana.txt [%d]\n",MyIMEVER,di);
            for di = 0; di < dicents; di++ {
                fprintf(stderr,
                    "%s %s\n",romkana[di].pat,romkana[di].out);
            }
        }
    }
    fclose(rk);

    //romkana[dicents].pat = "//ddump"
    //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
    */
    return 0;
}
func matchlen(stri string, pati string)(int){
    if strBegins(stri,pati) {
        return len(pati)
    }else{
        return 0
    }
}
func convs(src string)(string){
    var si int;
    var sx = len(src);
    var di int;
    var mi int;
    var dstb []byte

    for si = 0; si < sx; { // search max. match from the position
        if strBegins(src[si:],"%x/") {
            // %x/integer/ // s/a/b/
            ix := strings.Index(src[si+3:],"/")
            if 0 < ix {
                var iv int = 0
                //fmt.Sscanf(src[si+3:si+3+ix],"%d",&iv)
                fmt.Sscanf(src[si+3:si+3+ix],"%v",&iv)
                sval := fmt.Sprintf("%x",iv)
                bval := []byte(sval)
                dstb = append(dstb,bval...)
                si = si+3+ix+1
                continue
            }
        }
        if strBegins(src[si:],"%d/") {
            // %d/integer/ // s/a/b/
            ix := strings.Index(src[si+3:],"/")
            if 0 < ix {
                var iv int = 0
                fmt.Sscanf(src[si+3:si+3+ix],"%v",&iv)
                sval := fmt.Sprintf("%d",iv)
                bval := []byte(sval)
                dstb = append(dstb,bval...)
                si = si+3+ix+1
                continue
            }
        }
        if strBegins(src[si:],"%t") {
            now := time.Now()
            if true {
                date := now.Format(time.Stamp)
                dstb = append(dstb,[]byte(date)...)
                si = si+3
            }
            continue
        }
        var maxlen int = 0;
        var len int;
        mi = -1;
        for di = 0; di < dicents; di++ {
            len = matchlen(src[si:],romkana[di].pat);
            if( maxlen < len ){
                maxlen = len;
                mi = di;
            }
        }
        if( 0 < maxlen ){
            out := romkana[mi].out;
            dstb = append(dstb,[]byte(out)...);
            si += maxlen;
        }else{
            dstb = append(dstb,src[si])
            si += 1;
        }
    }
    return string(dstb)
}
func trans(src string)(int){
    dst := convs(src);
    xfputss(dst,stderr);
    return 0;
}

//--------------------------------------------------------------- LINEEDIT
// "?" at the top of the line means searching history

// should be compatilbe with Telnet
const (
    EV_MODE     = 255
    EV_IDLE     = 254
    EV_TIMEOUT  = 253

    GO_UP       = 252   // k
    GO_DOWN     = 251   // j
    GO_RIGHT    = 250   // l
    GO_LEFT     = 249   // h
    DEL_RIGHT   = 248   // x
    GO_TOPL     = 'A'-0x40  // 0
    GO_ENDL     = 'E'-0x40  // $

    GO_TOPW     = 239   // b
    GO_ENDW     = 238   // e
    GO_NEXTW    = 237   // w

    GO_FORWCH   = 229   // f
    GO_PAIRCH   = 228   // %

    GO_DEL      = 219   // d

    HI_SRCH_FW  = 209   // /
    HI_SRCH_BK  = 208   // ?
    HI_SRCH_RFW = 207   // n
    HI_SRCH_RBK = 206   // N
)

// should return number of octets ready to be read immediately
//fprintf(stderr,"\n--Select(%v %v)\n",err,r.Bits[0])


var EventRecvFd = -1 // file descriptor
var EventSendFd = -1
const EventFdOffset = 1000000
const NormalFdOffset = 100

func putEvent(event int, evarg int){
    if true {
        if EventRecvFd < 0 {
            var pv = []int{-1,-1}
            syscall.Pipe(pv)
            EventRecvFd = pv[0]
            EventSendFd = pv[1]
            //fmt.Printf("--De-- EventPipe created[%v,%v]\n",EventRecvFd,EventSendFd)
        }
    }else{
        if EventRecvFd < 0 {
            // the document differs from this spec
            // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
            sv,err := syscall.Socketpair(syscall.AF_UNIX,syscall.SOCK_STREAM,0)
            EventRecvFd = sv[0]
            EventSendFd = sv[1]
            if err != nil {
                fmt.Printf("--De-- EventSock created[%v,%v](%v)\n",
                    EventRecvFd,EventSendFd,err)
            }
        }
    }
    var buf = []byte{ byte(event)}
    n,err := syscall.Write(EventSendFd,buf)
    if err != nil {
        fmt.Printf("--De-- putEvent[%v](%3v)(%v %v)\n",EventSendFd,event,n,err)
    }
}
func ungets(str string){
    for _,ch := range str {
        putEvent(int(ch),0)
    }
}
func (gsh*GshContext)xReplay(argv[]string){
    hix := 0
    tempo := 1.0
    xtempo := 1.0
    repeat := 1

    for _,a := range argv { // tempo
        if strBegins(a,"x") {
            fmt.Sscanf(a[1:],"%f",&xtempo)
            tempo = 1 / xtempo
            //fprintf(stderr,"--Dr-- tempo=[%v]%v\n",a[2:],tempo);
        }else
        if strBegins(a,"r") { // repeat
            fmt.Sscanf(a[1:],"%v",&repeat)
        }else
        if strBegins(a,"!") {
            fmt.Sscanf(a[1:],"%d",&hix)
        }else{
            fmt.Sscanf(a,"%d",&hix)
        }
    }
    if hix == 0 || len(argv) <= 1 {
        hix = len(gsh.CommandHistory)-1
    }
    fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n",hix,xtempo,repeat)
        //dumpEvents(hix)
    //gsh.xScanReplay(hix,false,repeat,tempo,argv)
    go gsh.xScanReplay(hix,true,repeat,tempo,argv)
}

// <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
// 2020-0827 GShell-0.2.3
/*
func FpollIn1(fp *os.File,usec int)(uintptr){
    nfd := 1

    rdv := syscall.FdSet {}
```

```
	fd1 := fp.Fd()
	bank1 := fd1/32
	mask1 := int32(1 << fd1)
	rdv.Bits[bank1] = mask1

	fd2 := -1
	bank2 := -1
	var mask2 int32 = 0

	if 0 <= EventRecvFd {
		fd2 = EventRecvFd
		nfd = fd2 + 1
		bank2 = fd2/32
		mask2 = int32(1 << fd2)
		rdv.Bits[bank2] |= mask2
		//fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
	}

	tout := syscall.NsecToTimeval(int64(usec*1000))
	//n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
	err := syscall.Select(nfd,&rdv,nil,nil,&tout)
	if err != nil {
		//fmt.Printf("--De-- select() err(%v)\n",err)
	}
	if err == nil {
		if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
			if false {
				fmt.Printf("--De-- got Event\n")
			}
			return uintptr(EventFdOffset + fd2)
		}else
		if (rdv.Bits[bank1] & mask1) != 0 {
			return uintptr(NormalFdOffset + fd1)
		}else{
			return 1
		}
	}else{
		return 0
	}
}
*/
func fgetcTimeout1(fp *os.File,usec int)(int){
  READ1:
	//readyFd := FpollIn1(fp,usec)
	readyFd := CFpollIn1(fp,usec)
	if readyFd < 100 {
		return EV_TIMEOUT
	}

	var buf [1]byte

	if EventFdOffset <= readyFd {
		fd := int(readyFd-EventFdOffset)
		_,err := syscall.Read(fd,buf[0:1])
		if( err != nil ){
			return EOF;
		}else{
			if buf[0] == EV_MODE {
				recvEvent(fd)
				goto READ1
			}
			return int(buf[0])
		}
	}

	_,err := fp.Read(buf[0:1])
	if( err != nil ){
		return EOF;
	}else{
		return int(buf[0])
	}
}

func visibleChar(ch int)(string){
	switch {
		case '!' <= ch && ch <= '~':
			return string(ch)
	}
	switch ch {
		case ' ': return "\\s"
		case '\n': return "\\n"
		case '\r': return "\\r"
		case '\t': return "\\t"
	}
	switch ch {
		case 0x00: return "NUL"
		case 0x07: return "BEL"
		case 0x08: return "BS"
		case 0x0E: return "SO"
		case 0x0F: return "SI"
		case 0x1B: return "ESC"
		case 0x7F: return "DEL"
	}
	switch ch {
		case EV_IDLE: return fmt.Sprintf("IDLE")
		case EV_MODE: return fmt.Sprintf("MODE")
	}
	return fmt.Sprintf("%X",ch)
}
func recvEvent(fd int){
	var buf = make([]byte,1)
	_,_ = syscall.Read(fd,buf[0:1])
	if( buf[0] != 0 ){
		romkanmode = true
	}else{
		romkanmode = false
	}
}
func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
	var Start time.Time
	var events = []Event{}
	for _,e := range Events {
		if hix == 0 || e.CmdIndex == hix {
			events = append(events,e)
		}
	}
	elen := len(events)
	if 0 < elen {
		if events[elen-1].event == EV_IDLE {
			events = events[0:elen-1]
		}
	}
	for r := 0; r < repeat; r++ {
		for i,e := range events {
```

```
            nano := e.when.Nanosecond()
            micro := nano / 1000
            if Start.Second() == 0 {
                Start = time.Now()
            }
            diff := time.Now().Sub(Start)
            if replay {
                if e.event != EV_IDLE {
                    putEvent(e.event,0)
                    if e.event == EV_MODE { // event with arg
                        putEvent(int(e.evarg),0)
                    }
                }
            }else{
            fmt.Printf("%7.3fms #%-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",
                    float64(diff)/1000000.0,
                    i,
                    e.CmdIndex,
                    e.when.Format(time.Stamp),micro,
                    e.event,e.event,visibleChar(e.event),
                    float64(e.evarg)/1000000.0)
            }
            if e.event == EV_IDLE {
                d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
                //nsleep(time.Duration(e.evarg))
                nsleep(d)
            }
        }
    }
}
func dumpEvents(arg[]string){
    hix := 0
    if 1 < len(arg) {
        fmt.Sscanf(arg[1],"%d",&hix)
    }
    for i,e := range Events {
        nano := e.when.Nanosecond()
        micro := nano / 1000
        //if e.event != EV_TIMEOUT {
        if hix == 0 || e.CmdIndex == hix {
            fmt.Printf("#%-3v !%-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",i,
                e.CmdIndex,
                e.when.Format(time.Stamp),micro,
                e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
        }
        //}
    }
}
func fgetcTimeout(fp *os.File,usec int)(int){
    ch := fgetcTimeout1(fp,usec)
    if ch != EV_TIMEOUT {
        now := time.Now()
        if 0 < len(Events) {
            last := Events[len(Events)-1]
            dura := int64(now.Sub(last.when))
            Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
        }
        Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
    }
    return ch
}

var AtConsoleLineTop = true
var TtyMaxCol = 72 // to be obtained by ioctl?
var EscTimeout = (100*1000)
var (
    MODE_VicMode    bool    // vi compatible command mode
    MODE_ShowMode   bool
    romkanmode  bool    // shown translation mode, the mode to be retained
    MODE_Recursive  bool    // recursive translation
    MODE_CapsLock   bool    // software CapsLock
    MODE_LowerLock  bool    // force lower-case character lock
    MODE_ViInsert   int // visible insert mode, should be like "I" icon in X Window
    MODE_ViTrace    bool    // output newline before translation
)
type IInput struct {
    lno     int
    lastlno     int
    pch     []int   // input queue
    prompt      string
    line        string
    right       string
    inJmode     bool
    pinJmode    bool
    waitingMeta string  // waiting meta character
    LastCmd     string
}
func (iin*IInput)Getc(timeoutUs int)(int){
    ch1 := EOF
    ch2 := EOF
    ch3 := EOF
    if( 0 < len(iin.pch) ){ // deQ
        ch1 = iin.pch[0]
        iin.pch = iin.pch[1:]
    }else{
        ch1 = fgetcTimeout(stdin,timeoutUs);
    }
    if( ch1 == 033 ){ /// escape sequence
        ch2 = fgetcTimeout(stdin,EscTimeout);
        if( ch2 == EV_TIMEOUT ){
        }else{
            ch3 = fgetcTimeout(stdin,EscTimeout);
            if( ch3 == EV_TIMEOUT ){
                iin.pch = append(iin.pch,ch2) // enQ
            }else{
                switch( ch2 ){
                    default:
                        iin.pch = append(iin.pch,ch2) // enQ
                        iin.pch = append(iin.pch,ch3) // enQ
                    case '[':
                        switch( ch3 ){
                            case 'A': ch1 = GO_UP; // ^
                            case 'B': ch1 = GO_DOWN; // v
                            case 'C': ch1 = GO_RIGHT; // >
                            case 'D': ch1 = GO_LEFT; // <
                            case '3':
                ch4 := fgetcTimeout(stdin,EscTimeout);
                                if( ch4 == '~' ){
                //fprintf(stderr,"x[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
                                    ch1 = DEL_RIGHT
                                }
                        }
                    case '\\':
                //ch4 := fgetcTimeout(stdin,EscTimeout);
				//fprintf(stderr,"y[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
					switch( ch3 ){
						case '~': ch1 = DEL_RIGHT
					}
				}
			}
		}
	}
	return ch1
}
func (inn*IInput)clearline(){
	var i int
	fprintf(stderr,"\r");
	// should be ANSI ESC sequence
	for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
		fputc(' ',os.Stderr);
	}
	fprintf(stderr,"\r");
}
func (iin*IInput)Redraw(){
	redraw(iin,iin.lno,iin.line,iin.right)
}
func redraw(iin *IInput,lno int,line string,right string){
	inMeta := false
	showMode := ""
	showMeta := "" // visible Meta mode on the cursor position
	showLino := fmt.Sprintf("!%d! ",lno)
	InsertMark := "" // in visible insert mode

	if MODE_VicMode {
	}else
	if 0 < len(iin.right) {
		InsertMark = " "
	}

	if( 0 < len(iin.waitingMeta) ){
		inMeta = true
		if iin.waitingMeta[0] != 033 {
			showMeta = iin.waitingMeta
		}
	}
	if( romkanmode ){
		//romkanmark = " *";
	}else{
		//romkanmark = "";
	}
	if MODE_ShowMode {
		romkan := "--"
		inmeta := "-"
		inveri := ""
		if MODE_CapsLock {
			inmeta = "A"
		}
		if MODE_LowerLock {
			inmeta = "a"
		}
		if MODE_ViTrace {
			inveri = "v"
		}
		if MODE_VicMode {
			inveri = ":"
		}
		if romkanmode {
			romkan = "\343\201\202"
			if MODE_CapsLock {
				inmeta = "R"
			}else{
				inmeta = "r"
			}
		}
		if inMeta {
			inmeta = "\\"
		}
		showMode = "["+romkan+inmeta+inveri+"]";
	}
	Pre := "\r" + showMode + showLino
	Output := ""
	Left := ""
	Right := ""
	if romkanmode {
		Left = convs(line)
		Right = InsertMark+convs(right)
	}else{
		Left = line
		Right = InsertMark+right
	}
	Output = Pre+Left
	if MODE_ViTrace {
		Output += iin.LastCmd
	}
	Output += showMeta+Right
	for len(Output) < TtyMaxCol { // to the max. position that may be dirty
		Output += " "
		// should be ANSI ESC sequence
		// not necessary just after newline
	}
	Output += Pre+Left+showMeta // to set the cursor to the current input position
	fprintf(stderr,"%s",Output)

	if MODE_ViTrace {
		if 0 < len(iin.LastCmd) {
			iin.LastCmd = ""
			fprintf(stderr,"\r\n")
		}
	}
	AtConsoleLineTop  = false
}
// <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
func delHeadChar(str string)(rline string,head string){
	_,clen := utf8.DecodeRune([]byte(str))
	head = string(str[0:clen])
	return str[clen:],head
}
func delTailChar(str string)(rline string, last string){
	var i = 0
	var clen = 0
	for {
		_,siz := utf8.DecodeRune([]byte(str)[i:])
		if siz <= 0 { break }
		clen = siz
		i += siz
	}
	last = str[len(str)-clen:]
	return str[0:len(str)-clen],last
```

```
}

// 3> for output and history
// 4> for keylog?
// <a name="getline">Command Line Editor</a>
func xgetline(lno int, prevline string, gsh*GshContext)(string){
	var iin IInput
	iin.lastlno = lno
	iin.lno = lno

	CmdIndex = len(gsh.CommandHistory)
	if( isatty(0) == 0 ){
		if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
			iin.line = "exit\n";
		}else{
		}
		return iin.line
	}
	if( true ){
		//var pts string;
		//pts = ptsname(0);
		//pts = ttyname(0);
		//fprintf(stderr,"--pts[0] = %s\n",pts?pts:"?");
	}
	if( false ){
		fprintf(stderr,"! ");
		fflush(stderr);
		sfgets(&iin.line,LINESIZE,stdin);
		return iin.line
	}
	system("/bin/stty -echo -icanon");
	xline := iin.xgetline1(prevline,gsh)
	system("/bin/stty echo sane");
	return xline
}
func (iin*IInput)Translate(cmdch int){
	romkanmode = !romkanmode;
	if MODE_ViTrace {
		fprintf(stderr,"%v\r\n",string(cmdch));
	}else
	if( cmdch == 'J' ){
		fprintf(stderr,"J\r\n");
		iin.inJmode = true
	}
	iin.Redraw();
	loadDefaultDic(cmdch);
	iin.Redraw();
}
func (iin*IInput)Replace(cmdch int){
	iin.LastCmd = fmt.Sprintf("\\%v",string(cmdch))
	iin.Redraw();
	loadDefaultDic(cmdch);
	dst := convs(iin.line+iin.right);
	iin.line = dst
	iin.right = ""
	if( cmdch == 'I' ){
		fprintf(stderr,"I\r\n");
		iin.inJmode = true
	}
	iin.Redraw();
}
// aa 12 a1a1
func isAlpha(ch rune)(bool){
	if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
		return true
	}
	return false
}
func isAlnum(ch rune)(bool){
	if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
		return true
	}
	if '0' <= ch && ch <= '9' {
		return true
	}
	return false
}

// 0.2.8 2020-0901 created
// <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
func (iin*IInput)GotoTOPW(){
	str := iin.line
	i := len(str)
	if i <= 0 {
		return
	}
	//i0 := i
	i -= 1
	lastSize := 0
	var lastRune rune
	var found = -1
	for 0 < i { // skip preamble spaces
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if !isAlnum(lastRune) { // character, type, or string to be searched
			i -= lastSize
			continue
		}
		break
	}
	for 0 < i {
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if lastSize <= 0 { continue } // not the character top
		if !isAlnum(lastRune) { // character, type, or string to be searched
			found = i
			break
		}
		i -= lastSize
	}
	if found < 0 && i == 0 {
		found = 0
	}
	if 0 <= found {
		if isAlnum(lastRune) { // or non-kana character
		}else{ // when positioning to the top o the word
			i += lastSize
		}
		iin.right = str[i:] + iin.right
		if 0 < i {
			iin.line = str[0:i]
		}else{
			iin.line = ""
		}
	}
	//fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
	//fmt.Printf("") // set debug messae at the end of line
}
// 0.2.8 2020-0901 created
func (iin*IInput)GotoENDW(){
	str := iin.right
	if len(str) <= 0 {
		return
	}
	lastSize := 0
	var lastRune rune
	var lastW = 0
	i := 0
	inWord := false

	lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
	if isAlnum(lastRune) {
		r,z := utf8.DecodeRuneInString(str[lastSize:])
		if 0 < z && isAlnum(r) {
			inWord = true
		}
	}
	for i < len(str) {
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if lastSize <= 0 { break } // broken data?
		if !isAlnum(lastRune) { // character, type, or string to be searched
			break
		}
		lastW = i // the last alnum if in alnum word
		i += lastSize
	}
	if inWord {
		goto DISP
	}
	for i < len(str) {
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if lastSize <= 0 { break } // broken data?
		if isAlnum(lastRune) { // character, type, or string to be searched
			break
		}
		i += lastSize
	}
	for i < len(str) {
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if lastSize <= 0 { break } // broken data?
		if !isAlnum(lastRune) { // character, type, or string to be searched
			break
		}
		lastW = i
		i += lastSize
	}
DISP:
	if 0 < lastW {
		iin.line = iin.line + str[0:lastW]
		iin.right = str[lastW:]
	}
	//fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
	//fmt.Printf("") // set debug messae at the end of line
}
// 0.2.8 2020-0901 created
func (iin*IInput)GotoNEXTW(){
	str := iin.right
	if len(str) <= 0 {
		return
	}
	lastSize := 0
	var lastRune rune
	var found = -1
	i := 1
	for i < len(str) {
		lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
		if lastSize <= 0 { break } // broken data?
		if !isAlnum(lastRune) { // character, type, or string to be searched
			found = i
			break
		}
		i += lastSize
	}
	if 0 < found {
		if isAlnum(lastRune) { // or non-kana character
		}else{ // when positioning to the top o the word
			found += lastSize
		}
		iin.line = iin.line + str[0:found]
		if 0 < found {
			iin.right = str[found:]
		}else{
			iin.right = ""
		}
	}
	//fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
	//fmt.Printf("") // set debug messae at the end of line
}
// 0.2.8 2020-0902 created
func (iin*IInput)GotoPAIRCH(){
	str := iin.right
	if len(str) <= 0 {
		return
	}
	lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
	if lastSize <= 0 {
		return
	}
	forw := false
	back := false
	pair := ""
	switch string(lastRune){
		case "{": pair = "}"; forw = true
		case "}": pair = "{"; back = true
		case "(": pair = ")"; forw = true
		case ")": pair = "("; back = true
		case "[": pair = "]"; forw = true
		case "]": pair = "["; back = true
		case "<": pair = ">"; forw = true
		case ">": pair = "<"; back = true
		case "\"": pair = "\""; // context depednet, can be f" or back-double quote
		case "'": pair = "'"; // context depednet, can be f' or back-quote
		// case Japanese Kakkos
	}
	if forw {
		iin.SearchForward(pair)
	}
	if back {
		iin.SearchBackward(pair)
	}
```

```
}
// 0.2.8 2020-0902 created
func (iin*IInput)SearchForward(pat string)(bool){
	right := iin.right
	found := -1
	i := 0
	if strBegins(right,pat) {
		_,z := utf8.DecodeRuneInString(right[i:])
		if 0 < z {
			i += z
		}
	}
	for i < len(right) {
		if strBegins(right[i:],pat) {
			found = i
			break
		}
		_,z := utf8.DecodeRuneInString(right[i:])
		if z <= 0 { break }
		i += z
	}
	if 0 <= found {
		iin.line = iin.line + right[0:found]
		iin.right = iin.right[found:]
		return true
	}else{
		return false
	}
}
// 0.2.8 2020-0902 created
func (iin*IInput)SearchBackward(pat string)(bool){
	line := iin.line
	found := -1
	i := len(line)-1
	for i = i; 0 <= i; i-- {
		_,z := utf8.DecodeRuneInString(line[i:])
		if z <= 0 {
			continue
		}
		//fprintf(stderr,"-- %v %v\n",pat,line[i:])
		if strBegins(line[i:],pat) {
			found = i
			break
		}
	}
	//fprintf(stderr,"--%d\n",found)
	if 0 <= found {
		iin.right = line[found:] + iin.right
		iin.line = line[0:found]
		return true
	}else{
		return false
	}
}
// 0.2.8 2020-0902 created
// search from top, end, or current position
func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
	if forw {
		for _,v := range gsh.CommandHistory {
			if 0 <= strings.Index(v.CmdLine,pat) {
				//fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
				return true,v.CmdLine
			}
		}
	}else{
		hlen := len(gsh.CommandHistory)
		for i := hlen-1; 0 < i ; i-- {
			v := gsh.CommandHistory[i]
			if 0 <= strings.Index(v.CmdLine,pat) {
				//fprintf(stderr,"\n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
				return true,v.CmdLine
			}
		}
	}
	//fprintf(stderr,"\n--De-- not-found(%v)\n",pat)
	return false,"(Not Found in History)"
}
// 0.2.8 2020-0902 created
func (iin*IInput)GotoFORWSTR(pat string,gsh*GshContext){
	found := false
	if 0 < len(iin.right) {
		found = iin.SearchForward(pat)
	}
	if !found {
		found,line := gsh.SearchHistory(pat,true)
		if found {
			iin.line = line
			iin.right = ""
		}
	}
}
func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
	found := false
	if 0 < len(iin.line) {
		found = iin.SearchBackward(pat)
	}
	if !found {
		found,line := gsh.SearchHistory(pat,false)
		if found {
			iin.line = line
			iin.right = ""
		}
	}
}
func (iin*IInput)getstring1(prompt string)(string){ // should be editable
	iin.clearline();
	fprintf(stderr,"\r%v",prompt)
	str := ""
	for {
		ch := iin.Getc(10*1000*1000)
		if ch == '\n' || ch == '\r' {
			break
		}
		sch := string(ch)
		str += sch
		fprintf(stderr,"%s",sch)
	}
	return str
}

// search pattern must be an array and selectable with ^N/^P
var SearchPat = ""
var SearchForw = true

```

```
func (iin*IInput)xgetline1(prevline string, gsh*GshContext)(string){
	var ch int;

	MODE_ShowMode = false
	MODE_VicMode = false
	iin.Redraw();
	first := true

	for cix := 0; ; cix++ {
		iin.pinJmode = iin.inJmode
		iin.inJmode = false

		ch = iin.Getc(1000*1000)

		if ch != EV_TIMEOUT && first {
			first = false
			mode := 0
			if romkanmode {
				mode = 1
			}
			now := time.Now()
			Events = append(Events,Event{now,EV_MODE,int64(mode),CmdIndex})
		}
		if ch == 033 {
			MODE_ShowMode = true
			MODE_VicMode = !MODE_VicMode
			iin.Redraw();
			continue
		}
		if MODE_VicMode {
			switch ch {
				case '0': ch = GO_TOPL
				case '$': ch = GO_ENDL
				case 'b': ch = GO_TOPW
				case 'e': ch = GO_ENDW
				case 'w': ch = GO_NEXTW
				case '%': ch = GO_PAIRCH

				case 'j': ch = GO_DOWN
				case 'k': ch = GO_UP
				case 'h': ch = GO_LEFT
				case 'l': ch = GO_RIGHT
				case 'x': ch = DEL_RIGHT
				case 'a': MODE_VicMode = !MODE_VicMode
					ch = GO_RIGHT
				case 'i': MODE_VicMode = !MODE_VicMode
					iin.Redraw();
					continue
				case '~':
					right,head := delHeadChar(iin.right)
					if len([]byte(head)) == 1 {
						ch = int(head[0])
						if( 'a' <= ch && ch <= 'z' ){
							ch = ch + 'A'-'a'
						}else
						if( 'A' <= ch && ch <= 'Z' ){
							ch = ch + 'a'-'A'
						}
						iin.right = string(ch) + right
					}
					iin.Redraw();
					continue
				case 'f': // GO_FORWCH
					iin.Redraw();
					ch = iin.Getc(3*1000*1000)
					if ch == EV_TIMEOUT {
						iin.Redraw();
						continue
					}
					SearchPat = string(ch)
					SearchForw = true
					iin.GotoFORWSTR(SearchPat,gsh)
					iin.Redraw();
					continue
				case '/':
					SearchPat = iin.getstring1("/") // should be editable
					SearchForw = true
					iin.GotoFORWSTR(SearchPat,gsh)
					iin.Redraw();
					continue
				case '?':
					SearchPat = iin.getstring1("?") // should be editable
					SearchForw = false
					iin.GotoBACKSTR(SearchPat,gsh)
					iin.Redraw();
					continue
				case 'n':
					if SearchForw {
						iin.GotoFORWSTR(SearchPat,gsh)
					}else{
						iin.GotoBACKSTR(SearchPat,gsh)
					}
					iin.Redraw();
					continue
				case 'N':
					if !SearchForw {
						iin.GotoFORWSTR(SearchPat,gsh)
					}else{
						iin.GotoBACKSTR(SearchPat,gsh)
					}
					iin.Redraw();
					continue
			}
		}
		switch ch {
			case GO_TOPW:
				iin.GotoTOPW()
				iin.Redraw();
				continue
			case GO_ENDW:
				iin.GotoENDW()
				iin.Redraw();
				continue
			case GO_NEXTW:
				// to next space then
				iin.GotoNEXTW()
				iin.Redraw();
				continue
			case GO_PAIRCH:
				iin.GotoPAIRCH()
				iin.Redraw();
				continue
		}

```

```
        //fprintf(stderr,"A[%02X]\n",ch);
        if( ch == '\\' || ch == 033 ){
            MODE_ShowMode = true
            metach := ch
            iin.waitingMeta = string(ch)
            iin.Redraw();
                // set cursor //fprintf(stderr,"???\b\b\b")
            ch = fgetcTimeout(stdin,2000*1000)
                // reset cursor
            iin.waitingMeta = ""

            cmdch := ch
            if( ch == EV_TIMEOUT ){
                if metach == 033 {
                    continue
                }
                ch = metach
            }else
            /*
            if( ch == 'm' || ch == 'M' ){
                mch := fgetcTimeout(stdin,1000*1000)
                if mch == 'r' {
                    romkanmode = true
                }else{
                    romkanmode = false
                }
                continue
            }else
            */
            if( ch == 'k' || ch == 'K' ){
                MODE_Recursive = !MODE_Recursive
                iin.Translate(cmdch);
                continue
            }else
            if( ch == 'j' || ch == 'J' ){
                iin.Translate(cmdch);
                continue
            }else
            if( ch == 'i' || ch == 'I' ){
                iin.Replace(cmdch);
                continue
            }else
            if( ch == 'l' || ch == 'L' ){
                MODE_LowerLock = !MODE_LowerLock
                MODE_CapsLock = false
                if MODE_ViTrace {
                    fprintf(stderr,"%v\r\n",string(cmdch));
                }
                iin.Redraw();
                continue
            }else
            if( ch == 'u' || ch == 'U' ){
                MODE_CapsLock = !MODE_CapsLock
                MODE_LowerLock = false
                if MODE_ViTrace {
                    fprintf(stderr,"%v\r\n",string(cmdch));
                }
                iin.Redraw();
                continue
            }else
            if( ch == 'v' || ch == 'V' ){
                MODE_ViTrace = !MODE_ViTrace
                if MODE_ViTrace {
                    fprintf(stderr,"%v\r\n",string(cmdch));
                }
                iin.Redraw();
                continue
            }else
            if( ch == 'c' || ch == 'C' ){
                if 0 < len(iin.line) {
                    xline,tail := delTailChar(iin.line)
                    if len([]byte(tail)) == 1 {
                        ch = int(tail[0])
                        if( 'a' <= ch && ch <= 'z' ){
                            ch = ch + 'A'-'a'
                        }else
                        if( 'A' <= ch && ch <= 'Z' ){
                            ch = ch + 'a'-'A'
                        }
                        iin.line = xline + string(ch)
                    }
                }
                if MODE_ViTrace {
                    fprintf(stderr,"%v\r\n",string(cmdch));
                }
                iin.Redraw();
                continue
            }else{
                iin.pch = append(iin.pch,ch) // push
                ch = '\\'
            }
        }
        switch( ch ){
            case 'P'-0x40: ch = GO_UP
            case 'N'-0x40: ch = GO_DOWN
            case 'B'-0x40: ch = GO_LEFT
            case 'F'-0x40: ch = GO_RIGHT
        }
        //fprintf(stderr,"B[%02X]\n",ch);
        switch( ch ){
            case 0:
                continue;

            case '\t':
                iin.Replace('j');
                continue
            case 'X'-0x40:
                iin.Replace('j');
                continue

            case EV_TIMEOUT:
                iin.Redraw();
                if iin.pinJmode {
                    fprintf(stderr,"\\J\r\n")
                    iin.inJmode = true
                }
                continue
            case GO_UP:
                if iin.lno == 1 {
                    continue
                }
                cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
                if ok {
                    iin.line = cmd
```

```
					iin.right = ""
					iin.lno = iin.lno - 1
				}
				iin.Redraw();
				continue
			case GO_DOWN:
				cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
				if ok {
					iin.line = cmd
					iin.right = ""
					iin.lno = iin.lno + 1
				}else{
					iin.line = ""
					iin.right = ""
					if iin.lno == iin.lastlno-1 {
						iin.lno = iin.lno + 1
					}
				}
				iin.Redraw();
				continue
			case GO_LEFT:
				if 0 < len(iin.line) {
					xline,tail := delTailChar(iin.line)
					iin.line = xline
					iin.right = tail + iin.right
				}
				iin.Redraw();
				continue;
			case GO_RIGHT:
				if( 0 < len(iin.right) && iin.right[0] != 0 ){
					xright,head := delHeadChar(iin.right)
					iin.right = xright
					iin.line += head
				}
				iin.Redraw();
				continue;
			case EOF:
				goto EXIT;
			case 'R'-0x40: // replace
				dst := convs(iin.line+iin.right);
				iin.line = dst
				iin.right = ""
				iin.Redraw();
				continue;
			case 'T'-0x40: // just show the result
				readDic();
				romkanmode = !romkanmode;
				iin.Redraw();
				continue;
			case 'L'-0x40:
				iin.Redraw();
				continue
			case 'K'-0x40:
				iin.right = ""
				iin.Redraw();
				continue
			case 'E'-0x40:
				iin.line += iin.right
				iin.right = ""
				iin.Redraw();
				continue
			case 'A'-0x40:
				iin.right = iin.line + iin.right
				iin.line = ""
				iin.Redraw();
				continue
			case 'U'-0x40:
				iin.line = ""
				iin.right = ""
				iin.clearline();
				iin.Redraw();
				continue;
			case DEL_RIGHT:
				if( 0 < len(iin.right) ){
					iin.right,_ = delHeadChar(iin.right)
					iin.Redraw();
				}
				continue;
			case 0x7F: // BS? not DEL
				if( 0 < len(iin.line) ){
					iin.line,_ = delTailChar(iin.line)
					iin.Redraw();
				}
				/*
				else
				if( 0 < len(iin.right) ){
					iin.right,_ = delHeadChar(iin.right)
					iin.Redraw();
				}
				*/
				continue;
			case 'H'-0x40:
				if( 0 < len(iin.line) ){
					iin.line,_ = delTailChar(iin.line)
					iin.Redraw();
				}
				continue;
		}
		if( ch == '\n' || ch == '\r' ){
			iin.line += iin.right;
			iin.right = ""
			iin.Redraw();
			fputc(ch,stderr);
			AtConsoleLineTop  = true
			break;
		}
		if MODE_CapsLock {
			if 'a' <= ch && ch <= 'z' {
				ch = ch+'A'-'a'
			}
		}
		if MODE_LowerLock {
			if 'A' <= ch && ch <= 'Z' {
				ch = ch+'a'-'A'
			}
		}
		iin.line += string(ch);
		iin.Redraw();
	}
EXIT:
	return iin.line + iin.right;
}

func getline_main(){
    line := xgetline(0,"",nil)
    fprintf(stderr,"%s\n",line);
/*
    dp = strpbrk(line,"\r\n");
    if( dp != NULL ){
        *dp = 0;
    }

    if( 0 ){
        fprintf(stderr,"\n(%d)\n",int(strlen(line)));
    }
    if( lseek(3,0,0) == 0 ){
        if( romkanmode ){
            var buf [8*1024]byte;
            convs(line,buff);
            strcpy(line,buff);
        }
        write(3,line,strlen(line));
        ftruncate(3,lseek(3,0,SEEK_CUR));
        //fprintf(stderr,"outsize=%d\n",(int)lseek(3,0,SEEK_END));
        lseek(3,0,SEEK_SET);
        close(3);
    }else{
        fprintf(stderr,"\r\ngotline: ");
        trans(line);
        //printf("%s\n",line);
        printf("\n");
    }
*/
}
//== end ========================================================= getline

//
// $USERHOME/.gsh/
//      gsh-rc.txt, or gsh-configure.txt
//              gsh-history.txt
//              gsh-aliases.txt // should be conditional?
//
func (gshCtx *GshContext)gshSetupHomedir()(bool) {
    homedir,found := userHomeDir()
    if !found {
        fmt.Printf("--E-- You have no UserHomeDir\n")
        return true
    }
    gshhome := homedir + "/" + GSH_HOME
    _, err2 := os.Stat(gshhome)
    if err2 != nil {
        err3 := os.Mkdir(gshhome,0700)
        if err3 != nil {
            fmt.Printf("--E-- Could not Create %s (%s)\n",
                gshhome,err3)
            return true
        }
        fmt.Printf("--I-- Created %s\n",gshhome)
    }
    gshCtx.GshHomeDir = gshhome
    return false
}
func setupGshContext()(GshContext,bool){
    gshPA := syscall.ProcAttr {
        "", // the staring directory
        os.Environ(), // environ[]
        []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
        nil, // OS specific
    }
    cwd, _ := os.Getwd()
    gshCtx := GshContext {
        cwd, // StartDir
        "", // GetLine
        []GChdirHistory { {cwd,time.Now(),0} }, // ChdirHistory
        gshPA,
        []GCommandHistory{}, //something for invokation?
        GCommandHistory{}, // CmdCurrent
        false,
        []int{},
        syscall.Rusage{},
        "", // GshHomeDir
        Ttyid(),
        false,
        false,
        []PluginInfo{},
        []string{},
        " ",
        "v",
        ValueStack{},
        GServer{"",""}, // LastServer
        "", // RSERV
        cwd, // RWD
        CheckSum{},
    }
    err := gshCtx.gshSetupHomedir()
    return gshCtx, err
}
func (gsh*GshContext)gshelllh(gline string)(bool){
    ghist := gsh.CmdCurrent
    ghist.WorkDir,_ = os.Getwd()
    ghist.WorkDirX = len(gsh.ChdirHistory)-1
    //fmt.Printf("--D--ChdirHistory(@%d)\n",len(gsh.ChdirHistory))
    ghist.StartAt = time.Now()
    rusagev1 := Getrusagev()
    gsh.CmdCurrent.FoundFile = []string{}
    fin := gsh.tgshelll(gline)
    rusagev2 := Getrusagev()
    ghist.Rusagev = RusageSubv(rusagev2,rusagev1)
    ghist.EndAt = time.Now()
    ghist.CmdLine = gline
    ghist.FoundFile = gsh.CmdCurrent.FoundFile

    /* record it but not show in list by default
    if len(gline) == 0 {
        continue
    }
    if gline == "hi" || gline == "history" { // don't record it
        continue
    }
    */
    gsh.CommandHistory = append(gsh.CommandHistory, ghist)
    return fin
}
// <a name="main">Main loop</a>
func script(gshCtxGiven *GshContext) (_ GshContext) {
    gshCtxBuf,err0 := setupGshContext()
    if err0 {
        return gshCtxBuf;
```

```
	}
	gshCtx := &gshCtxBuf

	//fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
	//resmap()

	/*
	if false {
		gsh_getlinev, with_exgetline :=
			which("PATH",[]string{"which","gsh-getline","-s"})
		if with_exgetline {
			gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
			gshCtx.GetLine = toFullpath(gsh_getlinev[0])
		}else{
		fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
		}
	}
	*/

	ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
	gshCtx.CommandHistory = append(gshCtx.CommandHistory,ghist0)

	prevline := ""
	skipping := false
	for hix := len(gshCtx.CommandHistory); ; {
		gline := gshCtx.getline(hix,skipping,prevline)
		if skipping {
			if strings.Index(gline,"fi") == 0 {
				fmt.Printf("fi\n");
				skipping = false;
			}else{
				//fmt.Printf("%s\n",gline);
			}
			continue
		}
		if strings.Index(gline,"if") == 0 {
			//fmt.Printf("--D-- if start: %s\n",gline);
			skipping = true;
			continue
		}
		if false {
			os.Stdout.Write([]byte("gotline:"))
			os.Stdout.Write([]byte(gline))
			os.Stdout.Write([]byte("\n"))
		}
		gline = strsubst(gshCtx,gline,true)
		if false {
			fmt.Printf("fmt.Printf %%v - %v\n",gline)
			fmt.Printf("fmt.Printf %%s - %s\n",gline)
			fmt.Printf("fmt.Printf %%x - %s\n",gline)
			fmt.Printf("fmt.Printf %%U - %s\n",gline)
			fmt.Printf("Stouut.Write -")
			os.Stdout.Write([]byte(gline))
			fmt.Printf("\n")
		}
		/*
		// should be cared in substitution ?
		if 0 < len(gline) && gline[0] == '!' {
			xgline, set, err := searchHistory(gshCtx,gline)
			if err {
				continue
			}
			if set {
				// set the line in command line editor
			}
			gline = xgline
		}
		*/
		fin := gshCtx.gshelllh(gline)
		if fin {
			break;
		}
		prevline = gline;
		hix++;
	}
	return *gshCtx
}
func main() {
	gshCtxBuf := GshContext{}
	gsh := &gshCtxBuf
	argv := os.Args

	if( isin("wss",argv) ){
		gj_server(argv[1:]);
		return;
	}
	if( isin("wsc",argv) ){
		gj_client(argv[1:]);
		return;
	}
	if 1 < len(argv) {
		if isin("version",argv){
			gsh.showVersion(argv)
			return
		}
		if argv[1] == "gj" {
			if argv[2] == "listen" { go gj_server(argv[2:]); }
			if argv[2] == "server" { go gj_server(argv[2:]); }
			if argv[2] == "serve"  { go gj_server(argv[2:]); }
			if argv[2] == "client" { go gj_client(argv[2:]); }
			if argv[2] == "join"   { go gj_client(argv[2:]); }
		}
		comx := isinX("-c",argv)
		if 0 < comx {
			gshCtxBuf,err := setupGshContext()
			gsh := &gshCtxBuf
			if !err {
				gsh.gshellv(argv[comx+1:])
			}
			return
		}
	}
	if 1 < len(argv) && isin("-s",argv) {
	}else{
		gsh.showVersion(append(argv,[]string{"-l","-a"}...))
	}
	script(nil)
	//gshCtx := script(nil)
	//gshell(gshCtx,"time")
}

//</div></details>
//<details id="gsh-todo"><summary>Considerations</summary><div class="gsh-src">
// - inter gsh communication, possibly running in remote hosts -- to be remote shell
// - merged histories of multiple parallel gsh sessions
// - alias as a function or macro
// - instant alias end environ export to the permanent > ~/.gsh/gsh-alias and gsh-environ
// - retrieval PATH of files by its type
// - gsh as an IME with completion using history and file names as dictionaies
// - gsh a scheduler in precise time of within a millisecond
// - all commands have its subucomand after "---" symbol
// - filename expansion by "-find" command
// - history of ext code and output of each commoand
// - "script" output for each command by pty-tee or telnet-tee
// - $BUILTIN command in PATH to show the priority
// - "?" symbol in the command (not as in arguments) shows help request
// - searching command with wild card like: which ssh-*
// - longformat prompt after long idle time (should dismiss by BS)
// - customizing by building plugin and dynamically linking it
// - generating syntactic element like "if" by macro expansion (like CPP) >> alias
// - "!" symbol should be used for negation, don't wast it just for job control
// - don't put too long output to tty, record it into GSH_HOME/session-id/comand-id.log
// - making canonical form of command at the start adding quatation or white spaces
// - name(a,b,c) ... use "(" and ")" to show both delimiter and realm
// - name? or name! might be useful
// - htar format - packing directory contents into a single html file using data scheme
// - filepath substitution shold be done by each command, expecially in case of builtins
// - @N substition for the history of working directory, and @spec for more generic ones
// - @dir prefix to do the command at there, that means like (chdir @dir; command)
// - GSH_PATH for plugins
// - standard command output: list of data with name, size, resouce usage, modified time
// - generic sort key option -nm name, -sz size, -ru rusage, -ts start-time, -tm mod-time
//   -wc word-count, grep match line count, ...
// - standard command execution result: a list of string, -tm, -ts, -ru, -sz, ...
// - -tailf-filename like tail -f filename, repeat close and open before read
// - max. size and max. duration and timeout of (generated) data transfer
// - auto. numbering, aliasing, IME completion of file name (especially rm of quieer name)
// - IME "?" at the top of the command line means searching history
// - IME %d/0x10000/ %x/ffff/
// - IME ESC to go the edit mode like in vi, and use :command as :s/x/y/g to edit history
// - gsh in WebAssembly
// - gsh as a HTTP server of online-manual
//---END--- (^-^)//ITS more</div></details>

//<span class="gsh-golang-data">

var WorldDic = //<span id="gsh-world-dic">
"data:text/dic;base64,"+
"Ly8gTXlJTUUvMC4wLjEg6L6e5pu4ICgyMDIwLTA4MTlhKQpzZWthaSDkuJbnlYwKa28g44GT"+
"Cm5uIOOCkwpuaSDjgasKY2hpIOOBoQp0aSDjgaEKaGEg44GvCnNlIOOBmwprYSDjgYsKaSDj"+
"gYQK";
//</span>

var WnnDic = //<span id="gsh-wnn-dic">
"data:text/dic;base64,"+
"PG1ldGEgY2hhcnNldD0iVVRGLTgiPgo8dGV4dGFyZWEgY29scz04MCByb3dzPTQwPgovL2Rp"+
"Y3ZlcglHU2hlbGxcc0lNRVxzZGljdGlvbmFyeVxzZm9yXHNXbm5ccy8vXHMyMDIwLTA4MzAK"+
"R1NoZWxsCUdTaGVsbArjgo/jgZ/jgZcJ56eBCndhdGFzaGkJ56eBCndhdGFzaQnnp4EK44Gq"+
"44G+44GICeWQjeWJjQpuYW1hZQnlkI3liY0K44Gq44GL44GuCeS4remHjgpuYWthbm8J5Lit"+
"6YeOCndhCeOCjwp0YQnjgZ8Kc2kJ44GXCnNoaQnjgZcKbm8J44GuCm5hCeOBqgptYQnjgb4K"+
"ZQnjgYgKaGEJ44GvCm5hCeOBqgprYQnjgYsKbm8J44GuCmRlCeOBpwpzdQnjgZkKZVxzCWVj"+
"aG8KZGljCWRpYwplY2hvCWVjaG8KcmVwbGF5CXJlcGxheQpyZXBlYXQJcmVwZWF0CmR0CWRh"+
"dGVccysnJVklbSVkLSVIOiVNOiVTJwp0aW9uCXRpb24KJXQJJXQJLy8gdG8gYmUgYW4gYWN0"+
"aW9uCjwvdGV4dGFyZWE+Cg=="
//</span>

var SumomoDic = //<span id="gsh-sumomo-dic">
"data:text/dic;base64,"+
"PG1ldGEgY2hhcnNldD0iVVRGLTgiPgo8dGV4dGFyZWEgY29scz04MCByb3dzPTQwPgovL3Zl"+
"cglHU2hlbGxcc0lNRVxzZGljdGlvbmFyeVxzZm9yXHNTdW1vbW9ccy8vXHMyMDIwLTA4MzAK"+
"c3UJ44GZCm1vCeOCggpubwnjga4KdQnjgYYKY2hpCeOBoQp0aQnjgaEKdWNoaQnlhoUKdXRp"+
"CeWGhQpzdW1vbW8J44GZ44KC44KCCnN1bW9tb21vCeOBmeOCguOCguOCggptb21vCeahgwpt"+
"b21vbW8J5qGD44KCCiwsCeOAgQouLgnjgIIKPC90ZXh0YXJlYT4K"
//</span>

var SijimiDic = //<span id="gsh-sijimi-dic">
"data:text/dic;base64,"+
"PG1ldGEgY2hhcnNldD0iVVRGLTgiPgo8dGV4dGFyZWEgY29scz04MCByb3dzPTQwPgovL3Zl"+
"cglHU2hlbGxcc0lNRVxzZGljdGlvbmFyeVxzZm9yXHNTaGlqaW1pXHMvL1xzMjAyMC0wODMw"+
"CnNpCeOBlwpzaGkJ44GXCmppCeOBmAptaQnjgb8KbmEJ44GqCmp1CeOBmOOChQp4eXUJ44KF"+
"CnUJ44GGCm5pCeOBqwprbwnjgZMKYnUJ44G2Cm5uCeOCkwpubwnjga4KY2hpCeOBoQp0aQnj"+
"gaEKa2EJ44GLCnJhCeOCiQosLAnjgIEKLi4J44CCCnhuYW5hCeS4gwp4anV1CeWNgQp4bmkJ"+
"5LqMCmtveAnlgIsKa29xCeWAiwprb3gJ5YCLCm5hbmFqdXVuaXgJNzIKbmFuYWp1dW5peHgJ"+
"77yX77ySCm5hbmFqdXVuaVgJ77yX77ySCuS4g+WNgeS6jHgJNzIKa29idW5uCeWAi+WIhgp0"+
"aWthcmFxCeOBoeOBi+OCiQp0aWthcmEJ5YqbCmNoaWthcmEJ5YqbCjwvdGV4dGFyZWE+Cg="
//</span>

var JA_JKLDic = //<span id="gsh-ja-jkl-dic">
"data:text/dic;base64,"+
"Ly92ZXJsCU15SU1FamRpY2ptb3JzZWpKQWpKS0woMjAyMGowODE5KSheLV4pL1NhdG94SVRT"+
"CmtqamprbGtqa2tsa2psIOS4lueVjApqamtqamwJ44GCCmtqbAnjgYQKa2tqbAnjgYYKamtq"+
"amwJ44GICmtqa2trbAnjgYoKa2pra2wJ44GLCmpramtrbAnjgY0Ka2tramwJ44GPCmpramps"+
"CeOBkQpqampqbAnjgZMKamtqa2psCeOBlQpqamtqa2wJ44GXCmpqamtqbAnjgZkKa2pqamts"+
"CeOBmwpqamprbAnjgZ0KamtsCeOBnwpra2prbAnjgaEKa2pqa2wJ44GkCmtqa2pqbAnjgaYK"+
"a2tqa2tsCeOBqApramtsCeOBqgpqa2prbAnjgasKa2tra2wJ44GsCmpqa2psCeOBrQpra2pq"+
"bAnjga4Kamtra2wJ44GvCmpqa2tqbAnjgbIKampra2wJ44G1CmtsCeOBuApqa2tsCeOBuwpq"+
"a2tqbAnjgb4Ka2tqa2psCeOBvwpqbAnjgoAKamtra2psCeOCgQpqa2tqa2wJ44KCCmtqamwJ"+
"44KECmpra2pqbAnjgoYKampsCeOCiApra2tsCeOCiQpqamtsCeOCigpqa2pqa2wJ44KLCmpq"+
"amwJ44KMCmtqa2psCeOCjQpqa2psCeOCjwpramtramwJ44KQCmtqamtrbAnjgpEKa2pqamwJ"+
"44KSCmtqa2prbAnjgpMKa2pqa2psCeODvApra2wJ44KbCmtramprbAnjgpwKa2pramtqbAnj"+
"gIEK";
//</span>

//</span>
/*
<style id="gsh-references-style">
#references details { font-family:Georgia; }
#references a { font-family:Georgia; }
.wrap { white-space:normal; }
</style>
<details id="references"><summary>References</summary><div class="gsh-src">
Web technology
  <a href="https://html.spec.whatwg.org">HTML: The Living Standard</a> (September 2020)
    <a href="https://html.spec.whatwg.org/dev/">Developer Version</a>

  <a href="https://drafts.csswg.org">CSS Working Group Editor Drafts</a> (September 2020)

  <a href="https://developer.mozilla.org/ja/docs/Web">MDN web docs</a>
    <a href="https://developer.mozilla.org/ja/docs/Web/HTML/Element">HTML</a>
    <a href="https://developer.mozilla.org/en-US/docs/Web/CSS">CSS</a> : <span class="wrap">
     <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Selectors">selectors</a>
     <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/background-repeat">repeat</a></span>
    <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
    <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
    <a href="https://mdn-web-dna.s3-us-west-2.amazonaws.com/MDN-Browser-Compatibility-Report-2020.pdf">MDN Browser Compatibility Report (2020)</a>

Go language (August 2020 / Go 1.15)
  <a href="https://golang.org">The Go Programming Language</a>
```

```
    <a href="https://golang.org/pkg/">Packages</a>
      <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>

<a href="https://stackoverflow.com/">Stackoverflow</a>
<!--
<iframe src="https://golang.org" width="100%" height="300"></iframe>
-->
</div></details>
*/
/*
<details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>

<!-- h2>The full of this HTML including the Go code is here.</h2 -->
<details id="gsh-whole-view"><summary>Whole file</summary>
 <a name="whole-src-view"></a>
 <span id="src-frame"></span><!-- a window to show source code -->
</details>

<details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
 <a name="style-src-view"></a>
 <span id="gsh-style-view"></span>
</details>

<details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
 <a name="script-src-view"></a>
 <span id="gsh-script-view"></span>
</details>

<details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
 <a name="gsh-data-frame"></a>
 <span id="gsh-data-view"></span>
</details>

</div></details>
*/

/*
<div id="GshFooter0"></div>
<!-- 2020-09-17 SatoxITS, visible script { -- >
<details><summary>GJScript</summary>
<style>.gjscript { font-family:Georgia; }</style>
<pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJScript!'); }
  gjtest1()
</pre>
<script>
  gjs = document.getElementById('gjscript_1');
  //eval(gjs.innerHTML);
  //gjs.outerHTML = ""
</script>
</details><!-- ----------- END-OF-VISIBLE-PART ----------- } -->

<!--
// 2020-0906 added,
https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
https://developer.mozilla.org/en-US/docs/Web/CSS/position
-->
<span id="GshGrid">(^_^)//<small>{Hit j k l h}</small></span>

<span id="GStat"><br>
</span>
<span id="GMenu" onclick="GShellMenu(this)"></span>
<span id="GTop"></span>
<div id="GShellPlane" onclick="showGShellPlane();"></div>
<div id="RawTextViewer"></div>
<div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>

<style id="GshStyleDef">
 #LineNumbered table,tr,td {
   margin:0;
   padding:4px;
   spacing:0;
   border:12px;
 }
 textarea.LineNumber {
   font-size:12px;
   font-family:monospace,Courier New;
   color:#282;
   padding:4px;
   text-align:right;
 }
 textarea.LineNumbered {
   font-size:12px;
   font-family:monospace,Courier New;
   padding:4px;
   wrap:off;
 }
 #RawTextViewer{
   z-index:0;
   position:fixed; top:0px; left:0px;
   width:100%; xxxheight:50px; xheight:0px;
   overflow:auto;
   color:#fff; background-color:rgba(128,128,256,0.2);
   font-size:12px;
   spellcheck:false;
 }
 #RawTextViewerClose{
   z-index:0;
   position:fixed; top:-100px; left:-100px;
   color:#fff; background-color:rgba(128,128,256,0.2);
   font-size:20px; font-family:Georgia;
   white-space:pre;
 }
 #xxxGShellPlane{
   z-index:0;
   position:fixed; top:0px; left:0px;
   width:100%; height:50px;
   overflow:auto;
   color:#fff; background-color:rgba(128,128,256,0.3);
   font-size:12px;
 }
 #xxxGTop{
   z-index:9;
   opacity:1.0;
   position:fixed; top:0px; left:0px;
   width:320px; height:20px;
   color:#fff; background-color:rgba(32,32,160,0.15);
   color:#fff; font-size:12px;
 }
 #xxxGPos{
   z-index:12;
   position:fixed; top:0px; left:0px;
   opacity:1.0;
   width:640px; height:30px;
   color:#fff; background-color:rgba(0,0,0,0.2);
   color:#fff; font-size:12px;
 }
 #GMenu{
   z-index:2000;
   position:fixed; top:250px; left:0px;
   opacity:1.0;
   width:100px; height:100px;
   color:#fff;
   color:#fff; background-color:rgba(0,0,0,0.0);
   color:#fff; font-size:16px; font-family:Georgia;
   background-repeat:no-repeat;
 }
 #xxxGStat{
   z-index:8;
   xopacity:0.0;
   position:fixed; top:20px; left:0px;
   xwidth:640px;
   width:100%; height:90px;
   color:#fff; background-color:rgba(0,0,128,0.04);
   font-size:20px; font-family:Georgia;
 }
 #GLog{
   z-index:10;
   position:fixed; top:50px; left:0px;
   opacity:1.0;
   width:640px; height:60px;
   color:#fff; background-color:rgba(0,0,128,0.10);
   font-size:12px;
 }
 #GshGrid {
   z-index:11;
   xopacity:0.0;
   position:fixed; top:0px; left:0px;
   width:320px; height:30px;
   color:#9f9; font-size:16px;
 }
 xbody {display:none;}
 .gsh-link{color:green;}
 #gsh {border-width:1;margin:0;padding:0;}
 #gsh {font-family:monospace,Courier New;color:#ddf;font-size:8px;}
 #gsh header{height:100px;}
 #xgsh header{height:100px;background-image:url(GShell-Logo00.png);}
 #GshMenu{font-size:14pt;color:#c44;}
 .GshMenu1{
    font-size:14pt;color:#2a2;padding:4px; text-align:right;
 }
 .GshMenu1:hover{
    font-size:14pt;color:#fff;font-wait:bold;background-color:#2a2;
 }
 #GshFooter{height:100px;background-size:80px;background-repeat:no-repeat;}
 #gsh note{color:#000;font-size:10pt;}
 #gsh h2{color:#24a;font-family:Georgia;font-size:18pt;}
 #gsh h3{color:#24a;font-family:Georgia;font-size:16pt;}
 #gsh details{color:#888;background-color:#fff;font-family:monospace;}
 #gsh summary{font-size:16pt;color:#fff;background-color:#8af;height:30px;}
 #gsh pre{font-size:11pt;color:#223;background-color:#faffff;}
 #gsh a{color:#24a;}
 #gsh a[name]{color:#24a;font-size:16pt;}
 #gsh .gsh-src{white-space:pre;font-family:monospace,Courier New;font-size:11pt;}
 #gsh .gsh-src{background-color:#faffff;color:#223;}
 #gsh-src-src{spellcheck:false}
 #SrcTextarea{white-space:pre;font-family:Courier New;font-size:10pt;}
 #SrcTextarea{background-color:#faffff;color:#223;}
 .gsh-code {white-space:pre;font-family:Courier New !important;}
 .gsh-code {color:#024;font-size:11pt; background-color:#fafaff;}
 .gsh-golang-data {display:none;}
 #gsh-WinId {color:#000;font-size:14pt;}

 .gsh-document {font-size:11pt;background-color:#fff;font-family:Georgia;}
 .gsh-document {color:#000;background-color:#fff !important;}
 .gsh-document > h2{color:#000;background-color:#fff !important;}
 .gsh-document details{color:#000;background-color:#fff;font-family:Georgia;}
 .gsh-document p{max-width:550pt;color:#000;background-color:#fff;font-family:Georgia;}
 .gsh-document address{width:500pt;color:#000;background-color:#fff;font-family:Georgia;}

 @media print {
  #gsh pre{font-size:11pt !important;}
 }
</style>

<!--
// Logo image should be drawn by JavaScript from a meta-font.
// CSS seems not follow line-splitted URL
-->
<script id="gsh-data">
//GSellLogo="QR-ITS-more.jp.png"
GSellLogo="data:image/png;base64,\
```

```
iVBORw0KGgoAAAANSUhEUgAAAQEAAAB/CAYAAADvs3f4AAAAAXNSR0IArs4c6QAAAHhlWElm\
TU0AKgAAAAgABAEaAAUAAAABAAAAPgEbAAUAAAABAAAARgEoAAMAAAABAAIAAIdpAAQAAAAB\
AAAATgAAAAAAAABIAAAAAQAAAEgAAAABAAOgAQADAAAAAQABAACgAgAEAAAAAQAAAQGgAwAE\
AAAAAQAAAH8AAAAAYx1BhgAAAAlwSFlzAAALEwAACxMBAJqcGAAAF3RJREFUeAHtnQuUFNWZ\
x++t7ukZ3iCggO/jY6Osb8WgMzAvn7uG4+bISTR7YnQXdQPCkGj2aNwlD2MSlRkeUaPnoCdu\
4iuJx7jriYZ50DOGmF2VqIBEiSggCoiMMA+mu+vu//ZMD9U1dau6a2aUbv91GKrq3vvdx6/q\
fnXvdx8tBA8SIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAES\
IAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIIFDl4A8dLP2\
2eXs9H9+ftSkSdHxsic2qqdE7YusS+1qaalKfnY5YsokMHwEPtdK4MQFz5UeExlbLYSaYUl5\
npDiLKXEZClFiRM53JSUaq9ScqcU6i+2kK3StuONy5reEGKJ7Qw7mOvKec2ToqOiZwoljhFS\
jbOVHCstMRb3USXEJ8hFu7DsdmFb2+xU4vWWFVXbBpMeZUlAE/hcKoGab66eKGOlNykh56PC\
HxH2VVBKoRKqh3qUeKi1YdaOfONJ56OkdI6w5BwomnOQlyPzi0N9DLmXpFK/60p2P/Piyovf\
N8mfM+/nJWNGnjw9KqOTToLVGSFt2p2Ril1gn3ij0Vk7YsoWVMzEuVPfPlRKYdfOak2LRSB0q\
zrWocCOG6qEhvgRaCj/dktj3g7dXXH4gKN6aRS0zpYzerqS6RAoZDQqfk79SKTRXHu/e+9FN\
L66as88pU/PN1pNlTLQJKSc73dPXSr20ur7iiwPcC8QhbNnCyhUIlryyOTQvYF5JfvqBL7jx\
+cNHjBj5gJRyDlJHy39o84D40H2Qtx8THaPeFuIOU+w2eQxEXPv2eQpyhtadhk5MyhNAo\
+cNHjBj5gJRyDlJHy39o84D40H2Qtx8THaPeFuIOU+w1C+KnyhK5FGEv0WGgAExB83eXMoLY\
rikbd9gHEP52VgQl4h89FUA6kJyYFbbQbnzLJg4zFiesnDHCwvUoeiVQOb/5C9FY9DlUueOH\
+zGhUh9nSqOqrm0uWgurkI9RpjBD4Y6uQcQdD5TUOW63zD3MHesy14V49isbdKyxbGHlCpFR\
UJ6toACF7F9VF58NBfDHT0MBaE74Ent+eWrrWr+Lz/QTw60AdB7QJUjps/OA7cOoBNBCeMUZ\
ttCu/coG28fLpvKElTPFV8juRasEahbHvxaR1guoeBPyfUDo4+OfeBdyb8L4tz9XeSXFAMOc\
bgGgov0g1zgGGw4jF392xnHhdc+Mwf3JTjfntZ2yC1YJBXNUt5KIKyck1sxXRdld6BmcevN\
aJovy/VBacMevqEP46/ZlnJjt9jx17VL53Zl5Mtvap1QGlNHw5pQDqXyNTQlZ2b8nGcMG2ZV\
qOoFjSdYvV0AZzDfayidv6FJ35CS4jXZk9hir7e27zm6p3T8hLJpkYicJpV1HtK/DJFU4Jw1\
lImhxM5IR9fzzgRKx4w/C+HQSPE+krbIyrN3qEPTNahsHaLDs2xh5Q5NCoPPVdEpgcqbm/8e\
7/zdOaHptag/mlKJ77U0/X0EcFJV/BLNg1f/BrXGjp5Wtx4dh3x1lfc7TnSt0yF/bUzAJ\
pd/CfU42AK2IUPlvTK1L/sJjyE5PVHqr728NzvfUzvvDODGy9GoopuuhmNLNfcTx48YHL2qH\
f/8hpXVu/43rQg9xtq6YtcvlXDC3fmWDQn9nbf2le7wKE1bOK65icBu0Eqhd3IaW82dwKPU\
hrauc6ZcWdkcjUZK8EUXMae71zUqwCu2nbi6eVn1Ji9/P7eW+ioMAogF+NI3iJLSf8dn9ipA\
WNW4rPy9jJxuPeDL/HXzNzgTsveslD2vvVvZX9foS4v/LfmqdEIpHDGlfM2uCW\
gJIy2wOENPaZ3fEcivd+ZYNCNJYtrNyhyGAo8jRoJTAUmRiqOCJnRW5FpbvrZ4F1qpwLJ6VK\
bVlWvbffLcRF04fEk0Kcxe5cxTYzNLbGH1g6M6v3FpsJ1YxKpalcwD7i+fzzsZDJbL4\
bVlWvbffLcRF04qazRD7176/rBjKylD5pBiZ5Wi4wQu7tikPBeCOpuW+Kj0sqP8GHNoAZuwL\
iOzuywDhQ9zBr2xoDRqVQFi5QxxH6OwVW46pvT+RxAJVLjW7vY9/+CeUBMkl68/rPQn\
mCufKzaldFN/yI8gA5iwC3dkIKhsyvZuCYSVG/KHFGkJ9tdYAwYt1n7Lv+A9H7C2X\
2qNzOvzCDYmfEtNy7QogXDXWIKAIQ7cOQZchyADWnerqN5xVXttcJsdGp2OtwmqWJU7A+Eh7\
yhYbUgm1IX7f7K1DwaRyUfN42FamIuxNDTdVEtamL6sYC9R26VtbZaW2px8Nfmehz3EM+mgsolk\
d3/ZnBGElXPGUWzXg1YCq5eW5/zBGy54aWOgwWKfnWbqptcevWT4FUBvov32gew8DLzDTMaj\
aupq7t/bMXX+yw/egJGKoTksy2d+gFBb9VoDvX5BlZTOR+Wfjyb0pP6U0XGOYNqR/quta3vB\
Fgeua6qv2d7vn8dFdV3rldBw34GSPg9i0DG9h5XWknh9kAaMmyJ6dklPzZmtD3cnu77vtw5C\
h/YrG1p7Wxp/VvuRDuc+wsq54ymm+8zzKOgyRSPRa4IKoGz1i8b6ytagcEPmb9v/m09cUATz\
Jow6tVnPcMxHzj+sNNpHsCJyja6csrRsMyrGkiwF4I5UiouliL1RW7fmNLeX3z2+/GfW1LU2\
Y572b6EAzkfYoPctJi15Q1nJyLdrFrUZp1/3pmkuG/yN9gAoGyMTf7neVIvx/6CHUghlluh/\
f9Uvo+gG7O3q7rzFL8xQ+zW+/8F6PW6fV7xSXhiNlayvWdz2X1ULm/4uL1mPwNoA5uGcdoL9\
ZFa6cgoxzhTG6Q4lNR5Doj9xuvlcy+rFbcujVsnLkKv0CefphUbICLRMvl+9KP4vngHg6Fc2\
NCqMSiCsnCkfxeD+mTflBwuxdmFbOZqT/l94225Y3TCrzpQWhthG2zHraJO/yb0kkdhpanZq\
GXwFf66/8Cb5AHcbzdpnhUjeG6YFow1gZeMmtqNCDekzTiXVuc3LK4yVTJepuq5tqSWFkXdA\
ufu9MfWiG3sqnNtcX76+3xEXQWWzVeqSpvrZmC2afYSVy46l+O4KvyVgicCugG2rp0yPTveJ\
o2Ulm2JWZEO+f6K0dFtNXfw2U9x7O/bqZct5z0Poi0+vdpyDJcdxrD34U9XCeHrloSktt3ug\
AcwtkOO9FZFn+gWtWdS6ODcFoDrAxneOCfRXWUSoK93pBZXN7vAe+gwr506/2O4LXgngLbrC\
76HgRdvetHz2WlMYVVqqm5zTTP5+7volRR/zJlOYlx+8ohOzEb+CV/0TU5ic3NGfjkSs30MZ\
tFUtil+Yi4yfAcwkjzqpZyb6HlgJebwpgLYxoO9/j8k//WW3xS32gQPHrV5aMTp1IDFN2Op6\
fz5ywF4HfmXD+/Buy4NVu73yEFbOK65icot+ZjP+8qf4JkYiTnGKTb/qST0zMKACq18jjPGL\
A4PCxYNpMKOtjREv84HpyOsws/BsqyT2RGZ6rzl0gA9sBhEp46hsP2ratmOJeGrugBWDB2Pw\
NYD1B4OSTMBmcmdS2E/GG2ZvrF7Uejsqyw/7A7guEH6Kyyl9q3fpQQvgXtx4dz+Ueg+Lmy5v\
bjjYtO+b5LSqpq5Nz6nwbFFhUdaYgemZy4ap1z5dlbByA3NQTC4F3RKYfOTkaUF9Xry0LwU8\
sDMC/H29oV0GTNV1C+iZhTu27rgAebkb4+8H3P553qOOyu/WHj21ZWbd7z2XLuv4fA1gmQSV\
2GML+6KmhorvaQWgne11yZ/glLX+IBNcn2FQ7F9Y5XQfN/qUa+Hr3UrAGg1MTLrG3bfPyEtp\
m6d5oyCZcJmzX9nQ2jAqgbBymXSL9VzQSgBfxUBjHpbXbzM+vKueRBRiotE/Bw8ogf/LIZhY\
/9Tcnsb68lt7DtgnQRE8lEvT2z9eWT5SjF7lFSZoVlyfTLvqUTOb62etccbRO1lHeS68SYeT\
2OzUdegWmRTW7S7ng7dKrVi9rLztoMPBK73nA4YrdZfM+5DZsymDymaHnClokvPOVHG5FrQS\
wCY6RwU9Dkx5MU9wQXMaX+ePguLw8/dvfg6U1LPvsPBpXspOniQwagElsm9gqNxctOEQlvj5\
7tBBBjAdHkMPdY0/q/irW1bf44t5cNKQKwAq7DsuJzHl6Clz8bk+1u2u78FXYWfklQ4/qY2x\
tYvjX8boyWN6zwc9/Ojwz7pUtv1Lp0NQ2UxLo8PKOdMu1uvooTDjLyxcrNWHEhjQWsyKrkPs\
2JHl4LpJicQXoyp6nMs5fYsKeile0G95+WXcEj3m5mcmjNe5b+lyHZYELxGjRmDnY/HtMK0S\
aPE7Md34PueUYz8DWDovSjzXVF/xsFe+Lpz/wjQQ9eiH94ZWqVS62+CUhV3lMtNjSHfXorHf\
wKgZg9FwIrTCRJwjWh5+/ocSLzQ1zG52BvItG+wOpqXRYeWcaRfrdbSgC5bD/PySxBHakPWO\
qZx9y4L10uABB4xk5we8qDsHO6++b0nwjzFXYaUViy6Ece0O1I7SAZkxOUgxtmZB9RcaVyxx\
2CbMBjAdTcruWWyKriwy4myTH9zt3R93/8Xlj0ESWetyy7qFIj1odwkAmhFEA2KD6DlwNe6h\
H52HuWwIaLQHQOUYZwr6yznTLs7rgu4OYBJq4JBWJCayRhTyeYx4X8/xCw+rus9L5yc50A+W\
8v0w0N2ZxAw7VADPZcEDpXpdsLXoDKefrwEM+yj47aEAa7yxzMjXm+61FzUL46ch7cOd6Q/m\
Wncf9BTvXbs6Z3hNxPIvm1kJhJUbTFkKRbaglQCWiwbuiiPtyKlhHwZaq8YKoeMcji9Iy9Ly\
Pwk79U/55Bk75fSXMchwhj79Y35xY7qu8YspvTbqSG+55hdjjn6YS6ErfyqVOL2xoeLrbmWj\
YwkqG5S2p1IOK5djzgs+2LB1B4Z6/gG+uosa6yuWOYljzcCuoG4llqxVQOYep1wu1xUL4pPR\
zD3GL6wlVE4jA35xePk1NlSuBb/34RcwB6JXGgz6rflBBjBbJH7tlWbGDRVdb4bieXgpPbhN\
NQT3iqMHz7ETHvuRxnv45r8FpfQWRnDiqVfV2qBlxEFl6+rqDLV82CTnVYBidBs2JfBpwMJP\
aW3rXYbqm9qXMLnmChjCnvUN5fKMRc2LbzJBk8mU55cn4x/2rLdJQzNjtKkyuuO1pdqccfMz\
gKGp/aHfXooVi+JTofimZuJyn8F7QHmhAMxdAaUeTX6c7F07sUUkgyq5Oz33vV/Z0C7b+scH\
LtnpltH3YeW84ipGt4JWAnu7Pn5xwqjxB4IMabBc3Q8rfLzPCJfTc0SF0b8NaDzSFWqYfhBU\
nm1djITHGhN3eSRt+42Mk5KWcTsxFMe35RJTvorP3rmn49VMOgfP8oiD19lX6IdvbXmkqjvb\
NfydX9m8WimZlMLKZeSL/VzQSkDPzcdYcyte7lq/B4XKfKQaNeK3mL47r29fQL/gaT+/vrEO\
gDTTX0U9UWbKUVMfh9MYuLZjVPzxxu0fPO0/pTedhOd/1XXxGZawfuXp6eGIlz+eme2X9lbo\
0xuUl19F0bLaKGgQhafa5NVPhxjK7X0gLuOMRm+JAFefsnnaKzLRhZXLyBf5ediUwKc1/wD7\
fD+JL72vEtDPEIqgWkZj6zFP/d5duzt+ZHihxfkLnhs7umT0l1AjKkyVScenpJ1WAlAACzAE\
dqV2Sx/S+nLN0dPelXVtD/SkUr+JL5/9VsbL75z+bYNS8Q2EuQN/Oa3x1/FJZS/VZ30EGcBg\
ePdtCYCR0RCKr3q6vL0pOf7XfXvDAaVzcGjQECZX56CyYcmxZ/7CyuWar2IIN2xK4NOC075/\
4yMTRk3XuwyfGJgmxt/xdbpt8uSRi7Fl1luoFJtQm3Ul7cKXfyqMVsfDvwpVq9RPAeh07FRv\
hUL4693pwu1YyN+FX0C+Cy0VrIWXzylh/w3n7fiibreUtTsVURMitjpKWRYmPKkZmHDzFciM\
dMflf6+eWl0/65lMmCDD2YFEl2dFycgj38aRAbQSPGX1sCGUcCaKrDOUyszauvgcZx6zAvTf\
LLGqFlXPjFjyIthCkphR+cN+r76LoLJ1d3d45i+snDv9Yr4veCWg9+SrXtx6G/arezLXB4WX\
tgzv7Wk4n+Z8f/FFzzUKIa3ky5ULmo9CE8N3HgLinI5IsRNy32hsXxoRnTBmBvWmiP9zT7o3\
j0q8vnN35zecGfY1gCm1w2/fviCjoJXytieolL0xvRGhMyNZ1/IJtL6Ww3j5y8j+7i1dyU57\
xLjDJmM+xOFQgtrucgEUTDVIpFcnovWAf2KAEvArG5T3tjBGQT+5rCIU+U1BzxPIPJumpRVP\
4YEuz9wP9xlfvw/0ppuyxDp9uNPyih9l/XNXovNSd5dGG8C8wms31CzfrkCQUTCZSHj+wm8q\
JV7XE3xM6WqjLSr6LVB668ToEXtHjJ/4Cdw24+uzFvsJrsT11RkFoOOALtznFZdf2SDl2QrQ\
8YSV88pDsboVhRLQD6exvrEOj9y4g9DQPkC5Zmjjyz021LdV7yb3zfL8qmsDmOFARTVWFC3i\
N1NQGwX1jEavqOMrZ78D2ZVefmHcdPfCU86nbFBB5rKFlfPMRHE6Fo0S0AtoVm/d8VV8km7D\
C58YrseFuLvspLpbx79z64erdZNyuNLKileJdalUak7j0orr315x+YA9CbQBDF/ck7JkHDdB\
E5sg69OKMH9pdRJd6v3vgEvYbdQcucSlVM9nO/QaPP3KZlve8zWCmJjk3OkX+30RKQE8KiwN\
blxafhe29JqBL8of8GKam6n5P9mdGP5bmUikpmc22tR7BHSKjjP0kmCktCf/KAMlsOJXtejK\
v7q+/OzmZbN/Z5IoHT3+NPgZn2eyx7uiZOJDM9xoyTcZBTOya+vndqW3URPijYxbmDOe1/au\
zq4BrYqgslmphGdLIKxcmLwXskzBGwa94OstveB+sf714IiK3oiO5mXod+r9/I2VxB0P9Ec3\
xp7XQYu8JGTqmcatO+NeY/99v3xbh+21bh03cnotljdfCZnzkeapSDN/vjDg4XP4Cnb8+W9p\
9zzduKz2Q3fevO5lytqtomo30pzk9Ec5sHOY+FXfVl5Or6xr5HkDFMGAadKQ3yAO9DydFdjj\
ppf5kjNq6qrnYi3DfyKI5h14oOKj1aZehBJ9NtWTfBAGvv1uIawS2xVTahfsB5OdfrpseEaP\
mRiFlXOm8Xm4xnP/fBy6aVg2fty5SkWno2mMPfSF3sgCf3o4UGGSj/wI548wVLfbVvab7Z0b\
Xx/MrwGlf9ZrXPQMbMx5CiAfjiHIyXjhsR7BKkMfG8mLT+D3CdJF2qod1vNN3V3d60xW7hyf\
koSVf0pEpkZFeqJWQtld70c6dnp1H7zi0z933hOLHWYJu1REhZ7ptxeVe69XWH+3Jdasm6tO\
iEWsY1G5j8Eaj2NR0adga7IeVOR2LBSCcVC8Z0u5Ue1JbspxVqHEcusjRKkYLW0VSSUinTmW\
LaycfxHpSwIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIk\
QAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAIkQAJ5Evh/ikTb\
m38w0ncAAAAASUVORK5CYII=";

GShellInsideIcon="data:image/png;base64,"+
"iVBORw0KGgoAAAANSUhEUgAAAFQAAAA4CAYAAABjXd/gAAAAAXNSR0IArs4c6QAAAHhlWElm"+
"TU0AKgAAAAgABAEaAAUAAAABAAAAPgEbAAUAAAABAAAARgEoAAMAAAABAAIAAIdpAAQAAAAB"+
"AAAATgAAAAAAAABIAAAAAQAAAEgAAAABAAOgAQADAAAAAQABAACgAgAEAAAAAQAAAFSgAwAE"+
"AAAAAQAAADgAAAAA2CVjOwAAAAlwSFlzAAALEwAACxMBAJqcGAAAD8xJREFUeAHtWwt0VMUZ"+
"npm7m5BIEKgpIILvF0gV0QokG5LsJvjAx2kF61vU4qtINkFAqRar9QXJJuZoMfVVpSri8VEV"+
"hGSTkN0Qj4D0SIMo1AeEKKIkIYQ8du9Mv7nZuWyWXZLABg1z7k7d/7555+Zb/5/Hv/MEnL0"+
"iSoCNKrSDkBY6rwyC22wnqkRehol/DSBkFAyhBBKRKARLJEQMRDwGoq2BV5ayJ/A2ge9HQUgt"+
"oaxWCFHDhNjo53TDcbW1m5sajJL5kP5HHJAM2dW/prrwgGQxqGhF+AdhXeroGQT5WSzIGI"+
"JWwbFfqPnNEdOvXXv9GnWVmu8z5dYcMnwak9+fh4q/AdY/Fb4plGEoVOjoe84wVhw9ABZ4Nl"+
"BN6hhNDP0UAvIdzLuPAuz0/5TubvyeeQAJqZ4xnDBZtMqJhIBDkRGl0OKfEIT/prwgGQT5WSzIGIz"+
"oaxWCFHDhNjo53TDcbW1m5csmaJL5kP5HHJAM2dW/prrwgGQxqGhF+AdhXeroGQT5WSzIGIz"+
"JWwbFfqPnNEdOvXv9GnWVmu8z5dYvcMnwak9+fh4q/AdY/Fb4plGEoVOjoe84wVhw9ABZ4Nl"+
"BN6hhNDP0UAvIdzLuPAuz0/5TubvyeeQAJqZ4xnDBZtMqJhIBDkRGlVOKfEITtborb515c+m"+
"7Y52I6H5fbT6mNGUimSAnYxyk9BpNfh+T+j8vdL8lE+jXaaU12OApudUnghzvJVyeq1RcUYW"+
"Q5OWDty27ZOfwxRlHTKyKpKEvqq4EuvFcIiiiGEvsAFfbnUNX6bUcco/EQd0AxnxQSY3r2o2wR0"+
"16uMikUrcm1ro1DXqIpIz6o4nzF2O04Reg9dLOX+q+qjQDqjhyvNKCAEKAqi6dl42nZ6iPawAV2nQhdYU8H+XSv8uoE6SmfGFUz"+
"umLBxKaDrWBP57982pr45r6t12PsnY2ytmA4esSdl1x2oOUeNKDp91WeyvyiANo4AuPVX4pz"+
"ba8eaGV+znyTJ7+p1Q8beq0Q5CFCebWFW5wf5Y/7prt1OmBAZZ7zFgUW8X1X0xr8MhE4Ees"+
"oguMWbi7lfgl8U+eVx1T11iXDWvLhqF+cmP9adG+hG0QcXSSVbYgDwtsXvLVLGYuKyaylCT"+
"mwr+h2JXysrO2tItQDOyQ7P4i4hyBHwTrvdwtm+UqFhcyQVfFGlGPzc9HY25HQEQHNYT6gyU3QXvI+9CFIZVCZ81HHwKohU+Q"+
"9gbe1Fkfn2D1+1+EssT4LdoN5U+NrelKvsmE4hvZnH4fTeh+Es"+
"mwr+h2JXysrO2tItQDOyPZPRY4VEiBtKXLaSzoQfLumOmZ5UwukbUKLpUKIl+2tXlwHPyPFg"+
"wKa5RLCJx6x7cn/NZRV6WMUVX5JaVRqVlhc/vs+qujxkwmT7aWfNC2d+d/xi8SMHWmZTD"+
"02Z4z9Q0Uga/wdSS3PHLw7W3Ww7Hn7xnWtdorURUjqGBewell7vykBCjiSaPafyEB7pOUsoKkr8I"+
"bTsLJQTHx0xbYwWYi7F8mHUUzHZk3LLq+GTeBCa+i+JJTzBe8tsSSgiOD0hoySGcfFucl/x8"+
"MH0/33T8ncsTV/3U8BBPhuvMNr8pzpRzMPbk3v/U2JcK4hoxfsHI5M+RdaMQqsjQ9UbhyqPbb6Xrvc4"+
"afpvgzOEftunV4wgVno3FeQ8OBtGoYL9HHHxunB6t2JWV6aVbwXHqvsvRqEt9lGVAhA2vTJEm"+
"3EkQ51O/Y7Tna07pRzKOvN+W5NnmcUGWMSIewGRwLoGpANWn7Zt/58UpiIfl"+
"OqI5V/Omr/rHnOVnNnmZUGWMSIewGRwLoGpANWn7Zt/58p6UA/uFhGmCH3N1Iu"+
"Oq15V/Omr/rHnRWWHlqM3En5mb76kukVC5cVpuxQ6RFNHk6N6WBaFGnRPnLymzHo5Yeola1D"+
"M++B5sThvdfv8w1DOAnbUKyw6FT5Ih5kGvOY3el9h2raG0gbi4YshFqOwxh9Bcrzo0LT0bxb"+
"4Bc16lbmSn4THfWhqjCEflq9ZEqbih9o6C6YsBjlLDXzU7J6bdEdvkh0ky/wEdiWLvJZ2azg"+
"tLAaOs65Kg5O2du4hgZHeAYPG/IBgMwwkoXYojf5J5YXpf0YYK+x3mMaIDQBNQ/evUWLS"+
"sxySyfdZ3jEcrfLZlbIPqPkU6LFfgO5fbBEW93OY/DBStB84xGrAh8HH1ByoSmEk72epkh0"+
"kznwIUQh2leFsfQBte0Oq6HxVLdDc/5bOj8prGmlZ3luNMGEbE7E5CAwjdLcz9i+hDa9KyM6"+
"0U1AGRMBMJEgxCaDOfDjLnBsB22xjPpEoBPgM0DU7FhofnRWGpALG7ooULSsS3tHRaKbjs/"+
"MIx9hdgX/fs2T1TUsBrKCMvA+FKsmILDcc6PBqIXchUNlaoqdaV8ouIdQlQS5m4vd6UFd4"+
"Q/EA8Btg/mvcruRXQAMr9JDSUpwN2T2B4wrH4EFyjOur8uicfO9wel4Epw0e+EGEi3f20O/u"+
"rHJNaVY89ntKfkViY6nKGAhynpBnpQngVHxru"+
"n9N00D+QaWE1FDo3lnPi2SczCHEi/go0EIdn7Q8mj5fVd2jY3LJnERH6paAbYBnpQngVHxrU"+
"H+b/MmbyTQD24fR73CdiInoFw8AwxcMtWpL6lqFG2VJMZCkYAXCoh8mL0ZvixODrFY99hmcS"+
"jYnFWRKdhfRqwfWrkBYD/gtQ1vOp95adY/BSNl7lAf2LKtfFO/dLV8whIdPpJ5BtDh1hAYXm"+
"nCAY3RKStz3K2PnBdD/xlwbHg79XLZz4gzs/tYO4JooFPQCO2B/Oh4aeiUQ9JINKd3j8hDaT2"+
"Bx1mAgpF+pL49VQ4gHE6ivHZfOhQ+ZmaVXYSEH9NdjjU5pEbU5pflk5zFlXrBrVjDZ3kCuE"+
"OX5Goqv8oaGfUrmcOl3Rw5o8EgfFtjV8r5iCQxQ4RrVXmkqNv3mrSrc7Pc8AiikqHhxiWTUX"+
"Y06RuyB1tS2rYnQsoX+D2k5WZid5AWockCxMc1asLwt4dkDbC6gQLnfhhHWpqTh6FvQEE3ZK"+
"PpP5LczyGIIE+Q2bKJfDQHp2xR8hWY7DIIlid0GKwYty9mqoIO3jJ3gi0WX+cA9vbW1gfazt"+
"ZYIhrIaC3tqsx1nDCYApDVR0NHbX5sJLW1W8pWHnHDR0BniOUy/S1gHMi3hb23K4wj5Mvbus"+
"rxwfAe6t9U31g+AKvBU8tUqGDNH6q2XomOEdjibuNX9dN4YLep52EVreT/IAJD9rajGsBJZl"+
"goR8o9DBazEfFIFpsxB8bkv9zt/LPKnTy9AZFLLbH134DA2NRFd84cI4rdkHeqxKiwRoI4uN"+
"PVYxdQzFuqD4sXLcU/HKF69s9Pt9FSpuhDqdL2dDFhObBBAupbFWOaYaz9qiy/eUumwvtfr4"+
"eTBLs2OgucZCWTAeBJCoK386rVpmZIyYsyo06uOSoowGqbUAzgQJUHsgZ55cF5e4kk93u1Ie"+
"k/WT+anFaspFh/yodkGR6DJPpKc15tjBSDOtORKgmyxCnBlOCCar9zvQrda04DjTLDYzjrFL"+
"9/mMMRQVx1pSap94NPWWsj4mDz5ieBs2RLjmgEcOI1hmLZXfAMs0dyKoNEtgBLpgGHbUIz6W"+
"X4Ft67eKisyys94vL0yrkTRo6932bE+13JCgDqNNPkHWqO9IdJUeLqRcDEetjDJkevgxVJDV"+
"XDMAMEwpWFBpvu11LFsmwWSuk3QcxeZgTFwuzRjhEMy6dyp+tL567yUGagCKfKdbBljKsGW9"+
"zV2YskHOulTD/hnjp8yH6zizscwKNJIlK1kIDXM34lQ0GHAjgg1AZnqWdzuh+mfIXASSHEeR"+
"QKcCxEYMPVJ70MnCjgsON8td1pBsjzlJodNG4Ej5FkoZ+nTv5BVMx1Lrn4bMMD+wFhvn1OhU"+
"mRxWQwUVpUDdNM1QOXVNG26BOjnBI5ca58RSuhXOgi2xjNWgIliEY2nE+b2tLSR1b14xCuvL"+
"d6CAqwDqWGxZq7FU2mOxWNfLzgH4q5F+SYkrZb7MIzUJ9Whf4iAuiN8Y52Qa52IBgo3yG8CO"+
"wjb5KmhKnf6pbz40XALahvEd8x2dAZAfhuwEznmm3FbKHG0t/DnwfSO/wTWcMZoHv8WgSHSD"+
"L8IPbr9kwLhKVDLav+8j3VID+rbUoPvHBXYD+zKBMmbac9Z+caefAkU/CR0s/D6yqWL79i2d"+
"eZqwxRyEgewU3GEa6GfsJ7Lbtzl0pxW2wBCinER21rfVffZqx/P/MdPej0+I73sG4yy+oWXX"+
"f+RYHZLViErHjg/7paY9GzfLfbziiURX6So0jtB18XGTYMOrXOPNjYVK7xDCkbrAke11dSAe"+
"jXRAwO6szHdkVz4RTAxr8pIhpk1/EsENF2dVnSTjR5+OCGTOXHky/ArX425UYXBKRECljw9j"+
"z9N+5j+qpcGIBb65rsHTRHJDL5pFBFTmq9/dB+pMhzlyPNPCyDxiSRk53tsB5ol1jX1MJ5EC"+
"I+ykpBJlGDjpq4BDYpJxnhKceAR+27Mqx2EC/gAnwLZwJ8D71VCJl3GyR9lNcMC/J8E9AjE0"+
"m5yZs+osgCmvIE0NB6Zk7BRQySTPoDEAz8ZJ3wp54C9pR9rjyKo6mwteAhzm4NDy35Ha3yVA"+
"ZWZ5TRGeppm4PeE2rqZEkngY0jOzKtMJ08uwu5jV2XXNTsfQUHyMG8qUvQHheXA6GLuaUJ7D"+
"KY61+Ey0ZyYui10T9ctiCih51c/i1xdBY9t0waaGLh0UX28ODdehRv6BNhyDQ7HrSgqSt3Sl"+
"Pd3W0GChWD48gD1xFpYQeQMSBuQtmTeyLTi9N37LC7f1DTunY1UzB+vwvIFba5/qzp8sDgpQ"+
"CZjcSeHAPx/eHVwJJ3/tv3Xb692pwC8J9MANw0ex9t7ANZIV6dR3f3U+aECVcBy0pUHYg4jD"+
"yUufbBJ0UacOA5X5Zwzl/5ksu2JuxJyAi7XiB4yVc7syVkaqctQAVQXI/wJxjc3GsYd0Di+G"+
"2+z5nvqTlSrzQMKMrMrzsGicCpfh9bijsgqXC+bD7xn2pLc78qMOqCrcuIdP9KnwN94lex8r"+
"gtdU2s8VYsYei/pcBrfkFNTBgnP9RdRPXujqhNOVevcYoKpwOJ5vgiv8Khz9/k7RDkUorxMd"+
"Q/m58uAOuzxpLRPwbgOgyxB/q9iVbB59RLM+PQ7oZXM8A1rb6Ddxu/sMeb/ogrCO3nANks5b"+
"DXfHqLA0NOq0sa0pxn9K3VfGudOOxMS4OE2La+2j9WO6SIRvOxFnGsPgoT8Z1nAKVh3S038C"+
"xvKNOCGogpBKnNevLH4quTZcWdGk9TigsrL4l10xjiieLXbZ3ums8u1ecPmPPCE1ahdeeVws"+
"z73l+Zfa2UnvuHwbAeAOaN0OmHENLmd8DdpXuE21QU/Qvyyft+99U6T36BP+kC7aRQr6NqdE"+
"mnxEQKUmt7WRPwtd3ITll2s3Zzf3hlVCKFSHBFCN6+/qjD0qz6rUtT9VEeOfa7vq7m5tI/dD"+
"295mGh25YkHSDyq9t4XKhHq03sYf/3Eha+AxrWnBBeHc6uq6XXUbAKSDcC0VK4G7ejOYsm2H"+
"REMNEKF9WDhLs18hly/wA+RiDRjHOJ22Ij+p1OA5DH4OyaQkcQocuVZiBl4JjRwHH8CD+7tA"+
"0FuxPWSASoCgmV4sZZZhW5rXGyecX1wny7+E/+IqFeUK/R89knneGSwJ7wAAAABJRU5ErkJg"+
"gg==";

GShellFavicon="data:image/png;base64,\
iVBORw0KGgoAAAANSUhEUgAAAKwAAAB/CAYAAABymylZAAAAAXNSR0IArs4c6QAAAHhlWElm\
TU0AKgAAAAgABAEaAAUAAAABAAAAPgEbAAUAAAABAAAARgEoAAMAAAABAAIAAIdpAAQAAAAB\
AAAATgAAAAAAAABIAAAAAQAAAEgAAAABAAOgAQADAAAAAQABAACgAgAEAAAAAQAAAKygAwAE\
AAAAAQAAAH8AAAAACt6tZwAAAAlwSFlzAAALEwAACxMBAJqcGAAADQRJREFUeAHtnQ9wFNUd\
x9/b21z+iYCKCiIK1amW1j/jH6BCkstFEFth1IGpRWdstQoqkEunttrW2nFqO1YTIlatinZ0\
amdAqY6jIyOXi7kgg1arVv74b3BAQPkbVAjJ3e3r94WcJpe93csmcbjb784kd/ve7723+3FZ\
ffv+nxA8SIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAE\
SIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESIAESMCGgLRx84/Tylk/EYasHcANHxRK\
XiEuf65tAHEwaD8ImP2wLTxTadyBmzrT+42pzRSrd3peQvpXsMtrhgNYCn8fewHXFUap+zyH\
ZUASIAESIAESOKII+LPR9dzsk5ELvxdKBTzlhpQpkbIeFle+8Jan8AzkmYA/67BKXiek+pmQ\
hsf7VweE6PyDZ+oM6JmAxwzznN6REVCqS4SQXwihLI8XXtkXAHUkYB4/Na7gpKVVPbnKnF0dsS8i4\
Nbqem3098iHoOS8sa5O44oUmz+EZcEAE/FWHX/nKwwCjsPXgwgnAa/sY9m+gPQOQ2qA/y8iGwiHw\
hsf7VweE6PyDZ+oM6JmAxwzznN6REVCqS4SQXwihLI8XtFFcuWqHx7AMRgIkQAIkQAJHIAH/\
Nbqem3098iHoOS8sa5O44oUmz+EZcEAE/FWHXfnjs0FLd/YP80ZNJkRALvAWlqEGg4C/BGsE\
rgK0AMb/d3uCJ1WJsIyVnsIy0KAQ8FeVYNmsYWJYR5F3cqmUmLt6v/fwDDl1ZeDt8YeBXVgDj\
65pV5ccdZ46QncHyziL1ZeDt1K7m5lAyw2ywTmXNDa8cLpLjlWWOiolAzJtKAM9ikeUH4rpzPEo\
2D8HBZtDXp+28KXicYGjq9Fve6Eh5QVF9CrE/2qnEo/BrdwGcX2H6eUIxsqGuXiXdKZfVBmz9\
fawMFi9QQt4CcZ7gYOroBVF9CrE/2qnEo/ElFa4DDqHalosCUt4rpJzsGLGNJ56Z10QqVRtt\
rHrDxjvvnShYmyysWPTq6UEzEEGJeS2EWmpj4skJ8SVQAt/zSeLLuz5aemlHZiRVkdhpAVH0\
F6R5ZaZff851OgmVOrtlSeXG/oTLB9s/GhU8s/lSnKwhwDXuiMkeHbh86ixYwUrIZRmYWSMQ\
zeRQBRdVySIJ0IRJmcHvYrLGv1/wuMu5v7fHXmonlqNFnkzuT\
zKJx8KU4BpKtm4s9z9IbdPWZyG7V+Xmph4yU+gBRnEJtTHrDs+8O/R11z7khjOKq+7t9hNDo\
lyXYy/PDWlOy4aUL2KQSUpZTV3MsJl7VFZSu5uZ5SyXXkOQnw2JN3fRLf5WMfZSEPKpEMy\
ZSfqQnQWU/kGQRFYkoUEYu3bHJBQTSMTDcM9jmAI/3BqEaL5JWgz4+lKeKLzzpNqYzPEkavK2OL\
FBaskF0LjtlcFUDC6dbHAM35PAVYRbKpbKNbU41A/mBVqqAGnpd28SGEeXmNiDwWaavUFeoRm\
VOwvJLZvbP1tKs8mTsaGvC2ZUe5xSYA7YBdEHJl17GCxOBeaTYh8WHU3FGSaLBSr5JPPhIIqwvdu\
Ga3kGJMpxjVVz3HP6MUBpVp+8HbLZpaH1TeS4J4dYQd9RDDEv8lWaP2mXBzvRW1lQsqfqWhVJe0\
sNbfGbdhDazKGTd8QSK8e24YyowvjmHzxH66OQy3ugmGl2P9ky9C5HRCkgLwOhYs0MjeT7IlSyTdS9\
YeLMh0rWw4pfHEGUcDpWYkgPqNAN7Mb7qxRUxk/d4DwPA8GpBCUfgTgTPZfjPIcC6VhzIdhnyFRnOqP3HfDx\
zMGGFeJYlpOwfkxwBUAXN7QmXthDu+Zjx6fgqjx+AgAdzEqzSpRkvuzI+0kUdxrsCwp9SbezkbgNqT\
M5IYTtDHV1MWhAdvaMcCv0kOM6Kc3kYS5IGfsP+mOPuxtKqNkP0ZVnWVTVBCBXBrfJM2Ae\
q7TkmyKtLQY4tdgjhHcDszEOKnRP5QJcyB3SdsgfOyUkO3NwObdeUb1eqm47mHjbeiNmy5+sKfxKt0iWWEP\
13o4HiKDXlZ0bXFe4MPhz5e5etVz4nYoI5HIW7DRHw3Jn/5bePH5BnqVXzvebkGzPBpMoh6CSc\
mDA7ZObG1PT4brdbQwpDQZgCHyGt7TKNJBpgsgDSB7nyXMhaESGBKAIm1qn3VaA9e3MHYWAYuMy+\
lhYNXbjH0XSb72xE25oM6yzpH2n3H0aPHSmHl4O3mcxV6cR5qW2+z9yHeTX5HRxPW+LT6yP7D6EvbZ0\
dS5QRN1l9ri2ujWvqNNQu5upAfwl7OPVO7LyMTK0XcuX4cIcgX7xF1eb2IfyhCm3oUYlAj4KdhiR5G\
5YQzLNPgPVFTh3Rh7NXSZaQNrY3l2ndsnmD07RJPk4HCyxPE7ZfeAnWyyhIkSVQhbqK1GJJM3JXRAnDd\
BYU+s1U8ld/3Y6vKWkjh6Jtw/qLc9Hd/9ojk5fUMrXUbjLlVtG0wdOpRu+9zudbBNcP2yNEQ40sT+RXa0D5o\
fxvwLctlj+0Pl7vMgm6OrduI3vQt3zc1tFHKYPwh7ZgtSgPJ2WgqI/wcT+1bZ1K2BHLe5tNAUmtT3+nj8\
EKy0rpWMs9dWXZDZbHzu5BTYAqWGMhe1CPWY6Y+Y0mvsbAP/3DyGYVYjpV3PThNs4WDgL/8uUy\
PdxwdYe0DaNR8ovW+XGVH/f8RsdUUXKjHgjc+XbVAnvTrUt9/G9k4WxYG8wmjOehOkjCU+aPs3ofk\
DUyzclsuyTGfpEz7DUSUXUxBXfe1GTKnRdL6G9mT51mGJg7dGbjZDTnN8T3XbjkXfXgAPaQ8TVqQa6\
/Wn1O33PCnEk5bOMWZGbyaIzEq9UWG5hqkCEHHfhIf9UjRyOk7M5nkSv71n0b0bkbS2QKtPPfwFpY6\
nEi/D0H4IM5Nm9nRK2F+FGKTvqMd7u5uL4vbOZ1sXHcsNjM3rQLbAN/5gBALBBa6lM5ceO1ex/MP9c\
aUgPpDNXO5KwQL9P8Su0u+SfnoO8tK4xcfcoS2g/P6G4+8rzAACQXiD7SsWLIXS/iB6Qf5cdCw8n\
S3cfJAVRc9JN89fLbBfpK9KcmPGsTMqSdb6Ib3Ge7W9dt3+kc0yRhFNG7JGxkEQIxZZtdiCOb\
GLp2U2BiwD0LBN6Q4L36BAddPgdpwsCwKJ7vkRMOpJDJRodbOVAM2uCjHC5mjhZqASoXvFBNpU0t\
nSzmPKRnDqXz8zjwBGzxYlNTHJd3gbSbCLFlc3ExkZSy+jR7LZBRsKUxkyY7FnMyzg7lpW2Ayhy\
JvTd2xTA2ZbA2vsOLQ8DyZYTDZ7w6n4fIfPH6ksiwfFYzB0qtHKPYQTdTMZCwpvFcR12bTsdzoYL\
eBADHBZv6H1JDBHzqLqmLc35YYgxGNmWQmhbsg2aoWEvRDMNWF1LgaVkpzDUy2vWXcbJWexM7Vw3\
a9dHQ+XRUcI7R3vEmJvsPcEKfkGyUFiK8BnhpOjSGTYfZAoiaAQWt8aeDG3Q4EF2Wy0uxPaKJEFNrhh\
cLwIs1vlY9B3kW3P5C3WnYdfGXPwA9HhbPSx03NLgKD91q+Cw3YlCCwgFFfLWPbGwJ3lERYMfVS\
EIhCGQ4KpXPYvKdf82C6rlYuxR6ea+z8Z90xRd8yqX5vjJB3dlFU2/9ryP3aHX5jYBKnzAPhZrUDtV\
Gj66GH0GYX5UM31KLJx1G8j6VdfNmXc5e0dVm0fjRRrGrX0onkEPVgRGrEHGfTzBdUHJtRHBw3gBPYF\
a2QRzuFIeyDJhMGfPF5vhTBJWvFIUPQ3aTL5TbgbDJUIPGtIL6ZNcPmp7OxIBN3X6frfOE0gD3UeQs\
0RtqkEaXtQjAwfc8ox6PSzD6J6QwbkJrr57IaDRNrxT53HZrGbmrCaNMqaFjKAQ2rfF5mkOq2iqS2+D\
/VO3nMBLSWqdaETNhmIE9kMvL5bX4Xn/BzL8JNXG9Y3Ug+qmWIyM3iMPH3k0Z5zfb8BRD5UgW0iDWl0\
3tk2Dr0aLB2Gqs2LtnT2BLJvPNJHvUBGH4GQe2tL8RQTYtn3s5ZsxuXaxHVmyR7AzEVoCq6GmjaAo3h\
Yb9N3l8WYP9uvbL5gS/h3J1tpt4tz/Sqc4m6qC62V1GOGt5GiiR7yZWB3lmmR3RaEuAAW0sLZyVt\
ioDtaq3/9IXQcWS3TD+N6LO4NbbjaEhE6JEqd+c9pbRSSv9aGe1BZxqKwcAxWMGbBsr31LWl8JbKFF\
a3CQO1AiCBEHYWHbudB+BOERROMm1uVxBsJ2zkr9UDbU1OCU0PoWoDVz6TaQX3bmkmCW6b7KzzJ2\
4dkMjDqo1MOU/JIxBMEK4Bu2KhnXIJC2wnd93FHsiNq42FiJb1U1jCD2I+3pqAgoHbRFf8QfJ9BAbw\
CikEOBqpCwQRBzAaM+5G+Bk4nIcnShz5ZSH2DuN0LUfsMOIQHhEGXiXUpd2v0hiEYrYF8bGPTqo\
XMbkzJS07f3fZDNP2y/y1NEUkhJ3UPNqpfDrWM6z/lUFPtY0z0H0djdbP3V3fnxMJAAi/G6Bx6r\
EiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiAB\
EiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiAB\
EiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiABEiAB\
EiABEiABEiABEiABEiABEiABEhD/B9wOq7SGUV++AAAAAElFTkSuQmCC";

ITSmoreQR="data:image/png;base64,\
iVBORw0KGgoAAAANSUhEUgAAAG8AAABvAQMAAADYCwwjAAAABlBMVEX///9BaeFHqDaJAAAB\
HklEQVQ4jdXTsa2EMAwGYCMX7sICkVgjXVaCBe7CArASXdaIlAWgS4HwM5zEVS+mvSgS+ZBQ\
8gcb4BdHyzwv8szMSaUBHNm+KAd4QC8LDpDn8ogT4UpPGci2jI8IGFx3eLwPWaHknVyWecev\
UEbDXaB0X2aNjueYDOzNklQassPCkjc4nW3E1SfwqYk6jU/vAkPhg0AlSFhve8Jt0dkwDMwr\
yMGSSuPyWHAr19k0tkV2sb3sdW2rUCqW88g4Rp1A9s1JPv9cTp1NRD4XFkin8XaQCIwT6Lzq\
ZO8dHw/4+U2GzqlS8gbqVmkfr1N6YXK8OqlD0OmlGTMvzPERA8AL9vvbOifpSoL33fsVytrL\
S9wiqDzznhUI38v5n783/gBuUs2eLg1c8gAAAABJRU5ErkJggg==";

</script>

<div id="GJFactory_1" class="xxxGJFactory" style=""></div>
<!--
https://developer.mozilla.org/en-US/docs/Web/CSS/line-height
-->
<style>
  .GJFactory{
    resize:both; overflow:scroll;
    position:static;
    border:1.2px dashed #282; xborder-radius:2px;
    margin:0px; padding:10px !important;
    width:340px; height:340px;
    flex-wrap: wrap;
    color:#fff; background-color:rgba(0,0,0,0.0);
    line-height:0.0;
    xxxcolor:#22a !important;
    text-shadow:2px 2px #ddf;
  }
  .GJFactory h1,h2,h3,h4 {
    xxxcolor:#22a !important;
  }
  xxxinput {
    border:1px dashed #0f0; border-radius:0px;
  }
  .GJWin:hover{
    color:#df8 !important;
    background-color:rgba(32,32,160,0.8) !important;
    line-height:0.0;
  }
  .GJWin:active{
    color:#df8 !important;
    background-color:rgba(224,32,32,0.8) !important;
    line-height:0.0;
  }
  .GJWin:focus{
    color:#df8 !important;
    background-color:rgba(32,32,32,1.0) !important;
    line-height:0.0;
  }
  .GJWin{
    z-index:10000;
    display:inline;
    position:relative;
    flex-wrap: wrap;
    top:0; left:0px;
    width:285px !important; height:205px !important;
    border:1px solid #eea; border-radius:2px;
    margin:0px; padding:0px;
    font-size:8pt;
    line-height:0.0;
    color:#fff; background-color:rgba(0,0,64,0.1) !important;
  }
  .GJTab{
    display:inline;
    position:relative;
    top:0px; left:0px;
    margin:0px; padding:2px;
    border:0px solid #000; border-radius:2px;
    width:90px; height:20px;
    font-family:Georgia;
    font-size:9pt;
    line-height:1.0;
    white-space:nowrap;
    color:#fff; background-color:rgba(0,0,64,0.7);
    text-align:center;
    vertical-align:middle;
  }
  .GJStat:focus{
    color:#df8 !important;
    background-color:rgba(32,32,32,1.0) !important;
    line-height:1.0;
  }
  .GJStat{
    display:inline;
    position:relative;
    top:0px; left:0px;
    margin:0px; padding:2px;
    border:0px solid #00f; border-radius:2px;
    width:166px; height:20px;
    font-family:monospace;
    font-size:9pt;
    line-height:1.0;
    color:#fff; background-color:rgba(0,0,64,0.2);
    text-align:center;
    vertical-align:middle;
  }
  .GJIcon{
    display:inline;
    position:relative;
    top:0px; left:1px;
    border:2px solid #44a;
    margin:0px; padding:1px;
    width:13.2; height:13.2px;
    border-radius:2px;
    font-family:Georgia;
    font-size:13.2px;
    line-height:1.0;
    white-space:nowrap;
    color:#fff; background-color:rgba(32,32,160,0.8);
    text-align:center;
    vertical-align:middle;
    text-shadow:0px 0px;
  }
  .GJText:focus{
    color:#fff !important;
    background-color:rgba(32,32,160,0.8) !important;
    line-height:1.0;
  }
  .GJText{
    display:inline;
    position:relative;
    top:0px; left:0px;
    border:0px solid #000; margin:0px; padding:0px;
    width:280px; height:160px;
    border:0px;
    font-family:Courier New,monospace !important;
    font-size:8pt;
    line-height:1.0;
    white-space:pre;
    color:#fff; xbackground-color:rgba(0,0,64,0.5);
    background-color:rgba(32,32,128,0.8) !important;
  }
  .GJMode{
    display:inline;
    position:relative;
    top:0px; left:0px;
    border:0px solid #000; border-radius:0px;
    margin:0px; padding:0px;
    width:280px; height:20px;
    font-size:9pt;
    line-height:1.0;
    white-space:nowrap;
```

```
    color:#fff; background-color:rgba(0,0,64,0.7);
    text-align:left;
    vertical-align:middle;
  }
</style>

<script id="gsh-script">
  // 2020-0909 added, permanet local storage
  // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
  var MyHistory = ""
  Permanent = localStorage;
  MyHistory = Permanent.getItem('MyHistory')
  if( MyHistory == null ){ MyHistory = "" }
  d = new Date()
  MyHistory = d.getTime()/1000+" "+document.URL+"\n" + MyHistory
  Permanent.setItem('MyHistory',MyHistory)
  //Permanent.setItem('MyWindow',window)

  var GJLog_Win = null
  var GJLog_Tab = null
  var GJLog_Stat = null
  var GJLog_Text = null
  var GJWin_Mode = null
  var FProductInterval = 0

  var GJ_FactoryID = -1
  var GJFactory = null
  if( e = document.getElementById('GJFactory_0') ){
    GJFactory_1.height = 0
    GJFactory = e
    e.setAttribute('class','GJFactory')
    var GJ_FactoryID = 0
  }else{
    GJFactory = GJFactory_1
    var GJ_FactoryID = 1
  }

  function GJFactory_Destroy(){
    gjf = GJFactory
    //gjf = document.getElementById('GJFactory')
    //alert('gfj='+gjf)
    if( gjf != null ){
        if( gjf.childNodes != null ){
            for( i = 0; i < gjf.childNodes.length; i++ ){
                gjf.removeChild(gjf.childNodes[i])
            }
        }
        gjf.innerHTML = ''
        gjf.style.width = 0
        gjf.style.height = 0
        gjf.removeAttribute('style')
        GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
        window.clearInterval(FProductInterval)
        return '-- Destroy: work product destroyed'
    }else{
        return '-- Destroy: work product not exist'
    }
  }

  var TransMode = false
  var onKeyControl = false
  var OnKeyShift = false
  var OnKeyAlt = false
  var OnKeyJ = false
  var OnKeyK = false
  var OnKeyL = false

  function GJWin_OnKeyUp(ev){
    keycode = ev.code;
    if( keycode == 'ShiftLeft' ){
        OnKeyShift = false
    }else
    if( keycode == 'ControlLeft' ){
        onKeyControl = false
    }else
    if( keycode == 'AltLeft' ){
        OnKeyAlt = false
    }else
    if( keycode == 'KeyJ' ){ OnKeyJ = false }else
    if( keycode == 'KeyK' ){ OnKeyK = false }else
    if( keycode == 'KeyL' ){ OnKeyL = false }else
    {
    }
    ev.preventDefault()
  }
  function and(a,b){ if(a){ if(b){ return true; } return false; } }
  function GJWin_OnKeyDown(ev){
    keycode = ev.code;
    mode = ''
    key = ''
    if( keycode == 'ControlLeft' ){
        onKeyControl = true
        ev.preventDefault()
        return;
    }else
    if( keycode == 'ShiftLeft' ){
        OnKeyShift = true
        ev.preventDefault()
        return;
    }else
    if( keycode == 'AltLeft' ){
        ev.preventDefault()
        OnKeyAlt = true
        return;
    }else
    if( keycode == 'Backquote' ){
        TransMode = !TransMode
        ev.preventDefault()
    }else
    if( and(keycode == 'Space', OnKeyShift) ){
        TransMode = !TransMode
        ev.preventDefault()
    }else
    if( keycode == 'ShiftRight' ){
        TransMode = !TransMode
    }else
    if( keycode == 'Escape' ){
        TransMode = true
        ev.preventDefault()
    }else
    if( keycode == 'Enter' ){
        TransMode = false
        //ev.preventDefault()
    }
    if( keycode == 'KeyJ' ){ OnKeyJ = true }else
    if( keycode == 'KeyK' ){ OnKeyK = true }else
    if( keycode == 'KeyL' ){ OnKeyL = true }else
    {
    }

    if( ev.altKey    ){ key += 'Alt+' }
    if( onKeyControl ){ key += 'Ctrl+' }
    if( OnKeyShift   ){ key += 'Shift+' }
    if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
    if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
    if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
    key += keycode

    if( TransMode ){
        //mode = "[\343\201\202r]"
        mode = "[あr]"
    }else{
        mode = '[---]'
    }
    ////  /gjmode.innerHTML = "[---]"
    GJWin_Mode.innerHTML = mode + ' ' + key
    //alert('Key:'+keycode)
    ev.stopPropagation()
    //ev.preventDefault()
  }
  function GJWin_OnScroll(ev){
    x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
    y = DragStatty = gsh.getBoundingClientRect().top.toFixed(0)
    GJLog_append('OnScroll: x='+x+',y='+y)
  }
  document.addEventListener('scroll',GJWin_OnScroll)
  function GJWin_OnResize(ev){
    w = window.innerWidth
    h = window.innerHeight
    GJLog_append('OnResize: w='+w+',h='+h)
  }
  window.addEventListener('resize',GJWin_OnResize)

  var DragStartX = 0
  var DragStartY = 0
  function GJWin_DragStart(ev){
    // maybe this is the grabbing position
    this.style.position = 'fixed'
    x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
    y = DragStatty = this.getBoundingClientRect().top.toFixed(0)
    GJLog_Stat.value = 'DragStart: x='+x+',y='+y
  }
  function GJWin_Drag(ev){
    x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
    this.style.left = x - DragStartX
    this.style.top = y - DragStartY
    this.style.zIndex = '30000'
    this.style.position = 'fixed'
    x = this.getBoundingClientRect().left.toFixed(0)
    y = this.getBoundingClientRect().top.toFixed(0)
    GJLog_Stat.value = 'x='+x+',y='+y
    ev.preventDefault()
    ev.stopPropagation()
  }
  function GJWin_DragEnd(ev){
    x = ev.clientX; y = ev.clientY
    //x = ev.pageX; y = ev.pageY
    this.style.left = x - DragStartX
    this.style.top = y - DragStartY
    this.style.zIndex = '30000'
    this.style.position = 'fixed'
    if( true ){
        console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
            +' parent='+this.parentNode.id)
    }
    x = this.getBoundingClientRect().left.toFixed(0)
    y = this.getBoundingClientRect().top.toFixed(0)
    GJLog_Stat.value = 'x='+x+',y='+y
    ev.preventDefault()
    ev.stopPropagation()
  }
  function GJWin_DragIgnore(ev){
    ev.preventDefault()
    ev.stopPropagation()
  }
  // 2020-09-15 let every object have console view!
  var GJ_ConsoleID = 0
  var PrevReport = new Date()
  function GJLog_StatUpdate(){
    txa = GJLog_Stat;
    if( txa == null ){
        return;
    }
    tmLap0 = new Date();
    p = txa.parentNode;
    pw = txa.getBoundingClientRect().width;
    ph = txa.getBoundingClientRect().height;
    //txa.value += '#'+p.id+' pw='+pw+', ph='+ph+'\n';
    tx1 = '#'+p.id+' pw='+pw+', ph='+ph+'\n';

    w = txa.getBoundingClientRect().width;
    h = txa.getBoundingClientRect().height;
    //txa.value += 'w='+w+', h='+h+'\n';
    tx1 += 'w='+w+', h='+h+'\n';

    //txa.value += '\n';
    //txa.value += DateShort() + '\n';
    tx1 += '\n';
    tx1 += DateShort() + '\n';
    tmLap1 = new Date();

    txa.value += tx1;
    tmLap2 = new Date();

    // vertical centering of the last line
    sHeight = txa.scrollHeight - 30; // depends on the font-size
    tmLap3 = new Date();

    txa.scrollTop = sHeight; // depends on the font-size
    tmLap4 = new Date();

    now = tmLap0.getTime();
    if( PrevReport == 0 || 10000 <= now-PrevReport ){
        PrevReport = now;
        console.log('StatBarUpdate:'
            + 'leng=' + txa.value.length + ' byte, '
            + 'time='  + (tmLap4 -tmLap0) + 'ms {'
            + 'tadd=' + (tmLap2 -tmLap1) + ', '
            + 'hcal=' + (tmLap3 -tmLap2) + ', '
            + 'scrl=' + (tmLap4 -tmLap3) + '}'
        );
    }
  }
  GJWin_StatUpdate = GJLog_StatUpdate;
  function GJ_showTime1(wid){
    //e = document.getElementById(wid);
    //console.log(wid.id+'.value.length='+wid.value.length)
    if( e != null ){
        //e.value = DateShort();
    }else{
        // should remove the Listener
    }
  }
  function GJWin_OnResizeTextarea(ev){
    this.value += 'resized:' + '\n'
  }
  function GJ_NewConsole(wname){
    wid = wname + '_' + GJ_ConsoleID
    GJ_ConsoleID += 1

    GJFactory.style.setProperty('width',360+'px'); //GJFsize
    GJFactory.style.setProperty('height',320+'px')
    e = GJFactory;
    console.log('GJFa #'+e.id+' from w='+e.style.width+', h='+e.style.height)

    if( GJFactory.innerHTML == "" ){
        GJFactory.innerHTML = '<'+'H3>GJ Factory_'+ GJ_FactoryID +'<'+'/H3><'+'hr>\n'
    }else{
        GJFactory.innerHTML += '<'+'hr>\n'
    }

    gjwin = GJLog_Win = document.createElement('span')
    gjwin.id = wid
    gjwin.setAttribute('class','GJWin')
    gjwin.setAttribute('draggable','true')
    gjwin.addEventListener('dragstart',GJWin_DragStart)
    gjwin.addEventListener('drag',GJWin_Drag)
    gjwin.addEventListener('dragend',GJWin_Drag)
    gjwin.addEventListener('dragover',GJWin_DragIgnore)
    gjwin.addEventListener('dragenter',GJWin_DragIgnore)
    gjwin.addEventListener('dragleave',GJWin_DragIgnore)
    gjwin.addEventListener('dragexit',GJWin_DragIgnore)
    gjwin.addEventListener('drop',GJWin_DragIgnore)
    gjwin.addEventListener('keydown',GJWin_OnKeyDown)

    gjtab = GJLog_Tab = document.createElement('textarea')
    gjtab.addEventListener('keydown',GJWin_OnKeyDown)
    gjtab.style.readonly = true
    gjtab.contenteditable = false
    gjtab.value = wid
    gjtab.id = wid + '_Tab'
    gjtab.setAttribute('class','GJTab')
    gjtab.setAttribute('spellcheck','false')
    gjwin.appendChild(gjtab)

    gjstat = GJLog_Stat = document.createElement('textarea')
    gjstat.addEventListener('keydown',GJWin_OnKeyDown)
    gjstat.id = wid + '_Stat'
    gjstat.value = DateShort()
    gjstat.setAttribute('class','GJStat')
    gjstat.setAttribute('spellcheck','false')
    gjwin.appendChild(gjstat)

    gjicon = document.createElement('span')
    gjicon.addEventListener('keydown',GJWin_OnKeyDown)
    gjicon.id = wid + '_Icon'
    gjicon.innerHTML = 'G<font color="#f44">J</font>'
    gjicon.setAttribute('class','GJIcon')
    gjicon.setAttribute('spellcheck','false')
    gjwin.appendChild(gjicon)

    gjtext = GJLog_Text = document.createElement('textarea')
    gjtext.addEventListener('keydown',GJWin_OnKeyDown)
    gjtext.addEventListener('keyup',GJWin_OnKeyUp)
    gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
    gjtext.id = wid + '_Text'
    gjtext.setAttribute('class','GJText')
    gjtext.setAttribute('spellcheck','false')
    gjwin.appendChild(gjtext)


    // user's mode as of IME
    gjmode = GJWin_Mode = document.createElement('textarea')
    gjmode.addEventListener('keydown',GJWin_OnKeyDown)
    gjmode.addEventListener('keydown',GJWin_OnKeyDown)
    gjmode.id = wid + '_Mode'
    gjmode.setAttribute('class','GJMode')
    gjmode.setAttribute('spellcheck','false')
    gjmode.innerHTML = '[---]'
    gjwin.appendChild(gjmode)

    gjwin.zIndex = 30000
    GJFactory.appendChild(gjwin)

    gjtab.scrollTop = 0
    gjstat.scrollTop = 0

    //x = gjwin.getBoundingClientRect().left.toFixed(0)
    //y = gjwin.getBoundingClientRect().top.toFixed(0)
    //gjwin.style.position = 'static'
    //gjwin.style.left = 0
    //gjwin.style.top = 0

    //update = '{'+wid+'.value=DateShort()}',
    update = '{GJ_showTime1('+wid+');}',
    // 2020-09-19 this causes memory leaks
    //FProductInterval = window.setInterval(update,200)
    //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
    //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
    FProductInterval = window.setInterval(GJ_showTime1,200,gjstat);
    return update
  }
  function xxxGJF_StripClass(){
    GJLog_Win.style.removeProperty('width')
    GJLog_Tab.style.removeProperty('width')
    GJLog_Stat.style.removeProperty('width')
    GJLog_Text.style.removeProperty('width')
    return "Stripped classes"
  }
  function isElem(id){
    return document.getElementById(id) != null
  }
```

```
  function GJLog_append(...args){
    txt = GJLog_Text;
    if( txt == null ){
        return; // maybe GJLog element is removed
    }
    logs = args.join(' ')
    txt.value += logs + '\n'
    txt.scrollTop = txt.scrollHeight
    //GJLog_Stat.value = DateShort()
  }
  //window.addEventListener('time',GJLog_StatUpdate)
  function test_GJ_Console(){
    window.setInterval(GJLog_StatUpdate,1000);
    GJ_NewConsole('GJ_Console')
    e = GJFactory;
    console.log('GJF0 #'+e.id+' from w='+e.style.width+', h='+e.style.height)
    e.style.width = 360; //GJFsize
    e.style.height = 320;
    console.log('GJF0 #'+e.id+' to w='+e.style.width+', h='+e.style.height)
  }
  /// test_GJ_Console();

  var StopConsoleLog = true
  // 2020-09-15 added,
  // log should be saved to permanet memory
  // const px = new Proxy(console.log,{ alert() })
  __console_log = console.log
  __console_info = console.info
  __console_warn = console.warn
  __console_error = console.error
  __console_exception = console.exception
  // should pop callstack info.
  console.exception = function(...args){
    __console_exception(...args)
    alert('-- got console.exception("'+args+'")')
  }
  console.error = function(...args){
    __console_error(...args)
    alert('-- got console.error("'+args+'")')
  }
  console.warn = function(...args){
    __console_warn(...args)
    alert('-- got console.warn("'+args+'")')
  }
  console.info = function(...args){
    alert('-- got console.info("'+args+'")')
    __console_info(...args)
  }
  console.log = function(...args){
    __console_log(...args)
    if( StopConsoleLog ){
        return;
    }
    if( 0 <= args[0].indexOf('!') ){
        //alert('-- got console.log("'+args+'")')
    }
    GJLog_append(...args)
  }

//document.getElementById('GshFaviconURL').href = GShellFavicon
document.getElementById('GshFaviconURL').href = GShellInsideIcon
//document.getElementById('GshFaviconURL').href = ITSmoreQR
//document.getElementById('GshFaviconURL').href = GSellLogo

// id of GShell HTML elemets
var E_BANNER = "GshBanner" // banner element in HTML
var E_FOOTER = "GshFooter" // footer element in HTML
var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
var E_GOCODE = "gsh-gocode" // Golang code of GSHell
var E_TODO   = "gsh-todo"   // TODO of GSHell
var E_DICT   = "gsh-dict"   // Dictionaly of GSHell

function bannerElem(){ return document.getElementById(E_BANNER); }
function bannerStyleFunc(){ return bannerElem().style; }
var bannerStyle = bannerStyleFunc()
bannerStyle.backgroundImage = "url("+GSellLogo+")";
//bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
//bannerStyle.backgroundImage = "url("+GShellFavicon+")";
GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";

function footerElem(){ return document.getElementById(E_FOOTER); }
function footerStyle(){ return footerElem().sytle; }
//footerElem().style.backgroundImage="url("+ITSmoreQR+")";
//footerStyle().backgroundImage = "url("+ITSmoreQR+")";

function html_fold(e){
    if( e.innerHTML == "Fold" ){
        e.innerHTML = "Unfold"
        document.getElementById('gsh-menu-exit').innerHTML=""
        document.getElementById('GshStatement').open=false
        GshFeatures.open = false
        document.getElementById('html-src').open=false
        document.getElementById(E_GINDEX).open=false
        document.getElementById(E_GOCODE).open=false
        document.getElementById(E_TODO).open=false
        document.getElementById('references').open=false
    }else{
        e.innerHTML = "Fold"
        document.getElementById('GshStatement').open=true
        GshFeatures.open = true
        document.getElementById(E_GINDEX).open=true
        document.getElementById(E_GOCODE).open=true
        document.getElementById(E_TODO).open=true
        document.getElementById('references').open=true
    }
}
function html_pure(e){
    if( e.innerHTML == "Pure" ){
        document.getElementById('gsh').style.display=true
        //document.style.display = false
        e.innerHTML = "Unpure"
    }else{
        document.getElementById('gsh').style.display=false
        //document.style.display = true
        e.innerHTML = "Pure"
    }
}

var bannerIsStopping = false
//NOTE: .com/JSREF/prop_style_backgroundposition.asp
function shiftBG(){
    bannerIsStopping = !bannerIsStopping
    bannerStyle.backgroundPosition = "0 0";
}
```

```
// status should be inherited on Window Fork(), so use the status in DOM
function html_stop(e,toggle){
    if( toggle ){
        if( e.innerHTML == "Stop" ){
            bannerIsStopping = true
            e.innerHTML = "Start"
        }else{
            bannerIsStopping = false
            e.innerHTML = "Stop"
        }
    }else{
        // update JavaScript variable from DOM status
        if( e.innerHTML == "Stop" ){ // shown if it's running
            bannerIsStopping = false
        }else{
            bannerIsStopping = true
        }
    }
}
html_stop(document.getElementById('GshMenuStop'),false) // onInit.
//html_stop(bannerElem(),false) // onInit.

//https://www.w3schools.com/jsref/met_win_setinterval.asp
function shiftBanner(){
    var now = new Date().getTime();
    //"console.log("now="+(now%10))
    if( !bannerIsStopping ){
        bannerStyle.backgroundPosition = ((now/10)%100000)+" 0";
    }
}
window.setInterval(shiftBanner,10); // onInit.

//  <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
// from embedded html to standalone page
var MyChildren = 0
function html_fork(){
    GJFactory_Destroy()
    MyChildren += 1
    WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
    newwin = window.open("",WinId,"");
    src = document.getElementById("gsh");
    srchtml = src.outerHTML
    newwin.document.write("/*<"+"html>\n");
    newwin.document.write(srchtml);
    newwin.document.write("<"+"/html>\n");
    newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
    newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
    newwin.document.close();
    newwin.focus();
}
function html_close(){
    window.close()
}
function win_jump(win){
    //win = window.top;
    win = window.openner; // https://developer.mozilla.org/ja/docs/Web/API/window.opener
    if( win == null ){
        console.log("jump to window.opener("+win+")(Error)\n")
    }else{
        console.log("jump to window.opener("+win+")\n")
        win.focus();
    }
}

// 0.2.9 2020-0902 created chekcsum of HTML
CRC32UNIX = 0x04C11DB7 // Unix cksum
function byteCRC32add(bigcrc,octstr,octlen){
    var crc = new Int32Array(1)
    crc[0] = bigcrc

    let oi = 0
    for( ; oi < octlen; oi++ ){
        var oct = new Int8Array(1)
        oct[0] = octstr[oi]
        for( bi = 0; bi < 8; bi++ ){
            //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
            ovf1 = crc[0] < 0 ? 1 : 0
            ovf2 = oct[0] < 0 ? 1 : 0
            ovf = ovf1 ^ ovf2
            oct[0] <<= 1
            crc[0] <<= 1
            if( ovf ){ crc[0] ^= CRC32UNIX }
        }
    }
    //console.log("--CRC32 byteAdd return crc="+crc[0]+","+oi+"/"+octlen+"\n")
    return crc[0];
}
function strCRC32add(bigcrc,stri,strlen){
    var crc = new Uint32Array(1)
    crc[0] = bigcrc
    var code = new Uint8Array(strlen);
    for( i = 0; i < strlen; i++){
        code[i] = stri.charCodeAt(i) // not charAt() !!!!
        //console.log("=== "+code[i].toString(16)+" <<== "+stri[i]+"\n")
    }
    crc[0] = byteCRC32add(crc,code,strlen)
    //console.log("--CRC32 strAdd return crc="+crc[0]+"\n")
    return crc[0]
}
function byteCRC32end(bigcrc,len){
    var crc = new Uint32Array(1)
    crc[0] = bigcrc
    var slen = new Uint8Array(4)
    let li = 0
        for( ; li < 4; ){
                slen[li] = len
        li += 1
                len >>= 8
                if( len == 0 ){
                        break
        }
        }
        crc[0] = byteCRC32add(crc[0],slen,li)
        crc[0] ^= 0xFFFFFFFF
        return crc[0]
}
function strCRC32(stri,len){
    var crc = new Uint32Array(1)
    crc[0] = 0
    crc[0] = strCRC32add(0,stri,len)
    crc[0] = byteCRC32end(crc[0],len)
    //console.log("--CRC32 "+crc[0]+" "+len+"\n")
    return crc[0]
}
```

```

DestroyGJLink = null; // to be replaced
DestroyFooter = null; // to be defined
DestroyEventSharingCodeview = function dummy(){}
Destroy_WirtualDesktop = function(){}
DestroyNaviButtons = function(){}

function getSourceText(){
    if( DestroyFooter != null ) DestroyFooter();
    version = document.getElementById('GshVersion').innerHTML
    sfavico = document.getElementById('GshFaviconURL').href;
    sbanner = document.getElementById('GshBanner').style.backgroundImage;
    spositi = document.getElementById('GshBanner').style.backgroundPosition;

    if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
    if( DestroyGJLink != null ) DestroyGJLink();
    DestroyEventSharingCodeview();
    Destroy_WirtualDesktop();
    DestroyIndexBar();
    DestroyNaviButtons();

    // these should be removed by CSS selector or class, after sevaed to non-printed attribute
    GshBanner.removeAttribute('style');
    document.getElementById('GshMenuSign').removeAttribute("style");
    styleGMenu = GMenu.getAttribute("style")
    GMenu.removeAttribute("style");
    styleGStat = GStat.getAttribute("style")
    GStat.removeAttribute("style");
    styleGTop = GTop.getAttribute("style")
    GTop.removeAttribute("style");
    styleGshGrid = GshGrid.getAttribute("style")
    GshGrid.removeAttribute("style");
    //styleGPos = GPos.getAttribute("style");
    //GPos.removeAttribute("style");
    //GPos.innerHTML = "";
    //styleGLog = GLog.getAttribute("style");
    //GLog.removeAttribute("style");
    //GLog.innerHTML = "";
    styleGShellPlane = GShellPlane.getAttribute("style")
    GShellPlane.removeAttribute("style")
    styleRawTextViewer = RawTextViewer.getAttribute("style")
    RawTextViewer.removeAttribute("style")
    styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
    RawTextViewerClose.removeAttribute("style")

    GshFaviconURL.href = "";

    //it seems that interHTML and outerHTML generate style="" for these (??)
    //GshBanner.removeAttribute('style');
    //GshFooter.removeAttribute('style');
    //GshMenuSign.removeAttribute('style');
    GshBanner.style=""
    GshMenuSign.style=""

    textarea = document.createElement("textarea")
    srchtml = document.getElementById("gsh").outerHTML;
        //textarea = document.createElement("textarea")
        // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
        // with Chromium?/ after reloading from file:///
        textarea.innerHTML = srchtml
    // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
        var rawtext = textarea.value
        //textarea.destroy()
        //rawtext = gsh.textContent // this removes #include <FILENAME> too
        var orgtext = ""
         + "/*<"+"html>\n"  // lost preamble text
         + rawtext
         + "<"+"/html>\n"   // lost trail text
        ;

    tlen = orgtext.length
    //console.log("getSourceText: length="+tlen+"\n")
    document.getElementById('GshFaviconURL').href                 = sfavico;

    document.getElementById('GshBanner').style.backgroundImage    = sbanner;
    document.getElementById('GshBanner').style.backgroundPosition = spositi;

    GStat.setAttribute("style",styleGStat)
    GMenu.setAttribute("style",styleGMenu)
    GTop.setAttribute("style",styleGTop)
    //GLog.setAttribute("style",styleGLog)
    //GPos.setAttribute("style",styleGPos)
    GshGrid.setAttribute("style",styleGshGrid)
    GShellPlane.setAttribute("style",styleGShellPlane)
    RawTextViewer.setAttribute("style",styleRawTextViewer)
    RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
    canontext = orgtext.replace(' style=""','')
    // open="" too
    return canontext
}
function getDigest(){
    var text = ""
    text = getSourceText()
    var digest = ""
    tlen = text.length
    digest = strCRC32(text,tlen) + " " + tlen
    return { text, digest }
}
function html_digest(){
    version = document.getElementById('GshVersion').innerHTML
    let {text, digest} = getDigest()
    alert("cksum: " + digest + " " + version)
}
function charsin(stri,char){
    ln = 0;
    for( i = 0; i < stri.length; i++ ){
        if( stri.charCodeAt(i) == char.charCodeAt(0) )
            ln++;
    }
    return ln;
}

//class digestElement extends HTMLElement { }
//< script>customElements.define('digest',digestElement)< /script>
function showDigest(e){
    result = 'version=' + GshVersion.innerHTML + '\n'
    result += 'lines='   + e.dataset.lines   + '\n'
        + 'length='  + e.dataset.length  + '\n'
        + 'crc32u='  + e.dataset.crc32u  + '\n'
        + 'time='    + e.dataset.time    + '\n';

    alert(result)
}

```

```
function html_sign(e){
    if( RawTextViewer.style.zIndex == 1000 ){
        hideRawTextViewer()
        return
    }
    DestroyIndexBar();
    DestroyNaviButtons();
    DestroyEventSharingCodeview();
    Destroy_WirtualDesktop();
    GJFactory_Destroy()
    if( DestroyGJLink != null ) DestroyGJLink();
    //gsh_digest_.innerHTML = "";
    text = getSourceText() // the original text
    tlen = text.length
    digest = strCRC32(text,tlen)
    //gsh_digest_.innerHTML = digest + " " + tlen
    //text = getSourceText() // the text with its digest
    Lines = charsin(text,'\n')

    name = "gsh"
    sid = name + "-digest"
    d = new Date()
    signedAt = d.getTime()

    sign = '/'+'*<'+'span\n'
        + ' id="' + sid + '"\n'
        + ' class="_digest_"\n'
        + ' data-target-id="'+name+'"\n'
        + ' data-crc32u="'  + digest    + '"\n'
        + ' data-length="'  + tlen      + '"\n'
        + ' data-lines="'   + Lines     + '"\n'
        + ' data-time="'    + signedAt  + '"\n'
        + ' ><' + '/span>\n*'+'/\n'

    text = sign + text

    txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td>'
        + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">'
    for( i = 1; i <= Lines; i++ ){
        txthtml += i.toString() + '\n'
    }
    txthtml += ""
        + '<' + '/textarea>'
        + '<' + '/td><' + 'td>'
        + '<' + 'textarea cols=150 rows=' + Lines + 'spellcheck="false"'
        + ' class="LineNumbered">'
        + text + '<'+'/textarea>'
        + '<' + '/td><' + '/tr><' + '/table>'

    for( i = 1; i <= 30; i++ ){
        txthtml += '<br>\n'
    }
    RawTextViewer.innerHTML = txthtml
    RawTextViewer.spellcheck = false // (spelcheck above seems ineffective)

    btn = e
    e.style.color = "rgba(128,128,255,0.9)";
    y = e.getBoundingClientRect().top.toFixed(0)
    //h = e.getBoundingClientRect().height.toFixed(0)
    RawTextViewer.style.top = Number(y) + 30
    RawTextViewer.style.left = 100;
    RawTextViewer.style.height = window.innerHeight - 20;
    //RawTextViewer.style.Opacity = 1.0;
    //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
    RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
    RawTextViewer.style.zIndex = 1000;
    RawTextViewer.style.display = true;

    if( RawTextViewerClose.style == null ){
        RawTextViewerClose.style = "";
    }
    RawTextViewerClose.style.top = Number(y) + 10
    RawTextViewerClose.style.left = 100;
    RawTextViewerClose.style.zIndex = 1001;

    ScrollToElement(CurElement,RawTextViewerClose)
}
function hideRawTextViewer(){
    RawTextViewer.style.left = 10000;
    RawTextViewer.style.zIndex = -100;
    RawTextViewer.style.Opacity = 0.0;
    RawTextViewer.style = null
    RawTextViewer.innerHTML = "";

    GshMenuSign.style.color = "rgba(255,128,128,1.0)";
    RawTextViewerClose.style.top = 0;
    RawTextViewerClose.style = null
}

// source code viewr
function frame_close(){
    srcframe = document.getElementById("src-frame");
    srcframe.innterHTML = "";
    //srcframe.style.cols = 1;
    srcframe.style.rows = 1;
    srcframe.style.height = 0;
    srcframe.style.display = false;
    src = document.getElementById("SrcTextarea");
    src.innerHML = ""
    //src.cols = 0
    src.rows = 0
    src.display = false
    //alert("--closed--")
}
//<!-- | <span onclick="html_view();">Source</span> -->
//<!-- | <span onclick="frame_close();">SourceClose</span> -->
//<!--| <span>Download</span> -->
function frame_open(){
    DestroyIndexBar();
    DestroyNaviButtons();
    if( DestroyFooter != null ) DestroyFooter();
    document.getElementById('GshFaviconURL').href = "";
    oldsrc = document.getElementById("GENSRC");
    if( oldsrc != null ){
        //alert("--I--(erasing old text)")
        oldsrc.innterHTML = "";
        return
    }else{
        //alert("--I--(no old text)")
    }
    styleBanner = GshBanner.getAttribute("style")
    GshBanner.removeAttribute("style")
    if( document.getElementById('GJC_1') ){ GJC_1.remove() }

    GshFaviconURL.href = "";
    GStat.removeAttribute('style')
    GshGrid.removeAttribute('style')
    GshMenuSign.removeAttribute('style')
    //GPos.removeAttribute('style')
    //GPos.innerHTML = "";
    //GLog.removeAttribute('style')
    //GLog.innerHTML = "";
    GMenu.removeAttribute('style')
    GTop.removeAttribute('style')
    GShellPlane.removeAttribute('style')
    RawTextViewer.removeAttribute('style')
    RawTextViewerClose.removeAttribute('style')

    if( DestroyGJLink != null ) DestroyGJLink();
    GJFactory_Destroy()
    Destroy_WirtualDesktop();
    DestroyEventSharingCodeview();

    src = document.getElementById("gsh");
    srchtml = src.outerHTML
    srcframe = document.getElementById("src-frame");
    srcframe.innerHTML = ""
     + "<"+"cite id=\"GENSRC\">\n"
     + "<"+"style>\n"
     + "#GENSRC textarea{tab-size:4;}\n"
     + "#GENSRC textarea{-o-tab-size:4;}\n"
     + "#GENSRC textarea{-moz-tab-size:4;}\n"
     + "#GENSRC textarea{spellcheck:false;}\n"
     + "</"+"style>\n"
     + "<"+'textarea id="SrcTextarea" cols=100 rows=20 class="gsh-code" spellcheck="false">'
     + "/*<"+"html>\n"  // lost preamble text
     + srchtml
     + "<"+"/html>\n"   // lost trail text
     + "</"+"textarea>\n"
     + "</"+"cite><!-- GENSRC -->\n";

    //srcframe.style.cols = 80;
    //srcframe.style.rows = 80;

    GshBanner.setAttribute('style',styleBanner)
}
function fill_CSSView(){
    part = document.getElementById('GshStyleDef')
    view = document.getElementById('gsh-style-view')
    view.innerHTML = ""
     + "<"+'textarea cols=100 rows=20 class="gsh-code">'
     + part.innerHTML
     + "<"+"/textarea>"
}
function fill_JavaScriptView(){
    jspart = document.getElementById('gsh-script')
    view = document.getElementById('gsh-script-view')
    view.innerHTML = ""
     + "<"+'textarea cols=100 rows=20 class="gsh-code">'
     + jspart.innerHTML
     + "<"+"/textarea>"
}
function fill_DataView(){
    part = document.getElementById('gsh-data')
    view = document.getElementById('gsh-data-view')
    view.innerHTML = ""
     + "<"+'textarea cols=100 rows=20 class="gsh-code">'
     + part.innerHTML
     + "<"+"/textarea>"
}
function jumpto_StyleView(){
    jsview = document.getElementById('html-src')
    jsview.open = true
    jsview = document.getElementById('gsh-style-frame')
    jsview.open = true
    fill_CSSView()
}
function jumpto_JavaScriptView(){
    jsview = document.getElementById('html-src')
    jsview.open = true
    jsview = document.getElementById('gsh-script-frame')
    jsview.open = true
    fill_JavaScriptView()
}
function jumpto_DataView(){
    jsview = document.getElementById('html-src')
    jsview.open = true
    jsview = document.getElementById('gsh-data-frame')
    jsview.open = true
    fill_DataView()
}
function jumpto_WholeView(){
    jsview = document.getElementById('html-src')
    jsview.open = true
    jsview = document.getElementById('gsh-whole-view')
    jsview.open = true
    frame_open()
}
function html_view(){
    html_stop();

    banner = document.getElementById('GshBanner').style.backgroundImage;
    footer = document.getElementById('GshFooter').style.backgroundImage;
    document.getElementById('GshBanner').style.backgroundImage = "";
    document.getElementById('GshBanner').style.backgroundPosition = "";
    document.getElementById('GshFooter').style.backgroundImage = "";

    //srcwin = window.open("","CodeView2","");
    srcwin = window.open("","","");
    srcwin.document.write("<span id=\"gsh\">\n");

    src = document.getElementById("gsh");
    srcwin.document.write("<"+"style>\n");
    srcwin.document.write("textarea{tab-size:4;}\n");
    srcwin.document.write("textarea{-o-tab-size:4;}\n");
    srcwin.document.write("textarea{-moz-tab-size:4;}\n");
    srcwin.document.write("</style>\n");
    srcwin.document.write("<h2>\n");
    srcwin.document.write("<"+"span onclick=\"window.close();\">Close</span> | \n");
    //srcwin.document.write("<"+"span onclick=\"html_stop();\">Run</span>\n");
    srcwin.document.write("</h2>\n");
    srcwin.document.write("<"+"textarea id=\"gsh-src-src\" cols=100 rows=60>");
    srcwin.document.write("/*<"+"html>\n");
    srcwin.document.write("<"+"span id=\"gsh\">");
    srcwin.document.write(src.innerHTML);
    srcwin.document.write("<"+"/span><"+"/html>\n");
    srcwin.document.write("</"+"textarea>\n");

    document.getElementById('GshBanner').style.backgroundImage = banner;
    document.getElementById('GshFooter').style.backgroundImage = footer

    sty = document.getElementById("GshStyleDef");
    srcwin.document.write("<"+"style>\n");
    srcwin.document.write(sty.innerHTML);
    srcwin.document.write("<"+"/style>\n");

    run = document.getElementById("gsh-script");
    srcwin.document.write("<"+"script>\n");
    srcwin.document.write(run.innerHTML);
    srcwin.document.write("<"+"/script>\n");

    srcwin.document.write("<"+"/span><"+"/html>\n"); // gsh span
    srcwin.document.close();
    srcwin.focus();
}
GSH = document.getElementById("gsh")

//GSH.onclick = "alert('Ouch!')"
//GSH.css = "{background-color:#eef;}"
//GSH.style = "background-color:#eef;"
//GSH.style.display = false;
//alert('Ouch0!')
//GSH.style.display = true;

// 2020-0904 created, tentative
document.addEventListener('keydown',jgshCommand);
//CurElement = GshStatement
CurElement = GshMenu
MemElement = GshMenu

function nextSib(e){
    n = e.nextSibling;
    for( i = 0; i < 100; i++ ){
        if( n == null ){
            break;
        }
        if( n.nodeName == "DETAILS" ){
            return n;
        }
        n = n.nextSibling;
    }
    return null;
}
function prevSib(e){
    n = e.previousSibling;
    for( i = 0; i < 100; i++ ){
        if( n == null ){
            break;
        }
        if( n.nodeName == "DETAILS" ){
            return n;
        }
        n = n.previousSibling;
    }
    return null;
}
function setColor(e,eName,eColor){
    if( e.hasChildNodes() ){
        s = e.childNodes;
        if( s != null ){
            for( ci = 0; ci < s.length; ci++ ){
                if( s[ci].nodeName == eName ){
                    s[ci].style.color = eColor;
                    //s[ci].style.backgroundColor = eColor;
                    break;
                }
            }
        }
    }
}

// https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
function showCurElementPosition(ev){
//  if( document.getElementById("GPos") == null ){
//      return;
//  }
//  if( GPos == null ){
//      return;
//  }
    e = CurElement
    y = e.getBoundingClientRect().top.toFixed(0)
    x = e.getBoundingClientRect().left.toFixed(0)

    h = ev + " "
    h += 'y='+y+", "+ 'x='+x+" -- "
    h += "w=" + window.innerWidth + ", h=" + window.innerHeight + " -- "
    //GPos.test = h
    //GPos.innerHTML = h
//  GPos.innerHTML = h
}

function zero2(n){
    if( n < 10 ){
        return '0' + n;
    }else{
        return n;
    }
}
function DateHourMin(){
    d = new Date();
    //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
    return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
}
function DateShort(){
    d = new Date()
    return  d.getFullYear()
        + "/" + zero2(d.getMonth())
        + "/" + zero2(d.getDate())
        + " " + zero2(d.getHours())
        + ":" + zero2(d.getMinutes())
        + ":" + zero2(d.getSeconds())
}
function DateLong0(ms){
    d = new Date();
    d.setTime(ms);
    return
        //d.getFullYear() + "/" + d.getMonth() + "/" + d.getDate() + " "
        //+ d.getHours() + ":" + d.getMinutes() + ":" + d.getSeconds()
        DateShort()
        + "." + d.getMilliseconds()
        + " " + d.getTimezoneOffset()/60
        + " "
        + d.getTime()  + "." + d.getMilliseconds()

}
function DateLong(){
    return DateLong0(new Date());
}
function GShellMenu(e){
    //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
    showGShellPlane()
}
// placements of planes
function GShellResizeX(ev){
    //if( document.getElementById("GMenu") != null ){
        GMenu.style.left = window.innerWidth - 100
        GMenu.style.top = window.innerHeight - 90 - 200
        //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)

    //}
    GStat.style.width = window.innerWidth
    //if( document.getElementById("GPos") != null ){
        //GPos.style.width = window.innerWidth
        //GPos.style.top = window.innerHeight - 30; //GPos.style.height
    //}
    //if( document.getElementById("GLog") != null ){
    //  GLog.style.width = window.innerWidth
        //GLog.innerHTML = ""
    //}
    //if( document.getElementById("GLog") != null ){
        //GLog.innerHTML = "Resize: w=" + window.innerWidth +
        //", h=" + window.innerHeight
    //}
    showCurElementPosition(ev)
}
function GShellResize(){
    GShellResizeX("[RESIZE]")
}
window.onresize = GShellResize
var prevNode = null
var LogMouseMoveOverElement = false;
function GJSH_OnMouseMove(ev){
    if( LogMouseMoveOverElement == false ){
        return;
    }
    x = ev.clientX
    y = ev.clientY
    d = new Date()
    t = d.getTime() / 1000
    if( document.elementFromPoint ){
        e = document.elementFromPoint(x,y)
        if( e != null ){
            if( e == prevNode ){
            }else{
                console.log('Mo-'+t+'('+x+','+y+') '
                    +e.nodeType+' '+e.tagName+'#'+e.id)
                prevNode = e
            }
        }else{
            console.log(t+'('+x+','+y+') no element')
        }
    }else{
        console.log(t+'('+x+','+y+') no elementFromPoint')
    }
}
window.addEventListener('mousemove',GJSH_OnMouseMove);

function GJSH_OnMouseMoveScreen(ev){
    x = ev.screenX
    y = ev.screenY
    d = new Date()
    t = d.getTime() / 1000
    console.log(t+'('+x+','+y+') no elementFromPoint')
}
//screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);

function ScrollToElement(oe,ne){
    ne.scrollIntoView()
    ny = ne.getBoundingClientRect().top.toFixed(0)
    nx = ne.getBoundingClientRect().left.toFixed(0)
    //GLog.innerHTML = "["+ny+","+nx+"]"
    //window.scrollTo(0,0)

    GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
    GshGrid.style.left = '250px';
    GshGrid.style.zIndex = 0
    if( false ){
        oy = oe.getBoundingClientRect().top.toFixed(0)
        ox = oe.getBoundingClientRect().left.toFixed(0)
        y = e.getBoundingClientRect().top.toFixed(0)
        x = e.getBoundingClientRect().left.toFixed(0)
        window.scrollTo(x,y)
        ny = e.getBoundingClientRect().top.toFixed(0)
        nx = e.getBoundingClientRect().left.toFixed(0)
        //GLog.innerHTML = "["+oy+","+ox+"]->["+y+","+x+"]->["+ny+","+nx+"]"
    }
}
function showGShellPlane(){
    if( GShellPlane.style.zIndex == 0 ){
        GShellPlane.style.zIndex = 1000;
        GShellPlane.style.left = 30;
        GShellPlane.style.height = 320;
        GShellPlane.innerHTML = DateLong() + "<br>" +
            "-- History --<br>" + MyHistory;
    }else{
        GShellPlane.style.zIndex = 0;
        GShellPlane.style.left = 0;
        GShellPlane.style.height = 50;
        GShellPlane.innerHTML = "";
    }
}
var SuppressGJShell = false
function jgshCommand(kevent){
    if( SuppressGJShell ){
        return
    }
    key = kevent
    keycode = key.code
    //GStat.style.width = window.innerWidth
    GStat.style.backgroundColor = "rgba(0,0,0,0.4)"

    console.log("JSGsh-Key:"+keycode+"(^-^)//")
    if( keycode == "Slash" ){
        console.log('('+x+','+y+') ')
        e = document.elementFromPoint(x,y)
        console.log('('+x+','+y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
    }else
    if( keycode == "Digit0" ){ // fold side-bar
        // "Zero page"
        showGShellPlane();
    }else
    if( keycode == "Digit1" ){ // fold side-bar
        primary.style.width = "94%"
        secondary.style.width = "0%"
        secondary.style.opacity = 0
        GStat.innerHTML = "[Single Column View]"
    }else
    if( keycode == "Digit2" ){ // unfold side-bar
        primary.style.width = "58%"
        secondary.style.width = "36%"
        secondary.style.opacity = 1
        GStat.innerHTML = "[Double Column View]"
    }else
    if( keycode == "KeyU" ){ // fold/unfold all
        html_fold(GshMenuFold);
        location.href = "#"+CurElement.id;
    }else
    if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
        CurElement.open = !CurElement.open;
    }else
    if( keycode == "ArrowRight" ){ // unfold the element
        CurElement.open = true
    }else
    if( keycode == "ArrowLeft" ){ // unfold the element
        CurElement.open = false
    }else
    if( keycode == "KeyI" ){ // inspect the element
        e = CurElement
        //GLog.innerHTML =
        GJLog_append("Current Element: " + e + "<br>"
            + "name="+e.nodeName + ", "
            + "id="+e.id + ", "
            + "children="+e.childNodes.length + ", "
            + "parent="+e.parentNode.id + "<br>"
            + "text="+e.textContent)
        GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
        return
    }else
    if( keycode == "KeyM" ){ // memory the position
        MemElement = CurElement
    }else
    if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
        e = nextSib(CurElement)
        if( e != null ){
            setColor(CurElement,"SUMMARY","#fff")
            setColor(e,"SUMMARY","#8f8") // should be complement ?
            oe = CurElement
            CurElement = e
            //location.href = "#"+e.id;
            ScrollToElement(oe,e)
        }
    }else
    if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
        oe = CurElement
        e = prevSib(CurElement)
        if( e != null ){
            setColor(CurElement,"SUMMARY","#fff")
            setColor(e,"SUMMARY","#8f8") // should be complement ?
            CurElement = e
            //location.href = "#"+e.id;
            ScrollToElement(oe,e)
        }else{
            e = document.getElementById("GshBanner")
            if( e != null ){
                setColor(CurElement,"SUMMARY","#fff")
                CurElement = e
                ScrollToElement(oe,e)
            }else{
                e = document.getElementById("primary")
                if( e != null ){
                    setColor(CurElement,"SUMMARY","#fff")
                    CurElement = e
                    ScrollToElement(oe,e)
                }
            }
        }
    }else
    if( keycode == "KeyR" ){
        location.reload()
    }else
    if( keycode == "KeyJ" ){
        GshGrid.style.top = '120px';
        GshGrid.innerHTML = '(>_<){Down}';
    }else
    if( keycode == "KeyK" ){
        GshGrid.style.top = '0px';
        GshGrid.innerHTML = '(^-^){Up}';
    }else
    if( keycode == "KeyH" ){
        GshGrid.style.left = '0px';
        GshGrid.innerHTML = "('_'){Left}";
    }else
    if( keycode == "KeyL" ){
        //GLog.innerHTML +=
        GJLog_append(
            'screen='+screen.width+'px'+'<br>'+
            'window='+window.innerWidth+'px'+'<br>'
        )
        GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
        GshGrid.innerHTML = '(@_@){Right}';
    }else
    if( keycode == "KeyS" ){
        html_stop(GshMenuStop,true)
    }else
    if( keycode == "KeyF" ){
        html_fork()
    }else
    if( keycode == "KeyC" ){
        window.close()
    }else
    if( keycode == "KeyD" ){
        html_digest()
    }else
    if( keycode == "KeyV" ){
        e = document.getElementById('gsh-digest')
        if( e != null ){
            showDigest(e)
        }
    }

    showCurElementPosition("["+key.code+"] --");
    //if( document.getElementById("GPos") != null ){
        //GPos.innerHTML += "["+key.code+"] --"
    //}
    //GShellResizeX("["+key.code+"] --");
}
GShellResizeX("[INIT]");

DisplaySize = '-- Display: '+ 'screen='+screen.width+'px, '+'window='+window.innerWidth+'px';

let {text, digest} = getDigest()
//GLog.innerHTML +=
GJLog_append(
    '-- GShell: ' + GshVersion.innerHTML + '\n' +
    '-- Digest: ' + digest + '\n' +
    DisplaySize
    //+ "<br>" + "-- LastVisit:<br>" + MyHistory
)
GShellResizeX(null);

// <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
//Convert a string into an ArrayBuffer
//from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
function str2ab(str) {
    const buf = new ArrayBuffer(str.length);
    const bufView = new Uint8Array(buf);
    for (let i = 0, strLen = str.length; i < strLen; i++) {
        bufView[i] = str.charCodeAt(i);
    }
    return buf;
}
function importPrivateKey(pem) {
  const binaryDerString = window.atob(pemContents);
  const binaryDer = str2ab(binaryDerString);
  return window.crypto.subtle.importKey(
    "pkcs8",
    binaryDer,
    {
     name: "RSA-PSS",
     modulusLength: 2048,
     publicExponent: new Uint8Array([1, 0, 1]),
     hash: "SHA-256",
    },
    true,
    ["sign"]
  );
}
//importPrivateKey(ppem)

//key = {}
//buf = "abc"
//enc = "xyzxxxxxx"; //crypto.publicEncrypt(key,buf)
//b64 = btoa(enc)
//dec = atob(b64)
//GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
</script>
*/

/*
<!-- ----- GJConsole BEGIN { ----- -->
<span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
<details><summary>GJ Console</summary>
<p>
<span id="GJE_RootNode0"></span>
</p>
<style id="GJConsoleStyle">
  .GJConsole {
    z-index:1000;
    width:400; height:200px;
    margin:2px;
    color:#fff; background-color:#66a;
    font-size:12px; font-family:monospace,Courier New;
  }
</style>

<script id="GJConsoleScript" class="GJConsole">
  var PS1 = "% "
  function GJC_Keydown(keyevent){
    key = keyevent.code
    if( key == "Enter" ){
        GJC_Command(this)
        this.value += "\n" + PS1 // prompt
    }else
    if( key == "Escape"){
        SuppressGJShell = false
        GshMenu.focus() // should be previous focus
    }
  }
  var GJC_SessionId
  function GJC_SetSessionId(){
    var xd = new Date()
    GJC_SessionId = xd.getTime() / 1000
  }
  GJC_SetSessionId()
  function GJC_Memory(mem,args,text){
    argv = args.split(' ')
    cmd = argv[0]
    argv.shift()
    args = argv.join(' ')
    ret = ""

    if( cmd == 'clear' ){
        Permanent.setItem(mem,'')
    }else
    if( cmd == 'read' ){
        ret = Permanent.getItem(mem)
    }else
    if( cmd == 'save' ){
        val = Permanent.getItem(mem)
        if( val == null ){ val = "" }
        d = new Date()
        val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
        val += text.value
        Permanent.setItem(mem,val)
    }else
    if( cmd == 'write' ){
        val = Permanent.getItem(mem)
        if( val == null ){ val = "" }
        d = new Date()
        val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
        Permanent.setItem(mem,val)
    }else{
        ret = "Commands: write | read | save | clear"
    }
    return ret
  }
  // -- 2020-09-14 added TableEditor
  var GJE_CurElement = null; //GJE_RootNode
  GJE_NodeSaved = null
  GJE_TableNo = 1
  function GJE_StyleKeyCommand(kev){
    keycode = kev.code
    console.log('GJE-Key: '+keycode)
    if( keycode == 'Escape' ){
        GJE_SetStyle(this);
    }
    kev.stopPropagation()
    // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
  }
  var GJE_CommandMode = false
  function GJE_TableKeyCommand(kev,tab){
    wasCmdMode = GJE_CommandMode
    key = kev.code
    if( key == 'Escape' ){
        console.log("To command mode: "+tab.nodeName+"#"+tab.id)
        //tab.setAttribute('contenteditable','false')
        tab.style.caretColor = "blue"
        GJE_CommandMode = true
    }else
    if( key == "KeyA" ){
        tab.style.caretColor = "red"
        GJE_CommandMode = false
    }else
    if( key == "KeyI" ){
        tab.style.caretColor = "red"
        GJE_CommandMode = false
    }else
    if( key == "KeyO" ){
        tab.style.caretColor = "red"
        GJE_CommandMode = false
    }else
    if( key == "KeyJ" ){
        console.log("ROW-DOWN")
    }else
    if( key == "KeyK" ){
        console.log("ROW-UP")
    }else
    if( key == "Keyw" ){
        console.log("COL-FORW")
    }else
    if( key == "Keyb" ){
        console.log("COL-BACK")
    }

    kev.stopPropagation()
    if( wasCmdMode ){
        kev.preventDefault()
    }
  }
  function GJE_DragEvent(ev,elem){
    x = ev.clientX
    y = ev.clientY
    console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x+' y='+y)
  }
  // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
  // https://www.w3.org/TR/uievents/#events-mouseevents
  function GJE_DropEvent(ev,elem){
    x = ev.clientX
    y = ev.clientY
    this.style.x = x
    this.style.y = y
    this.style.position = 'absolute' // 'fixed'
    this.parentNode = gsh // just for test
    console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y='+y
        +' parent='+this.parentNode.id)
  }
  function GJE_SetTableStyle(ev){
    this.innerHTML = this.value; // sync. for external representation?
    if(false){
        stid = this.parentNode.id+this.id
            // and remove "_span" at the end
        e = document.getElementById(stid)
        //alert('SetTableStyle #'+e.id+'\n'+this.value)
        if( e != null ){
            e.innerHTML = this.value
        }else{
            console.log('Style Not found: '+stid)
        }
        //alert('event StopPropagetion: '+ev)
    }
  }
  function setCSSofClass(cclass,cstyle){
    const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
    rlen = ss.cssRules.length;
    let tabrule = null;
    rulex = -1

    // should skip white space at the top of cstyle
    sel = cstyle.charAt(0);
    selector = sel+cclass;
    console.log('-- search style rule for '+selector)

    for(let i = 0; i < rlen; i++){
        cr = ss.cssRules[i];
        console.log('CSS rule ['+i+'/'+rlen+'] '+cr.selectorText);
        if( cr.selectorText  === selector ){ // css class selector
            tabrule = ss.cssRules[i];
            console.log('CSS rule found for:['+i+'/'+rlen+'] '+selector);
            ss.deleteRule(i);
            //rlen = ss.cssRules.length;
            rulex = i
            // should search and replace the property here
        }
    }
    // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
    if( tabrule == null ){
        console.log('CSS rule NOT found for:['+rlen+'] '+selector);
        ss.insertRule(cstyle,rlen);
        ss.insertRule(cstyle,0); // override by 0?
        console.log('CSS rule inserted:['+(rlen+1)+']\n'+cstyle);
    }else{
        ss.insertRule(cstyle,rlen);
        ss.insertRule(cstyle,0);
        console.log('CSS rule replaced:['+(rlen+1)+']\n'+cstyle);
    }
  }
  function GJE_SetStyle(te){
    console.log('Apply the style to:'+te.id+'\n');
```

```
    console.log('Apply the style to:'+te.parentNode.id+'\n');
    console.log('Apply the style to:'+te.parentNode.class+'\n');
    cclass = te.parentNode.class;
    setCSSofClass(cclass,te.value); // should get selecter part from
                    // selector { rules }

    if(false){
    //console.log('Apply the style:')
    //stid = this.parentNode.id+this.id+"
    //stid = this.id+".style"
    css = te.value
    stid = te.parentNode.id+".style"
    e = document.getElementById(stid)
    if( e != null ){
        //console.log('Apply the style:'+e.id+'\n'+te.value);
        console.log('Apply the style:'+e.id+'\n'+css);
//      e.innerHTML = css; //te.value;
        //ncss = e.sheet;
        //ncss.insertRule(te.value,ncss.cssRules.length);
    }else{
        console.log('No element to Apply the style: '+stid)
    }
    tblid = te.parentNode.id+".table";
    e = document.getElementById(tblid);
    if( e != null ){
        //e.setAttribute('style',css);
        e.setProperty('style',css,'!impotant');
    }
    }
  }
  function makeTable(argv){
    //tid = ''
    cwe = GJE_CurElement
    tid = 'table_' + GJE_TableNo

    nt = new Text('\n')
    cwe.appendChild(nt)

    ne = document.createElement('span'); // the container
    cwe.appendChild(ne)
    ne.id = tid + '-span'
    ne.setAttribute('contenteditable',true)

    htspan = document.createElement('span'); // html part
    //htspan.id = tid + '-html'
    //ne.innerHTML = '\n'
    nt = new Text('\n')
    ne.appendChild(nt)
    ne.appendChild(htspan)

    htspan.id = tid
    htspan.setAttribute('class',tid)

    ne.setAttribute('draggable','true')
    ne.addEventListener('drag',GJE_DragEvent);
    ne.addEventListener('dragend',GJE_DropEvent);

    var col = 3
    var row = 2
    if( argv[0] != null ){
        col = argv[0]
        argv.shift()
    }
    if( argv[0] != null ){
        row = argv[0]
        argv.shift()
    }

    //ne.setAttribute('class',tid)
    ht = "\n"
    //ht += '<'+'table ' + 'id="'+tid+'"' + ' class="'+tid+'"'
    ht += '<'+'table '
        + ' onkeydown="GJE_TableKeyCommand(event,this)"'
        //+ ' ondrag="GJE_DragEvent(event,this)"\n'
        //+ ' ondragend="GJE_DropEvent(event,this)"\n'
        //+ ' draggable="true"\n'
        //+ ' contenteditable="true"'
        + '>\n'
    ht += '<'+'tbody>\n';
    for( r = 0; r < row; r++ ){
        ht += "<"+"tr>\n"
        for( c = 0; c < col; c++ ){
            ht += "<"+"td>"
            ht +=  "ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(c) + r
            ht += "<"+"/td>\n"
        }
        ht += "<"+"/tr>\n"
    }
    ht += '<'+'/tbody>\n';
    ht += '<'+'/table>\n';
    htspan.innerHTML = ht;
    nt = new Text('\n')
    ne.appendChild(nt)

    st = '#'+tid+' *{\n' // # for instanse specific
        +'  '+'border:1px solid #aaa;\n'
        +'  '+'background-color:#efe;\n'
        +'  '+'color:#222;\n'
        +'  '+'font-size:#14pt !important;\n'
        +'  '+'font-family:monospace,Courier New !important;\n'
        +'} /'+'* hit ESC to apply *'+'/\n'

    // wish script to be incldued
    //nj = document.createElement('script')
    //ne.appendChild(nj)
    //ne.innerHTML = 'function SetStyle(e){}'

    // selector seems lost in dynamic style appending
    if(false){
     ns = document.createElement('style')
     ne.appendChild(ns)
     ns.id = tid + '.style'
     ns.innerHTML = '\n'+st
     nt = new Text('\n')
     ne.appendChild(nt)
    }
    setCSSofClass(tid,st); // should be in JavaScript script?

    nx = document.createElement('textarea')
    ne.appendChild(nx)
    nx.id = tid + '-style_def'
    nx.setAttribute('class','GJ_StyleEditor')
    nx.spellcheck = false
    nx.cols = 60
    nx.rows = 10
    nx.innerHTML = '\n'+st
    nx.addEventListener('change',GJE_SetTableStyle);
    nx.addEventListener('keydown',GJE_StyleKeyCommand);
    //nx.addEventListener('click',GJE_SetTableStyle);

    nt = new Text('\n')
    cwe.appendChild(nt)

    GJE_TableNo += 1
    return 'created TABLE id="'+tid+'"'
  }
  function GJE_NodeEdit(argv){
    cwe = GJE_CurElement
    cmd = argv[0]
    argv.shift()
    args = argv.join(' ')
    ret = ""

    if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
        if( GJE_NodeSaved != null ){
            xn = GJE_RootNode
            GJE_RootNode = GJE_NodeSaved
            GJE_NodeSaved = xn
            ret = '-- did undo'
        }else{
            ret = '-- could not undo'
        }
        return ret
    }
    GJE_NodeSaved = GJE_RootNode.cloneNode()
    if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
        if( argv[0] == null ){
            ne = GJE_RootNode
        }else
        if( argv[0] == '..' ){
            ne = cwe.parentNode
        }else{
            ne = document.getElementById(argv[0])
        }
        if( ne != null ){
            GJE_CurElement = ne
            ret = "-- current node: " + ne.id
        }else{
            ret = "-- not found: " + argv[0]
        }
    }else
    if( cmd == '.mkt' || cmd == '.mktable' ){
        makeTable(argv)
    }else
    if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
        ne = document.createElement(argv[0])
        //ne.id = argv[0]
        ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
        cwe.appendChild(ne)
        if( cmd == '.m' || cmd == '.mk' ){
            GJE_CurElement = ne
        }
    }else
    if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
        cwe.id = argv[0]
    }else
    if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
    }else
    if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
        s = argv.join(' ')
        cwe.innerHTML = s
    }else
    if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
        cwe.setAttribute(argv[0],argv[1])
    }else
    if( cmd == '.l' ){
    }else
    if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
        ret = cwe.innerHTML
    }else
    if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
        ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
        for( we = cwe.parentNode; we != null; ){
            ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
            we = we.parentNode
        }
    }else
    {
        ret = "Command: mk | rm \n"
        ret += "   pw -- print current node\n"
        ret += "   mk type -- make node with name and type\n"
        ret += "   nm name -- set the id #name of current node\n"
        ret += "   rm name -- remove named node\n"
        ret += "   cd name -- change current node\n"
    }
    //alert(ret)
    return ret
  }
  function GJC_Command(text){
    lines = text.value.split('\n')
    line = lines[lines.length-1]
    argv = line.split(' ')
    text.value += '\n'
    if( argv[0] == '%' ){ argv.shift() }
    args0 = argv.join(' ')
    cmd = argv[0]
    argv.shift()
    args = argv.join(' ')

    if( cmd == 'nolog' ){
        StopConsoleLog = true
    }else
    if( cmd == 'new' ){
        if( argv[0] == 'table' ){
            argv.shift()
            console.log('argv='+argv)
            text.value += makeTable(argv)
        }else
        if( argv[0] == 'console' ){
            text.value += GJ_NewConsole('GJ_Console')
        }else{
            text.value += '-- new { console | table }'
        }
    }else
    if( cmd == 'strip' ){
        //text.value += GJF_StripClass()
    }else
    if( cmd == 'css' ){
```

```
  sel = '#table_1'
  if(argv[0]=='0')
  rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
  else
  rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}';
      document.styleSheets[3].deleteRule(0);
      document.styleSheets[3].insertRule(rule1,0);
      text.value += 'CSS rule added: '+rule1
  }else
  if( cmd == 'print' ){
      e = null;
      if( e == null ){
          e = document.getElementById('GJFactory_0')
      }
      if( e == null ){
          e = document.getElementById('GJFactory_1')
      }
      if( argv[0] != null ){
          id = argv[0]
          if( id  == 'f' ){
              //e = document.getElementById('GJE_RootNode');
          }else{
              e = document.getElementById(id)
          }
          if( e != null ){
              text.value += e.outerHTML
          }else{
              text.value += "Not found: " + id
          }
      }else{
          text.value += GJE_RootNode.outerHTML
          //text.value += e.innerHTML
      }
  }else
  if( cmd == 'destroy' ){
      text.value += GJFactory_Destroy()
  }else
  if( cmd == 'save' ){
      e = document.getElementById('GJFactory')
      Permanent.setItem('GJFactory-1',e.innerHTML)
      text.value += "-- Saved GJFactory"
  }else
  if( cmd == 'load' ){
      gjf = Permanent.getItem('GJFactory-1')
      e = document.getElementById('GJFactory')
      e.innerHTML = gjf
      // must restore EventListener
      text.value += "-- EventListener was not restored"
  }else
  if( cmd.charAt(0) == '.' ){
      argv0 = args0.split(' ')
      text.value += GJE_NodeEdit(argv0)
  }else
  if( cmd == 'cont' ){
      bannerIsStopping = false
      GshMenuStop.innerHTML = "Stop"
  }else
  if( cmd == 'date' ){
      text.value += DateLong()
  }else
  if( cmd == 'echo' ){
      text.value += args
  }else
  if( cmd == 'fork' ){
      html_fork()
  }else
  if( cmd == 'last' ){
      text.value += MyHistory
      //h = document.createElement("span")
      //h.innerHTML = MyHistory
      //text.value += h.innerHTML
      //tx = MyHistory.replace("\n","")
      //text.value += tx.replace("<"+"br>","\n") + "xxxx<"+"br>yyyy"
  }else
  if( cmd == 'ne' ){
      text.value += GJE_NodeEdit(argv)
  }else
  if( cmd == 'reload' ){
      location.reload()
  }else
  if( cmd == 'mem' ){
      text.value += GJC_Memory('GJC_Storage',args,text)
  }else
  if( cmd == 'stop' ){
      bannerIsStopping = true
      GshMenuStop.innerHTML = "Start"
  }else
  if( cmd == 'who' ){
      text.value += "SessionId="+GJC_SessionId+" "+document.URL
  }else
  if( cmd == 'wall' ){
      text.value += GJC_Memory('GJC_Wall','write',text)
  }else
  {
      text.value += "Commands: help | echo | date | last \n"
              + '          new  | save | load | mem  \n'
              + '          who  | wall | fork | nife'
  }
}

function GJC_Input(){
  if( this.value.endsWith("\n") ){ // remove NL added by textarea
      this.value = this.value.slice(0,this.value.length-1)
  }
}

var GCJ_Id = null
function GJC_Resize(){
  GJC_Id.style.zIndex = 20000
  GJC_Id.style.width = window.innerWidth - 16
  GJC_Id.style.height = 300
  GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
  GJC_Id.style.color = "rgba(255,255,255,1.0)"
}
function GJC_FocusIn(){
  this.spellcheck = false
  SuppressGJShell = true
  this.onkeydown = GJC_Keydown
  GJC_Resize()
}
function GJC_FocusOut(){
  SuppressGJShell = false
  this.removeEventListener('keydown',GJC_Keydown);
}
```

```
  window.addEventListener('resize',GJC_Resize);

  function GJC_OnStorage(e){
    //alert('Got Message')
    //GJC.value += "\n(((ReceivedMessage)))\n"
  }
  window.addEventListener('storage',GJC_OnStorage);
  //window.addEventListener('storage',()=>{alert('GotMessage')})

  function GJC_Setup(gjcId){
    gjcId.style.width = gsh.getBoundingClientRect().width
    gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
    //gjcId.value += "Date: " + DateLong() + "\n"
    gjcId.value += PS1
    gjcId.onfocus = GJC_FocusIn
    gjcId.addEventListener('input',GJC_Input);
    gjcId.addEventListener('focusout',GJC_FocusOut);
    GJC_Id = gjcId
  }
  function GJC_Clear(id){
  }
  if( document.getElementById("GJC_0") != null ){
    GJC_Setup(GJC_0)
  }else{
    document.write('<'+'textarea id="GJC_1" class="GJConsole"><'+'/textarea>')
    GJC_Setup(GJC_1)
    factory = document.createElement('span');
    gsh.appendChild(factory)
    GJE_RootNode = factory;
    GJE_CurElement = GJE_RootNode;
  }

  // TODO: focus handling
</script>
<style>
.GJ_StyleEditor {
  font-size:9pt !important;
  font-family:Courier New, monospace !important;
}
</style>

</details>
</span>
<!-- ----- GJConsole END } ----- -->
*/

/*
<span id="BlinderText">
<style id="BlinderTextStyle">
 #GJLinkView {
    xxposition:absolute; z-index:5000;
    position:relative;
    display:block;
    left:8px;
    color:#fff;
    width:800px; height:300px; resize:both;
    margin:0px; padding:4px;
    background-color:rgba(200,200,200,0.5) !important;
 }
 .MssgText {
    width:578px !important;
    resize:both !important;
    color:#000 !important;
 }
 .GjNote {
    font-family:Georgia !important;
    font-size:13pt !important;
    color:#22a !important;
 }
 .textField  {
    display:inline;
    border:0.5px solid #444;
    border-radius:3px;
    color:#000; background-color:#fff;
    width:106pt; height:18pt;
    margin:2px;
    padding:2px;
    resize:none;
    vertical-align:middle;
    font-size:10pt; font-family:Courier New;
 }
 .textLabel {
    border:0px solid #000 !important;
    background-color:rgba(0,0,0,0);
 }
 .textURL {
    width:300pt !important;
    border:0px solid #000 !important;
    background-color:rgba(0,0,0,0);
}
 .VisibleText {
 }
 .BlinderText {
    color:#000; background-color:#eee;
 }
 .joinButton {
    font-family:Georgia !important;
    font-size:11pt;
    line-height:1.1;
    height:18pt;
    width:50pt;
    padding:3px !important;
    text-align:center !important;
    border-color:#aaa !important;
    border-radius:5px;
    color:#fff; background-color:#4a4 !important;
    vertical-align:middle !important;
 }
 .SendButton {
    vertical-align:top;
 }
 .ws0_log {
    font-size:10pt;
    color:#000 !important;
    line-height:1.0;
    background-color:rgba(255,255,255,0.7) !important;
    font-family:Courier New,monospace !important;
    width:99.3%;
    white-space:pre;
 }
</style>

<!-- Form autofill test
Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafrm/formLogin" size="80">
```

```
<form method="POST" id="xxform" action="https://192.168.10.1/boafrm/formLogin">
dest?     <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
-->
<details><summary>Form Auto. Filling</summary>
<style>
 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
    display:inline !important; font-size:10pt !important; padding:1px !important;
 }
</style>
<span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
 SessionId:<input id="xxssid" class="xxinput" name="SESSION_ID" type="text" size="80">
           <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
</span>
<script>
 function XXSetFormAction(){
    xxform.setAttribute('action',xxserv.value);
}
xxform.setAttribute('action',xxserv.value);
xxserv.addEventListener('change',XXSetFormAction);
//xxserv.value = location.href;
</script>
</details>
*/

/*
<details id="BlinderTextClass" class="gsh-src"><summary>BlinderText</summary>
<span id="BlinderTextScript">
// https://w3c.github.io/uievents/#event-type-keydown
//
// 2020-09-21 class BlinderText - textarea element not to be readable
//
// BlinderText attributes
//  bl_plainText - null
//  bl_hideChecksum - [false]
//  bl_showLength - [false]
//  bl_visible - [false]
//  data-bl_config - []
//   - min. length
//   - max. length
//   - acceptable charset in generete text
//
function BlinderChecksum(text){
    plain = text.bl_plainText;
    return strCRC32(plain,plain.length).toFixed(0);
}
function BlinderKeydown(ev){
    pass = ev.target
    if( ev.code == 'Enter' ){
        ev.preventDefault();
    }
    ev.stopPropagation()
}
function BlinderKeyup1(ev){
    blind = ev.target
    if( ev.code == 'Backspace'){
        blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
    }else
    if( and(ev.code == 'KeyV', ev.ctrlKey) ){
        blind.bl_visible = !blind.bl_visible;
    }else
    if( and(ev.code == 'KeyL', ev.ctrlKey) ){
        blind.bl_showLength = !blind.bl_showLength;
    }else
    if( and(ev.code == 'KeyU', ev.ctrlKey) ){
        blind.bl_plainText = "";
    }else
    if( and(ev.code == 'KeyR', ev.ctrlKey) ){
        checksum = BlinderChecksum(blind);
        blind.bl_plainText = checksum; //.toString(32);
    }else
    if( ev.code == 'Enter' ){
        ev.stopPropagation();
        ev.preventDefault();
        return;
    }else
    if( ev.key.length != 1 ){
        console.log('KeyUp: '+ev.code+'/'+ev.key);
        return;
    }else{
        blind.bl_plainText += ev.key;
    }

    leng = blind.bl_plainText.length;
    //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
    checksum = BlinderChecksum(blind) % 10; // show last one digit only

    visual = '';
    if( !blind.bl_hideCheckSum || blind.bl_showLength ){
        visual += '[';
    }
    if( !blind.bl_hideCheckSum ){
        visual += '#'+checksum.toString(10);
    }
    if( blind.bl_showLength ){
        visual += '/' + leng;
    }
    if( !blind.bl_hideCheckSum || blind.bl_showLength ){
        visual += '] ';
    }
    if( blind.bl_visible ){
        visual += blind.bl_plainText;
    }else{
        visual += '*'.repeat(leng);
    }
    blind.value = visual;
}
function BlinderKeyup(ev){
    BlinderKeyup1(ev);
    ev.stopPropagation();
}
// https://w3c.github.io/uievents/#keyboardevent
// https://w3c.github.io/uievents/#uievent
// https://dom.spec.whatwg.org/#event
function BlinderTextEvent(){
    ev = event;
    blind = ev.target;
    console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
    if( ev.type == 'keyup' ){
        BlinderKeyup(ev);
    }else
    if( ev.type == 'keydown' ){
        BlinderKeydown(ev);
    }else{
        console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
    }
}
//< textarea hidden id="BlinderTextClassDef" class="textField""
// onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
// spellcheck="false">< /textarea>
//< textarea hidden id="gj_pass1"
//  class="textField BlinderText"
//  placeholder="PassWord1"
//  onkeydown="BlinderTextEvent()"
//  onkeyup="BlinderTextEvent()"
//  spellcheck="false"< /textarea>
function SetupBlinderText(parent,txa,phold){
    if( txa == null ){
        txa = document.createElement('textarea');
        //txa.id = id;
    }
    txa.setAttribute('class','textField BlinderText');
    txa.setAttribute('placeholder',phold);
    txa.setAttribute('onkeydown','BlinderTextEvent()');
    txa.setAttribute('onkeyup','BlinderTextEvent()');
    txa.setAttribute('spellcheck','false');
    //txa.setAttribute('bl_plainText','false');
    txa.bl_plainText = '';
    //parent.appendChild(txa);
}
function DestroyBlinderText(txa){
    txa.removeAttribute('class');
    txa.removeAttribute('placeholder');
    txa.removeAttribute('onkeydown');
    txa.removeAttribute('onkeyup');
    txa.removeAttribute('spellcheck');
    txa.bl_plainText = '';
}
//
// visible textarea like Username
//
function VisibleTextEvent(){
    if( event.code == 'Enter' ){
        if( event.target.NoEnter ){
            event.preventDefault();
        }
    }
    event.stopPropagation();
}
function SetupVisibleText(parent,txa,phold){
    if( false ){
        txa.setAttribute('class','textField VisibleText');
    }else{
        newclass = txa.getAttribute('class');
        if( and(newclass != null, newclass != '') ){
            newclass += ' ';
        }
        newclass += 'VisibleText';
        txa.setAttribute('class',newclass);
    }
    //console.log('SetupVisibleText class='+txa.class);
    txa.setAttribute('placeholder',phold);
    txa.setAttribute('onkeydown','VisibleTextEvent()');
    txa.setAttribute('onkeyup',  'VisibleTextEvent()');
    txa.setAttribute('spellcheck','false');
    cols = txa.getAttribute('cols');
    if( cols != null ){
        txa.style.width = '580px';
        //console.log('VisualText#'+txa.id+' cols='+cols)
    }else{
        //console.log('VisualText#'+txa.id+' NO cols')
    }
    rows = txa.getAttribute('rows');
    if( rows != null ){
        txa.style.height = '30px';
        txa.style.resize = 'both';
        txa.NoEnter = false;
    }else{
        txa.NoEnter = true;
    }
}
function DestroyVisibleText(txa){
    txa.removeAttribute('class');
    txa.removeAttribute('placeholder');
    txa.removeAttribute('onkeydown');
    txa.removeAttribute('onkeyup');
    txa.removeAttribute('spellcheck');
    cols = txa.removeAttribute('cols');
}
</span>
<script>
js = document.getElementById('BlinderTextScript');
eval(js.innerHTML);
//js.outerHTML = ""
</script>

</details>
</span>
*/

/*
<script id="GJLinkScript">
function gjkey_hash(text){
    return strCRC32(text,text.length) % 0x10000;
}
function gj_addlog(e,msg){
    now = (new Date().getTime() / 1000).toFixed(3);
    tstp = '['+now+'] '
    e.value += tstp + msg;
    e.scrollTop = e.scrollHeight;
}
function gj_addlog_cl(msg){
    ws0_log.value += '(console.log) ' + msg + '\n';
}
var GJ_Channel = null;
var GJ_Log = null;
var gjx; // the global variable
function GJ_Join(){
    target = gj_join;
    if( target.value == 'Leave' ){
        GJ_Channel.close();
        GJ_Channel = null;
        target.value = 'Join';
        return;
    }

    var ws0;
    var ws0_log;

    sav_console_log = console.error
    console.error = gj_addlog_cl
    ws0 = new WebSocket(gj_serv.innerHTML);
    console.error = sav_console_log

    GJ_Channel = ws0;
    ws0_log = document.getElementById('ws0_log');
    GJ_Log = ws0_log;

    now = (new Date().getTime() / 1000).toFixed(3);
    const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
    cst = wsstats[ws0.readyState];
    gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');

    ws0.addEventListener('error', function(event){
        gj_addlog(ws0_log,'stat error : transport error?\n');
    });
    ws0.addEventListener('open', function(event){
        GJLinkView.style.zIndex = 10000;
        //console.log('#'+GJLinkView.id+'.zIndex='+GJLinkView.style.zIndex);
        date1 = new Date().getTime();
        date2 = (date1 / 1000).toFixed(3);
        seed = date1.toString(16);

        // user name and key
        user = document.getElementById('gj_user').value;
        if( user.length == 0 ){
            gj_user.value = 'nemo';
            user = 'nemo';
        }
        key1 = document.getElementById('gj_ukey').bl_plainText;
        ukey = gjkey_hash(seed+user+key1).toString(16);

        // session name and key
        chan = document.getElementById('gj_chan').value;
        if( chan.length == 0 ){
            gj_chan.value = 'main';
            chan = 'main';
        }
        key2 = document.getElementById('gj_ckey').bl_plainText;
        ckey = gjkey_hash(seed+chan+key2).toString(16);

        msg = date2 +' JOIN ' + user + '|' + chan + ' ' + ukey + ':' + ckey;
        gj_addlog(ws0_log,'send '+msg+'\n');
        ws0.send(msg);

        target.value = 'Leave';
        //console.log('['+date2+'] #'+target.id+' '+target.value+'\n');
        //gj_addlog(ws0_log,'label '+target.value+'\n');
    });
    ws0.addEventListener('message', function(event){
        now = (new Date().getTime() / 1000).toFixed(3);
        msg = event.data;
        gj_addlog(ws0_log,'recv '+msg+'\n');

        argv = msg.split(' ')
        tstamp = argv[0];
        argv.shift();
        if( argv[0] == 'reload' ){
            location.reload()
        }
        argv.shift(); // command
        argv.shift(); // from|to
        if( argv[0] == 'auth' ){
            // doing authorization required
        }
        if( argv[0] == 'echo' ){
            now = (new Date().getTime() / 1000).toFixed(3);
            msg = now+' '+'RESP '+argv.join(' ');
            gj_addlog(ws0_log,'send '+msg+'\n');
            ws0.send(msg);
        }
        if( argv[0] == 'eval' ){
            argv.shift();
            js = argv.join(' ');
            ret = eval(js); // <------------ eval()
            gj_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
            now = (new Date().getTime() / 1000).toFixed(3);
            msg = now + ' ' + 'RESP ' + ret;
            ws0.send(msg);
            gj_addlog(ws0_log,'send '+msg+'\n')
        }
    });
    ws0.addEventListener('close', function(event){
        if( GJ_Channel == null ){
            gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
            return;
        }
        GJ_Channel.close();
        GJ_Channel = null;
        target.value = 'Join';
        gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
    });
}
function GJ_SendMessageUserPass(user,chan,msgbody){
    now = (new Date().getTime() / 1000).toFixed(3);
    msg = now +' ISAY '+user+'|'+chan+' '+ msgbody;
    gj_addlog(GJ_Log,'send '+msg+'\n');
    GJ_Channel.send(msg);
}
function GJ_SendMessage(msgbody){
    if( GJ_Channel == null ){
        gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
        return;
    }
    //target = event.target;
    user = document.getElementById('gj_user').value;
    chan = document.getElementById('gj_chan').value;
    GJ_SendMessageUserPass(user,chan,msgbody);
}
function GJ_Send(){
    msgbody = gj_sendText.value;
    GJ_SendMessage(msgbody);
}
</script>

<!-- ------------------------- GJLINK ------------------ -->
<!--
    - User can subscribe to a channel
    - A channel will be broadcasted
    - A channel can be a pattern (regular expression)
    - User is like From:(me) and channel is like To: or Recipient:
    - like VIABUS
        - watch message with SENDME, WATCH, CATCH, HEAR, or so
        - routing with path expression or name pattern (with routing with DNS like system)
-->
*/

//<span id="GJLinkGolang">
// <details id="GshWebSocket" class="gsh-src"><summary>Golang / JavaScript Link</summary>
// 2020-0920 created
// <a href="https://pkg.go.dev/golang.org/x/net/websocket">WS</a>
// <a href="https://godoc.org/golang.org/x/net/websocket">WS</a>
// INSTALL: go get golang.org/x/net/websocket
// INSTALL: sudo {apt,yum} install git (if git is not instlled yet)
// import "golang.org/x/net/websocket"
const gshws_origin = "http://locahost:9999"
const gshws_server = "localhost:9999"
const gshws_port = 9999
const gshws_path = "gjlink1"
const gshws_url = "ws://"+gshws_server+"/"+gshws_path
const GSHWS_MSGSIZE = (8*1024)
func fmtstring(fmts string, params ...interface{})(string){
    return fmt.Sprintf(fmts,params...)
}
func GSHWS_MARK(what string)(string){
    now := time.Now()
    us := fmtstring("%06d",now.Nanosecond() / 1000)
    mark := ""
    if( !AtConsoleLineTop ){
        mark += "\n"
        AtConsoleLineTop = true
    }
    mark += "["+now.Format(time.Stamp)+"."+us+"] -GJ-" + what + ": "
    return mark
}
func gchk(what string,err error){
    if( err != nil ){
        panic(GSHWS_MARK(what)+err.Error())
    }
}
func glog(what string, fmts string, params ...interface{}){
    fmt.Print(GSHWS_MARK(what))
    fmt.Printf(fmts+"\n",params...)
}

var WSV = []*websocket.Conn{}
func jsend(argv []string){
    if len(argv) <= 1 {
        fmt.Printf("--Ij %v [-m] command arguments\n",argv[0])
        return
    }
    argv = argv[1:]
    if( len(WSV) == 0 ){
        fmt.Printf("--Ej-- No link now\n")
        return
    }
    if( 1 < len(WSV) ){
        fmt.Printf("--Ij-- multiple links (%v)\n",len(WSV))
    }

    multicast := false // should be filtered with regexp
    if( 0 < len(argv) && argv[0] == "-m" ){
        multicast = true
        argv = argv[1:]
    }
    args := strings.Join(argv," ")

        now := time.Now()
        msec := now.UnixNano() / 1000000;
        tstamp := fmtstring("%.3f",float64(msec)/1000.0)
        msg := fmtstring("%v SEND gshell|* %v",tstamp,args)

    if( multicast ){
        for i,ws := range WSV {
            wn,werr := ws.Write([]byte(msg))
            if( werr != nil ){
                fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
            }
            glog("SQ",fmtstring("(%v) %v",wn,msg))
        }
    }else{
        i := 0
            ws := WSV[i]
            wn,werr := ws.Write([]byte(msg))
            if( werr != nil ){
                fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
            }
            glog("SQ",fmtstring("(%v) %v",wn,msg))
    }
}
func serv1(ws *websocket.Conn) {
    WSV = append(WSV,ws)
    //fmt.Print("\n")
    glog("CO","accepted connections[%v]",len(WSV))
    //remoteAddr := ws.RemoteAddr
    //fmt.Printf("-- accepted %v\n",remoteAddr)
    //fmt.Printf("-- accepted %v\n",ws.Config())
    //fmt.Printf("-- accepted %v\n",ws.Config().Header)
    //fmt.Printf("-- accepted %v // %v\n",ws,serv1)

    var reqb = make([]byte,GSHWS_MSGSIZE)
    for {
        rn, rerr := ws.Read(reqb)
        if( rerr != nil || rn < 0 ){
            glog("SQ",fmtstring("(%v,%v)",rn,rerr))
            break
        }
        req := string(reqb[0:rn])
        glog("SQ",fmtstring("(%v) %v",rn,req))

        margv := strings.Split(req," ");
        margv = margv[1:]
        if( 0 < len(margv) ){
            if( margv[0] == "RESP" ){
                // should forward to the destination
                continue;
            }
        }
        now := time.Now()
        msec := now.UnixNano() / 1000000;
        tstamp := fmtstring("%.3f",float64(msec)/1000.0)
        res := fmtstring("%v "+"CAST"+" %v",tstamp,req)
        wn, werr := ws.Write([]byte(res))
        gchk("SE",werr)
        glog("SR",fmtstring("(%v) %v",wn,string(res)))
    }
    glog("SF","WS response finish")

    wsv := []*websocket.Conn{}
    wsx := 0
    for i,v := range WSV {
        if( v != ws ){
            wsx = i
            wsv = append(wsv,v)
        }
    }
    WSV = wsv
    //glog("CO","closed %v",ws)
    glog("CO","closed connection [%v/%v]",wsx+1,len(WSV)+1)
    ws.Close()
}
// url ::= [scheme://]host[:port][/path]
func decomp_URL(url string){
}
func full_wsURL(){
}
func gj_server(argv []string) {
    gjserv := gshws_url
    gjport := gshws_server
    gjpath := gshws_path
    gjscheme := "ws"

    //cmd := argv[0]
    argv = argv[1:]
    if( 1 <= len(argv) ){
        serv := argv[0]
        if( 0 < strings.Index(serv,"://") ){
            schemev := strings.Split(serv,"://")
            gjscheme = schemev[0]
            serv = schemev[1]
        }
        if( 0 < strings.Index(serv,"/") ){
            pathv := strings.Split(serv,"/")
            serv = pathv[0]
            gjpath = pathv[1]
        }
        servv := strings.Split(serv,":")
        host := "localhost"
        port := 9999
        if( servv[0] != "" ){
            host = servv[0]
        }
        if( len(servv) == 2 ){
            fmt.Sscanf(servv[1],"%d",&port)
        }
        //glog("LC","hostport=%v (%v : %v)",servv,host,port)
        gjport = fmt.Sprintf("%v:%v",host,port)
        gjserv = gjscheme + "://" + gjport + "/" + gjpath
    }
    glog("LS",fmtstring("listening at %v",gjserv))
    http.Handle("/"+gjpath,websocket.Handler(serv1))
    err := error(nil)
    if( gjscheme == "wss" ){
        // https://golang.org/pkg/net/http/#ListenAndServeTLS
        //err = http.ListenAndServeTLS(gjport,nil)
    }else{
        err = http.ListenAndServe(gjport,nil)
    }
    gchk("LE",err)
}

func gj_client(argv []string) {
    glog("CS",fmtstring("connecting to %v",gshws_url))
    ws, err := websocket.Dial(gshws_url,"",gshws_origin)
    gchk("C",err)

    var resb = make([]byte, GSHWS_MSGSIZE)
    for qi := 0; qi < 3; qi++ {
        req := fmtstring("Hello, GShell! (%v)",qi)
        wn, werr := ws.Write([]byte(req))
        glog("QM",fmtstring("(%v) %v",wn,req))
        gchk("QE",werr)
        rn, rerr := ws.Read(resb)
        gchk("RE",rerr)
        glog("RM",fmtstring("(%v) %v",rn,string(resb)))
    }
    glog("CF","WS request finish")
}
//</details></span>

/*
<details><summary>GJ Link</summary>
<span id="GJLinkView" class="GJLinkView">
 <p>
 <note class="GjNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
 </p>
 <p>
 <span id="GJLink_1">
 <script id="gj_xxx1_gen">
 if( document.getElementById('gj_serv') == null ){ // executed twice??
  document.write('<'+'span id="gj_serv_label" class="textField textLabel">Server: <'+'/span>');
  document.write('<'+'span id="gj_serv" class="textField textURL" contenteditable><'+'/span>');
 }
 </script>
 <br>
 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
 <script id="gj_xxx2_gen">
 if( true ){
  document.write('<'+'textarea id="gj_user" class="textField"><'+'/textarea>');
  document.write('<'+'textarea id="gj_ukey" class="textField"><'+'/textarea>');
  document.write('<'+'textarea id="gj_chan" class="textField"><'+'/textarea>');
  document.write('<'+'textarea id="gj_ckey" class="textField"><'+'/textarea>');
 }
 </script>
 <br>
 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
 <script id="gj_sendText_gen">
 if( true ){
  document.write('<'+'textarea id="gj_sendText" class="textField MssgText" cols=60 rows=2><'+'/textarea>');
 }
 </script>
 </span></p>
 <p>
 <script id="ws0_log_gen">
 if( true ){
  document.write('<'+'textarea id="ws0_log" class="ws0_log"'
    +' cols=100 rows=10 spellcheck="false"><'+'/textarea>');
 }
 </script>
 </p>
</span>
<script>
function SetupGJLink(){
    SetupVisibleText(GJLink_1,gj_serv,'GJLinkSv');
    SetupVisibleText(GJLink_1,gj_user,'UserName');
    SetupBlinderText(GJLink_1,gj_ukey,'UserKey');
    SetupVisibleText(GJLink_1,gj_chan,'ChannelName');
    SetupBlinderText(GJLink_1,gj_ckey,'ChannelKey');
    SetupVisibleText(GJLink_1,gj_sendText,'Message');
    gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
}
SetupGJLink();
function iselem(eid){
    return document.getElementById(eid);
}
function DestroyGJLink1(){
    if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
    if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
    if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
    if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
    if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
    if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
    if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
    if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
}
DestroyGJLink = DestroyGJLink1;
</script>
</details>
*/

/*
<script id="HtmlCodeview-script">
  function showHtmlCode(otxa,code){
    if( event.target.value == 'ShowCode' ){
        txa = document.createElement('textarea');
        txa.id = otxa.id;
        txa.setAttribute('class','HtmlCodeviewText');
        otxa.parentNode.replaceChild(txa,otxa);
        txa.setAttribute('spellcheck','false');
        txa.value = code.innerHTML;
        txa.style.display = "block";
        txa.style.width = "100%";
        txa.style.height = "300px";
        event.target.value = 'HideCode';
    }else{
        otxa.style.display = "none";
        event.target.value = 'ShowCode';
    }
  }
</script>
<style id="HtmlCodeview-style">
  .HtmlCodeviewText {
    font-size:10pt;
    font-family:Courier New;
    white-space:pre;
  }
  .HtmlCodeviewButton {
    font-size:11pt;
    line-height:1.2;
    font-family:Georgia;
    border-radius:3px;
    color:#ddd; background-color:#333;
  }
</style>
*/

/*
<details><summary>Live HTML Snapshot</summary>
<span id="LiveHTML">
<!-- ---------- HTML SnapShot: Edit, save and load // 2020-0924 SatoxITS { -->
<input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="showLiveHTMLcode()">
<span id="LiveHTML_Codeview"></span>
<script id="LiveHtmlScript">
function showLiveHTMLcode(){
    showHtmlCode(LiveHTML_Codeview,LiveHTML);
}
var _editable = false;
var savSuppressGJShell = false;
function ToggleEditMode(){
    _editable = !_editable;
    if( _editable ){
        savSuppressGJShell = SuppressGJShell;
        SuppressGJShell = true;
        gsh.setAttribute('contenteditable','true');
        GshMenuEdit.innerHTML = 'Lock';
        GshMenuEdit.style.color = 'rgba(255,0,0,1)';
        GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
    }else{
        SuppressGJShell = savSuppressGJShell;
        gsh.setAttribute('contenteditable','false');
        GshMenuEdit.innerHTML = 'Edit';
        GshMenuEdit.style.color = 'rgba(16,160,16,1)';
        GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
    }
}
function html_edit(){
    ToggleEditMode();
}

// Live HTML (DOM) Snapshot onto browser's localStorage
// 2020-0923 SatoxITS
var htRoot = gsh // -- Element-ID, should be selectable
const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
                   // -- should be a [map] of URL
                   // -- should be with CSSOM as inline script
const htVersionTag = 'VersionTag'; // VesionTag Eelment-ID in the HTML (in DOM)
function showVersion(note,w,v,u,t){
    w.alert(note+': ' + v + '\n'
        + '-- URL:  ' + u + '\n'
        + '-- Time: ' + DateLong0(t*1000)
    );
}
function html_save(){
    u = document.URL;
    t = new Date().getTime() / 1000;
    v = '<'+'span id="'+htVersionTag+'" data-url="'+u+'" data-time="'+t+'">';
    v += '<'+'/span>\n';
    h += v + htRoot.outerHTML;
    localStorage.setItem(snappedHTML,h);
    showVersion("Saved",window,v,u,t);
}
```

```
function html_load(){
    h = localStorage.getItem(snappedHTML);
    if( h == null ){
        alert('No snapshot taken yet');
        return;
    }
    w = window.open('','','');
    d = w.document;
    d.write(h);
    w.focus();
    html_ver1("Loaded",w,d);
}
function html_ver1(note,w,d){
    if( (v = d.getElementById(htVersionTag)) != null ){
        h = v.outerHTML;
        u = v.getAttribute('data-url');
        t = v.getAttribute('data-time');
    }else{
        h = 'No version info. in the page';
        u = '';
        t = 0;
    }
    showVersion(note,w,v,u,t);
}
function html_ver0(){
    html_ver1("Version",window,document);
}
</script>
<!-- LiveHTML } -->
</span>
</details>
*/

/*
<details><summary>Event sharing</summary>
<span id="EventSharingCodeSpan">

<!-- ---------- Event sharing // 2020-0925 SatoxITS { -->

<div id="iftestTemplate" class="iftest" hidden="">
<style> .iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;} </style>
<span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
  function docadd(txt){
    document.body.append(txt);
    window.scrollTo(0,100000);
  }
  function frameClick(){
    xy = '(x='+event.x + ' y='+event.y+')';
    //docadd('Got Click on #'+event.target.id+' '+xy+ '\n');
    docadd('Got Click on #'+Fid.value+', '+xy+ '\n');
    window.scrollTo(0,100000);
    window.parent.postMessage('OnClick: '+xy,'*');
  }
  function frameMousemove(){
    if( false ){
    document.body.append('Mousemove on #'+event.target.id+' '
        + 'x='+event.x + ' y='+event.y + '\n');
    peerWin = window.frames.iframe1;
    document.body.append('Send to peer #'+peerWin+' ' + '\n');
    window.scrollTo(0,100000);
    peerWin.postMessage('Hi!','*');
    }
  }
  function frameKeydown(){
    msg = 'Got Keydown: #'+Fid.value+', ('+event.code+')';
    docadd(msg+'\n');
    window.parent.postMessage(msg,'*');
  }
  function frameOnMessage(){
    docadd('Message ' + event.data + '\n');
    window.scrollTo(0,100000);
  }
  if( document.getElementById('Fid') ){
    frameBody.id = Fid.value;
    h = '';
    h += '<'+'style>*{';
    h += 'font-size:10pt;white-space:pre-wrap;';
    h += 'font-family:Courier New;';
    h += '}<'+'/style>';
    h += 'I am '+Fid.value+'\n';
    document.write(h);
    window.addEventListener('click',frameClick);
    window.addEventListener('keydown',frameKeydown);
    window.addEventListener('message',frameOnMessage);
    window.addEventListener('mousemove',frameMousemove);
    window.parent.postMessage('Hi parent, I am '+Fid.value,'*');
  }
</script></span></div>

<style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
<h2>Inter-window communicaiton</h2>
<note>
frame0 >>> frame1 and frame2<br>
frame1 >>> frame0 and frame2<br>
frame2 >>> frame0 and frame1<br>
</note>
<div id="iframe-test">
<pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
<iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
<iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
<iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
</div>

<script id="if0-test-script">
  setupFrames0();
  setupFrames12();

  function setFrameSrcdoc(dst,src){
    if( true ){
        dst.contentWindow.document.write(src);
        // this makes browser waite close, and crash if accumulated !?
        // so it should be closed after write
        dst.contentWindow.document.close();
    }else{
        // to be erased before source dump
        // but shold be set for live snapshot
        dst.srcdoc = src;
    }
  }
  function setupFrames0(){
    ibody = iframe0.contentWindow.document.body;
    iframe0.style.width = "755px"
    //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
    window.addEventListener('message',messageFromChild);

    if0 = '';
    if0 += '<'+'pre style="font-family:Courier New;">';
    if0 += '<input id="Fid" value="iframe0">';
    if0 += iftestTemplate.innerHTML;
    setFrameSrcdoc(iframe0,if0);

    function clickOnChild(){
        console.log('clickOn #'+this.id);
    }
    function moveOnChild(){
        console.log('moveOn #'+this.id);
    }
    iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
    iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
  }
  function setupFrames12(){
    if1 = '<input id="Fid" value="iframe1">';
    if1 += iftestTemplate.innerHTML;
    setFrameSrcdoc(iframe1,if1);
    //iframe1.name = 'iframe1'; // this seems break contentWindow

    if2 = '<input id="Fid" value="iframe2">';
    if2 += iftestTemplate.innerHTML;
    setFrameSrcdoc(iframe2,if2);

    iframe1.addEventListener('message',messageFromChild);
        //iframe1.addEventListener('mouseover',moveOnChild);
    iframe2.addEventListener('message',messageFromChild);
        //iframe2.addEventListener('mouseover',moveOnChild);
    iframe1.contentWindow.postMessage('[parent0] Hi iframe1 -- from parent.','*');
    //iframe1.contentWindow.postMessage('Your peer is '+iframe2.contentWindow,'*');
    iframe2.contentWindow.postMessage('[parent0] Hi iframe2 -- from parent.','*');
    //iframe2.contentWindow.postMessage('Your peer is '+iframe1.contentWindow,'*');
  }
  function messageFromChild(){
    from = null;
    forw = null;
    if( event.source == iframe0.contentWindow ){
        from = '[iframe0] '
        forw = 'iframe12';
    }else
    if( event.source == iframe1.contentWindow ){
        from = '[iframe1] '
        forw = 'iframe2';
    }else
    if( event.source == iframe2.contentWindow ){
        from = '[iframe2] '
        forw = 'iframe1';
    }else
    {
        iframeHost.innerHTML += 'Message [unknown] '
        + ' orig=' + event.origin
        + ' data=' + event.data
        //+ ' from=' + event.source
        ;
    }
    msglog1 = from + event.data + ' -- '
        + ' from=' + event.source
        + ' orig=' + event.origin
        + ' name=' + event.source.name
        //+ ' port=' + event.ports
        //+ ' evid=' + event.lastEventId
        + '\n'
        ;
    if( true ){
        if( forw == 'iframe1' || forw == 'iframe12' ){
        iframe1.contentWindow.postMessage(from+event.data);
        }
        if( forw == 'iframe2' || forw == 'iframe12' ){
        iframe2.contentWindow.postMessage(from+event.data);
        }
    }
    txtadd0(msglog1);

    function txtadd0(txt){
        iframe0.contentWindow.document.body.append(txt);
        iframe0.contentWindow.scrollTo(0,100000);
    }
  }
  function es_ShowSelf(){
    iframe1.setAttribute('src',document.URL);
    iframe2.setAttribute('src',document.URL);
  }
</script>

<input class="HtmlCodeviewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
<input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
<span id="EventSharingCodeview"></span>
<script id="EventShareingScript">
function es_showHtmlCode(){
    showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
}
DestroyEventSharingCodeview = function(){
    //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
    EventSharingCodeview.innerHTML = "";
    iframe0.style = "";
    //iframe0.srcdoc = "erased";
    //iframe1.srcdoc = "erased";
    //iframe2.srcdoc = "erased";
}
</script>
<!-- EventSharing } -->
</span>
</details>
*/

/*
<!-- ---------- "GShell Inside" Nofitifaction { -->
<script id="script-gshell-inside">
var notices = 0;
function noticeGShellInside(){
    ver = '';
    if( ver = document.getElementById('GshVersion') ){
        ver = ver.innerHTML;
    }
    console.log('GJShell Inside (^-^)//'+ver);
    notices += 1;
    if( 2 <= notices ){
        document.removeEventListener('mousemove',noticeGShellInside);
    }
}
document.addEventListener('mousemove',noticeGShellInside);
noticeGShellInside();
```

```

const FooterName = 'GshFooter'
function DestroyFooter(){
    if( (footer = document.getElementById(FooterName)) != null ){
        //footer.parentNode.removeChild(footer);
        empty = document.createElement('div');
        empty.id = 'GshFooter0';
        footer.parentNode.replaceChild(empty,footer);
    }
}

footer = document.createElement('div');
footer.id = FooterName;
footer.style.backgroundImage = "url("+ITSmoreQR+")";
//GshFooter0.parentNode.appendChild(footer);
GshFooter0.parentNode.replaceChild(footer,GshFooter0);
</script>
<!-- } -->

<!--
    border:20px inset #888;
-->

//<span id="WirtualDesktopCodeSpan">
/*
<details><summary>Wirtual Desktop</summary>
<!-- ---------- Web Virtual Desktop // 2020-0927 SatoxITS { -->
<style>
.WirtualSpace {
    z-index:0;
    xwidth:1280px !important; xheight:720px !important;
    width:5120px; height:2880px;
    border-width:0px;
    xxbackground-color:rgba(32,32,160,0.8);
    xxbackground-image:url("WD-WallPaper03.png");
    xxbackground-size:100% 100%;
    color:#22a;xfont-family:Georgia;font-size:10pt;
    xxoverflow:scroll;
}
.WirtualGrid {
    z-index:0;
    position:absolute;
    width:800px; height:500px;
    border:1px inset #fff;
    color:rgba(192,255,192,0.8);
    font-family:Georgia, Courier New;
    text-align:right;
    vertical-align:middle;
    font-size:200px;
    text-shadow:4px 4px #ccf;
}
.WD_GridScroll {
    z-index:100000;
    background-color:rgba(200,200,200,0.1);
}
.WirtualDesktop {
    z-index:0;
    position:relative;
    resize:both !important;
    overflow:scroll;
    display:block;
    min-width:120px !important; min-height:60px !important;
    width:800px;
    height:500px;
    border:10px inset #228;
    border-width:30px; border-radius:20px;
    background-image:url("WD-WallPaper03.png");
    background-size:100% 100%;
    color:#22a;font-family:Georgia;font-size:10pt;
}
.comment {
    // overflow=scroll seems to bound childrens' view in the element span
    // specifying overflow seems fix the position of the element
}
.WirtualBrowserSpan {
    z-index:10;
    xxxborder:0.5px dashed #fff !important;
    border-color:rgba(255,255,255,0.5) !important;
    position:relative;
    left:100px;
    top:100px;
    display:block;
    resize:both !important;
    width:540px;
    height:320px;
    min-width:40px !important; min-height:20px !important;
    max-width:5120px !important; max-height:2880px !important;
    background-color:rgba(255,200,255,0.1);
    xoverflow:scroll;
}
.xWirtualBrowserLocationBar:focus {
    color:#f00;
    background-color:rgba(255,128,128,0.2);
}
.xWirtualBrowserLocationBar:active {
    color:#f00;
    background-color:rgba(128,255,128,0.2);
}
a.WirtualBrowserLocation {
    color:#ccc !important;
    text-decoration:none !important;
}
a.WirtualBrowserLocation:hover {
    color:#fff !important;
    text-decoration:underline;
}
.WirtualBrowserLocationBar {
    position:absolute;
    z-index:100000;
    display:block;
    width:400px;
    height:20px;
    padding-left:2px;
    line-height:1.1;
    vertical-align:middle;
    font-size:14px;
    color:#fff;
    background-color:rgba(128,128,128,0.2);
    font-family:Georgia;
}
.WirtualBrowserCommandBar {
    position:absolute;
    z-index:200000;
    xxxdisplay:inline;
    display:block;
    width:60px;
    height:20px;
    line-height:1.1;
    vertical-align:middle;
    font-size:14px;
    color:#fe4;
    background-color:rgba(128,128,128,0.1);
    font-family:Georgia;
    text-align:left;
    left:404px;
}
.WirtualBrowserFrame {
    xxposition:relative;
    position:absolute;
    xxdisplay:inline;
    display:block;
    z-index:10;
    resize:both !important;
    width:480px; height:240px;
    min-width:60px; min-height:30px;
    max-width:5120px; max-height:2880px;
    border-radius:6px;
    background-color:rgba(255,255,255,0.9);
    border-top:20px solid;
    border-right:4px solid;
    border-bottom:10px solid;
}
.WinFavicon {
    width:16px;
    height:16px;
    margin:1px;
    margin-right:3px;
    vertical-align:middle;
    background-color:rgba(255,255,255,1.0);
}
.WirtualDesktopMenuBar {
    xposition:absolute;
    color:#fff;
    font-size:7pt;
    text-align:right;
    padding-right:4px;
    background-color:rgba(128,128,128,0.7);
}
.WirtualDesktopCalender {
    color:#fff;
    font-size:22pt;
    text-align:right;
    padding-right:4px;
    xbackground-color:rgba(255,255,255,0.2);
}
.xxxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
    display:inline !important; font-size:10pt !important; padding:1px !important;
}
.WD_Config {
    display:inline !important;
    padding:2px !important;
    font-size:10pt !important;
    width:60pt !important;
    height:12pt !important;
    line-height:1.0pt !important;
    height:15pt !important;
}
.WD_Button {
    display:inline !important;
    padding:2px !important;
    color:#fff !important;
    background-color:#228 !important;
    font-size:10pt !important;
    width:60pt !important;
    height:12pt !important;
    line-height:1.0pt !important;
    height:16pt !important;
    border:2px inset #44a !important;
}
.WD_Href {
    display:inline !important;
    padding:2px !important;
    font-size:9pt !important;
    width:120pt !important;
    height:12pt !important;
    line-height:1.0pt !important;
    height:15pt !important;
}

.LiveHtmlCodeviewText {
    font-size:10pt;
    font-family:Courier New;
    xwhite-space:pre;
}

.WD_Panel {
    x-index:100 !important;
    color:#000 !important;
    margin-left:25px !important;
    width:800px !important;
    padding:4px !important;
    border:1px solid #888 !important;
    border-radius:6px !important;
    background-color:rgba(220,220,220,0.9) !important;
    font-size:9pt;
    font-family:Courier New;
}
.WD_Help {
    font-size:10pt !important;
    font-family:Courier New;
    line-height:1.2 !important;
    color:#000 !important;
    width:100% !important;
    background-color:rgba(240,240,255,0.8) !important;
}

.WB_Zoom {
}
</style>
<h2>CosmoScreen 0.0.8</h2>
<menu>
<span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
 g ... grid on/off<br>
 i ... zoom in<br>
 o ... zoom out<br>
 s ... save current scope<br>
 r ... restore saved scope<br>
</span>
```

```
</menu>
<div class="WD_Panel" draggable="true">
<p><!-- should be on the frame of the WD -->
Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
 < <input id="WD_Width_1" class="WD_Config" type="text">
 x <input id="WD_Height_1" class="WD_Config" type="text">
 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaler03.png</a>
</p>
<p>
Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
 < <input id="WS_1_Width" class="WD_Config" type="text">
 x <input id="WS_1_Height" class="WD_Config" type="text">
</p>
<p>
Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
 < <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
 < <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
 < <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
</p>
<p>
Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
shift+wheel for horizontal scroll
</p>
<p>
Scopexx <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
 < <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
</p>
<p>
Scopeyy <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
 < <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
</p>
<p>
Scopezz <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
 < <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
</p>
<p>
Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
</p>
<p>
Overflo <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
"scroll" imprisons windows inside the disiplay
</p>
</div>

<div id="WirtualDesktop_1" class="WirtualDesktop" draggable="true" contenteditable="true" style="">
<div id="WirtualDesktop_1_MenuBar" class="WirtualDesktopMenuBar" spellcheck="false">
<i>CosmoScreen 0.0.8</i><span id="WirtualDesktop_1_Clock"></span>
</div>
<div id="WirtualDesktop_1_Calender" class="WirtualDesktopCalender ">00:00</div>
<div align=right><h1>WirtualSpace 1.</h1></div>
<div id="WirtualDesktop_1_Content" class="WirtualSpace">

<div id="WirtualBrowser_1" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
<div id="WirtualBrowser_1_Location" class="WirtualBrowserLocationBar"></div>
<span id="WirtualBrowser_1_Command" class="WirtualBrowserCommandBar">Reload</span>
<iframe id="WirtualBrowser_1_Frame" class="WirtualBrowserFrame" style=""></iframe>
</div>

<div id="WirtualBrowser_2" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
<div id="WirtualBrowser_2_Location" class="WirtualBrowserLocationBar"></div>
<span id="WirtualBrowser_2_Command" class="WirtualBrowserCommandBar">Reload</span>
<iframe id="WirtualBrowser_2_Frame" class="WirtualBrowserFrame" style=""></iframe>
</div>

<div id="WirtualBrowser_3" class="WirtualBrowserSpan" spellcheck="false" draggable="true">
<div id="WirtualBrowser_3_Location" class="WirtualBrowserLocationBar"></div>
<span id="WirtualBrowser_3_Command" class="WirtualBrowserCommandBar">Reload</span>
<iframe id="WirtualBrowser_3_Frame" class="WirtualBrowserFrame" style=""></iframe>
</div>

<div id="WirtualDesktop_1_GridPlane" class="WirtualSpace"></div>
</div>
</div>

<input class="HtmlCodeviewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
<span id="WirtualDesktopCodeview"></span>
<script id="WirutalDesktopScript">
function vd_showHtmlCode(){
    codespan = document.getElementById('WirtualDesktopCodeSpan');
    showHtmlCode(WirtualDesktopCodeview,codespan);
    WirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
}
DestroyEventSharingCodeview = function(){
    WirtualDesktopCodeview.innerHTML = "";
}

function wdlog(log){
    if( GJ_Channel != null ){
        GJ_SendMessage('WD '+log);
    }
    console.log(log);
}
var topMostWin = 10000;
function onEnterWin(e){
    t = e.target;
    oindex = t.style.zIndex;
    //if( oindex == '' ) oindex = 0;
    //t.saved_zIndex = oindex;
    //t.style.zIndex = 10000;
    topMostWin += 1;
    t.style.zIndex = topMostWin;
    nindex = t.style.zIndex;
    wdlog('Enter '+t+' #'+t.id+'('+oindex+'->'+nindex+')');
    e.stopPropagation();
    e.preventDefault();
}
function onClickWin(e){ // can detect click on the thick border?  t = e.target;
    oindex = t.style.zIndex;
    topMostWin += 1;
    t.style.zIndex = topMostWin;
    nindex = t.style.zIndex;
    wdlog('Click '+t+' #'+t.id+'('+oindex+'->'+nindex+')');
    //e.stopPropagation();
    //e.preventDefault();
}
```

```
function onLeaveWin(e){
    t = e.target;
    //oindex = t.style.zIndex;
    //nindex = t.saved_zIndex;
    //t.style.zIndex = nindex;
    //wdlog('Leave '+e.target+' #'+e.target.id+'('+oindex+'->'+nindex+')');
    e.stopPropagation();
    e.preventDefault();
}

var WinDragstartX; // event
var WinDragstartY;
var WinDragstartTX; // target
var WinDragstartTY;

function onWinDragstart(e){
    WinDragstartX = e.x;
    WinDragstartY = e.y;

    t = e.target;

    //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
    //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
    if( t.style.left == '' ){
        WinDragstartTX = x0 = 0;
        t.style.left = '0px';
    }else{
        //WinDragstartTX = x0 = Number(t.style.left);
        WinDragstartTX = x0 = parseInt(t.style.left);
    }
    if( t.style.top == '' ){
        WinDragstartTY = y0 = 0;
        t.style.top = '0px';
    }else{
        //WinDragstartTY = y0 = Number(t.style.top);
        WinDragstartTY = y0 = parseInt(t.style.top);
    }
    if( true ){ // to be undo
        t.wasAtX = WinDragstartTX;
        t.wasAtY = WinDragstartTY;
    }
    wdlog('DragSTA #'+t.id
        + ' event('+e.x+','+e.y+')'
        + ' position=' + t.style.position
        + ' style left,top('+t.style.left+','+t.style.top+')'
    );
    e.stopPropagation();
    //e.preventDefault();
    return true;
}
function onWinDragEvent(wh,e,set,dolog){
    t = e.target;
    dx = e.x - WinDragstartX;
    dy = e.y - WinDragstartY;
    nx = WinDragstartTX + dx;
    ny = WinDragstartTY + dy;
    log = 'Drag'+wh+' #'+t.id
        + ' event0('+WinDragstartX+','+WinDragstartY+')'
        + ' event('+e.x+','+e.y+')'
        + ' diff('+dx+','+dy+')'
        + ' (' + nx + ',' + ny + ')'
        + ' (' + t.style.left + ',' + t.style.top + ')'
        + ' wasAt(' + t.wasAtX + ',' + t.wasAtY + ')'
    ;
    if( e.x != 0 || e.y != 0 ){
        if( set == true ){
            //t.style.x = nx + 'px'; // not effective
            //t.style.y = ny + 'px'; // not effective
            t.style.left = nx + 'px';
            t.style.top  = ny + 'px';
            log += ' Set';
        }else{
            log += ' NotSet';
            if( !dolog ){
                log = '';
            }
        }
    }else{
        log += ' What?'; // the type is event start?
        if( !dolog ){
            log = '';
        }
    }
    if( and(dolog, log != '') ){
        wdlog(log);
    }
    if( true ){
        // should be propargeted to parent in FireFox ?
        e.stopPropagation();
    }
    e.preventDefault();
    return false;
}
function onWinDrag(e){
    return onWinDragEvent('Ing',e,true,false);
}
function onWinDragend(e){
    return onWinDragEvent('End',e,false,true);
}
function onWinDragexit(e){
    return onWinDragEvent('Exit',e,false,true);
}
function onWinDragover(e){
    return onWinDragEvent('Over',e,false,true);
}
function onWinDragenter(e){
    return onWinDragEvent('Enter',e,false,true);
}
function onWinDragleave(e){
    return onWinDragEvent('Leave',e,false,true);
}
function onWinDragdrop(e){
    return onWinDragEvent('Drop',e,false,true);
}
function onFaviconChange(e){
    wdlog('--Favicon #'+e.target.id+' href='+e.details.href);
}
var savedSuppressGJShell = false;
function stopGShell(e){
    //wdlog('enter Gsh STOP');
    savedSuppressGJShell = SuppressGJShell;
    SuppressGJShell = true;
    e.stopPropagation();
    e.preventDefault();
```

```
}
function contGShell(e){
    //wdlog('leave Gsh STOP');
    SuppressGJShell = savedSuppressGJShell;
    e.stopPropagation();
    e.preventDefault();
}

function WD_onKeydown(e){
    keycode = e.code;
    console.log('Keydown #'+e.target.id+' '+keycode);
    if( keycode == 'KeyG' ){
        WD_setGrid1(WD_Grid_1);
    }else
    if( keycode == 'KeyI' ){
        WD_doZoomIN();
    }else
    if( keycode == 'KeyO' ){
        WD_doZoomOUT();
    }else
    if( keycode == 'KeyR' ){
        WD_RestoreScope(null);
    }else
    if( keycode == 'KeyS' ){
        WD_SaveScope(null);
    }
    e.stopPropagation();
    e.preventDefault();
}
function WD_onKeyup(e){
    e.stopPropagation();
    e.preventDefault();
}
WirtualDesktop_1.addEventListener('keydown',    e => { WD_onKeydown(e); });
WirtualDesktop_1_Content.addEventListener('keydown',    e => { WD_onKeydown(e); });
WD_Help_1.addEventListener('keydown',   e => { WD_onKeydown(e); });
WD_Help_1.addEventListener('keyup', e => { WD_onKeyup(e); });

function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
    function WirtualBrowserCommand(e,s,l,cmd,f){
        command = cmd.innerHTML
        if( command == "Reload" ){
            href_id = e.target.href_id;
            d = document.getElementById(href_id);
            wdlog('href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d);
            url = d.innerHTML;
            wdlog('---- Load href_tag=#'+href_id+'\n elem=#'+href_id+'\n href='+d
                +'\n url='+url);
            wdlog('---- Load target #'+f.id)+' with url='+url);
            f.src = url;
        }else{
            alert('unknown command"'+command+'" '+e.target.id+','+l.id+','+f.id);
        }
    }
    function onKeyDown(e){
        if( e.code == 'Enter' ){
            e.stopPropagation();
            e.preventDefault();
        }
    }
    function onKeyUp(e){
        if( e.code == 'Enter' ){
            e.stopPropagation();
            e.preventDefault();
            // should reload immediately ?
        }
    }

    if( false ){
        wdlog('start settle WirtualBrowser url='+u +'\n'
            + 'id=' + s.id + '\n'
            + 'width=' + s.style.width + '\n'
            + 'height=' + s.style.height
        );
    }
    // very iportant for WordPress ??
    s.style.width = f.style.width = 501; // for WordPress ...??
    s.style.height = f.style.height = 271; // for WordPress ...??
    if( false ){
        wdlog('midway settle WirtualBrowser url='+u +'\n'
            + 'id=' + s.id + '\n'
            + 'width=' + s.style.width + '\n'
            + 'height=' + s.style.height
        );
    }
    s.width = 502; // for WordPress ...??
    s.height = 272; // for WordPress ...??
    if( false ){
        wdlog('midway-2 settle WirtualBrowser url='+u +'\n'
            + 'id=' + s.id + '\n'
            + 'span-width=' + s.width + '\n'
            + 'span-height=' + s.height
        );
    }

    s.style.width = w + 'px';
    s.style.height = h + 'px';
    f.style.width = w + 'px';
    f.style.height = h + 'px';
    //f.style.setProperty('-webkit-transform','scale('+scale+')');
    f.style.setProperty('transform','scale('+scale+')');

    //wdlog('--x1-- u='+u+' width s='+s.style.width+',f='+f.style.width);
    //wdlog('--x2-- u='+u+' width s='+s.style.width+',f='+f.style.width);
    s.setAttribute('draggable','true')
    f.setAttribute('draggable','false'); // why necessary?
    l.setAttribute('draggable','false'); // why necessary?
    cmd.setAttribute('draggable','false'); // why necessary?
    s.addEventListener('dragstart', e => { onWinDragstart(e); });
    s.addEventListener('drag',  e => { onWinDrag(e); });
    s.addEventListener('exit',  e => { onWinDragexit(e); });
    s.addEventListener('dragend',   e => { onWinDragend(e); });
    s.addEventListener('dragexit',  e => { onWinDragexit(e); });
    s.addEventListener('dragenter', e => { onWinDragenter(e); });
    s.addEventListener('dragover',  e => { onWinDragover(e); });
    s.addEventListener('dragleave', e => { onWinDragleave(e); });
    s.addEventListener('drop',  e => { onWinDragdrop(e); });

    s.addEventListener('mouseenter',e => { onEnterWin(e); });
    s.addEventListener('mouseleave',e => { onLeaveWin(e); });

    if( false ){
        s.style.position = "absolute";
        s.style.x = x+'px';
        s.style.left = x+'px';
        s.style.y = y+'px';
        s.style.top = y+'px';
    }else{
        s.style.setProperty('position','absolute','important');
        s.style.setProperty('x',x+'px','important');
        s.style.setProperty('left',x+'px','important');
        s.style.setProperty('y',y+'px','important');
        s.style.setProperty('top',y+'px','important');
    }

    favicon = './favicon.ico';
    uv1 = u.split('://');
    if( 2 <= uv1.length ){
        uv2 = uv1[1].split('/');
        if( 2 <= uv2.length ){
            if( uv1[0] == 'file' ){
                //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/')
                //  + '/favicon.ico';
                favcion = './favicon.ico';
            }else{
                favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
            }
        }
    }
    //wdlog("----- favicon-url="+favicon);
    href_id = l.id + '_href';
    l.innerHTML = '<img class="WinFavicon" src="'+favicon+'">'
        + '<a id="'+href_id+'" class="WirtualBrowserLocation" href="'+u+'">'+u+'</a>';
    //l.addEventListener('click',   e => { onClickWin(e); });
    l.addEventListener('mouseenter',e => { stopGShell(e); });
    l.addEventListener('mouseleave',e => { contGShell(e); });
    l.addEventListener('keydown',   e => { onKeyDown(e); });
    l.addEventListener('keyup', e => { onKeyUp(e); });

    cmd.href_id = href_id;
    wdlog('(0)cmd=#'+cmd.id);
    wdlog('(1)href_id=#'+href_id);
    wdlog('(2)href_id=#'+cmd.href_id);
    cmd.addEventListener('click',   e => { WirtualBrowserCommand(e,s,l,cmd,f); });

    f.style.borderColor = c;
    f.src = u;
    //f.addEventListener('mouseenter',e => { onEnterWin(e); });
    //f.addEventListener('mouseleave',e => { onLeaveWin(e); });

    //s.addEventListener('click',   e => { onClickWin(e); });
    //f.addEventListener('click',   e => { wdlog('click wb1'); });
    f.addEventListener('mozbrowsericonchange',onFaviconChange);

    wdlog('done settle WirtualBrowser url='+u +'\n'
        + 'id=' + s.id + ' '
        + 'width=' + s.style.width + ' '
        + 'height=' + s.style.height + ' '
        + 'cmd=' + cmd.id
    );
}

dt = WirtualDesktop_1;
dt.style.width = "800px";
dt.style.height = "500px";
dt.addEventListener('dragstart',e => { onWinDragstart(e); });
dt.addEventListener('drag', e => { onWinDrag(e); });
dt.addEventListener('exit', e => { onWinDragexit(e); });

function GRonClick(){
    WD_SaveScope(null); // should be push
    t = event.target;
    x = t.getAttribute('data-leftx');
    y = t.getAttribute('data-topy');
    zoom = WD_Zoom_1_XY.value;
    x *= zoom;
    y *= zoom;
    WD_doScrollXY(event,x,y);
    //alert('scroll #'+t.id+' x='+x+', y='+y);
}
function WD_setGrid(e){
    t = e.target;
    WD_setGrid1(t);
}
function WD_setGrid1(t){
    //ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
    ds = WirtualDesktop_1_GridPlane;
    if( t.value == 'GridOn' ){
        for( col = 0; col < 16; col++ ){
            for( row = 0; row < 16; row++ ){
                g1 = document.createElement('span');
                g1.setAttribute('class','WirtualGrid');
                leftx = col * 800;
                topy = row * 500;
                gid = col + '.' + row
                label = '<'+'span '
                    + 'id="'+gid+'" '+'class="WD_GridScroll" '
                    + 'contenteditable="false" onclick="GRonClick()" '
                    + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
                    + '>'
                    + gid + '<'+'/span>';
                console.log('grid '+label);
                g1.innerHTML = label;
                g1.position = 'relative';
                g1.leftx = leftx;
                g1.topy = topy;
                g1.style.left = g1.leftx + 'px';
                g1.style.top = g1.topy + 'px';
                if( col % 2 == row % 2 ){
                    g1.style.backgroundColor = 'rgba(255,255,255,0.3)';
                }
                ds.appendChild(g1);
            }
        }
        t.value = 'GridOff';
    }else{
        ds.innerHTML = '';
        t.value = 'GridOn';
    }
}
WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });

var sav_scrollLeft;
var sav_scrollTOp;
var sav_nscale;
function WD_SaveScope(e){
    sav_scrollLeft = WD_Left_1.value;
    sav_scrollTop = WD_Top_1.value;
    sav_nscale = WD_Zoom_1_XY.value;
    //console.log('saved zoom='+sav_oscale+','+sav_nscale);
}
function WD_RestoreScope(e){
    WD_Zoom_1_XY.value = sav_nscale;
    WD_doZoom();

    WD_Left_1.value = sav_scrollLeft;
    WD_Top_1.value = sav_scrollTop;
    WD_doScroll(null);
}
WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });

function ignoreEvent(e){
    e.stopPropagation();
    //e.preventDefault();
}
WD_Width_1.value = dt.style.width;
WD_Width_1.addEventListener('keydown',ignoreEvent);
WD_Width_1.addEventListener('keyup',ignoreEvent);
WD_Height_1.value = dt.style.height;
WD_Height_1.addEventListener('keydown',ignoreEvent);
WD_Height_1.addEventListener('keyup',ignoreEvent);

function zoomMag(){
    return WD_Zoom_1_MAG.value;
}
WD_Zoom_1_MAG.addEventListener('keydown',ignoreEvent);
WD_Zoom_1_MAG.addEventListener('keyup',ignoreEvent);

function escale1(e,oscale,nscale){
    e.style.setProperty('transform','scale('+nscale+')');
    rscale = oscale / nscale;
    w = parseInt(e.style.width);
    h = parseInt(e.style.height);
    w = w * rscale; //(oscale/nscale);
    h = h * rscale; //(oscale/nscale);
    e.style.width = w + 'px';
    e.style.height = h + 'px';
}
function scaleWD(ds,oscale,nscale){
    if( true ){
        escale1(WirtualBrowser_1,oscale,nscale);
        escale1(WirtualBrowser_1_Location,oscale,nscale);
        escale1(WirtualBrowser_1_Command,oscale,nscale);
        escale1(WirtualBrowser_1_Frame,oscale,nscale);

        escale1(WirtualBrowser_2,oscale,nscale);
        escale1(WirtualBrowser_2_Location,oscale,nscale);
        escale1(WirtualBrowser_2_Command,oscale,nscale);
        escale1(WirtualBrowser_2_Frame,oscale,nscale);

        escale1(WirtualBrowser_3,oscale,nscale);
        escale1(WirtualBrowser_3_Location,oscale,nscale);
        escale1(WirtualBrowser_3_Command,oscale,nscale);
        escale1(WirtualBrowser_3_Frame,oscale,nscale);
    }
}
function WD_doZoom(){
    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
    oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
    nscale = WD_Zoom_1_XY.value;
    ds.style.zoom = nscale;
    WD_Zoom_1_XY.value = ds.style.zoom;
    WD_Zoom_1_XY.ovalue = ds.style.zoom;
    scaleWD(ds,oscale,nscale);
}
WD_Zoom_1.addEventListener('click',WD_doZoom);
WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);

function WD_doZoomOUT(){
    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
    oscale = WD_Zoom_1_XY.value;
    if( oscale == 0 || oscale == '' ){
        oscale = 1;
    }
    nscale = oscale / zoomMag();
    ds.style.zoom = nscale;
    WD_Zoom_1_XY.value = ds.style.zoom;
    WD_Zoom_1_XY.ovalue = ds.style.zoom;
    scaleWD(ds,oscale,nscale);
}
function WD_doZoomIN(){
    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
    oscale = WD_Zoom_1_XY.value;
    if( oscale == 0 || oscale == '' ){
        oscale = 1;
    }
    nscale = oscale * zoomMag();
    if( 4 < nscale ){
        alert('maybe too large, zoom='+nscale);
        return;
    }
    ds.style.zoom = nscale;
    WD_Zoom_1_XY.value = ds.style.zoom;
    WD_Zoom_1_XY.ovalue = ds.style.zoom;
    scaleWD(ds,oscale,nscale);
}
WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);

function WD_doResize(e){
    dt = WirtualDesktop_1;
    dt.style.width = WD_Width_1.value;
    dt.style.height = WD_Height_1.value;
    WD_Width_1.value = dt.style.width;
    WD_Height_1.value = dt.style.height;
}
WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });


function WD_doRSResize(e){
    //alert('Resize Space');
    ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
    ds.style.width = WS_1_Width.value;
    ds.style.height = WS_1_Height.value;
    WS_1_Width.value = ds.style.width;
    WS_1_Height.value = ds.style.height;
}
ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
ds.style.width = '5120px';
ds.style.height = '2880px';
```

```
WS_1_Width.value = ds.style.width;
WS_1_Height.value = ds.style.height;
WS_1_Width.addEventListener('keydown',ignoreEvent);
WS_1_Width.addEventListener('keyup',ignoreEvent);
WS_1_Height.addEventListener('keydown',ignoreEvent);
WS_1_Height.addEventListener('keyup',ignoreEvent);
WS_Resize_1.addEventListener('click',e => { WD_doRSResize(e); });

function WD_doScrollXY(e,sleft,stop){
    dt = WirtualDesktop_1;
    dt.scrollLeft = sleft;
    dt.scrollTop = stop;
    WD_Left_1.value = dt.scrollLeft;
    WD_Top_1.value = dt.scrollTop;
    console.log('--Scroll #'+dt.id+' ('+sleft+','+stop+')');
}
function WD_doScroll(e){
    //dt = WirtualDesktop_1_Content;
    dt = WirtualDesktop_1;
    sleft = parseInt(WD_Left_1.value);
    stop = parseInt(WD_Top_1.value);
    dt.scrollLeft = sleft;
    dt.scrollTop = stop;
    WD_Left_1.value = dt.scrollLeft;
    WD_Top_1.value = dt.scrollTop;
    console.log('--Scroll #'+dt.id+' ('+sleft+','+stop+')');
}
WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
WD_Left_1.addEventListener('keydown',ignoreEvent);
WD_Left_1.addEventListener('keyup',ignoreEvent);
WD_Top_1.addEventListener('keydown',ignoreEvent);
WD_Top_1.addEventListener('keyup',ignoreEvent);

function showScrollPosition(){
    if( false )
    console.log(
        'wstop=' + WirtualDesktop_1.style.top + ',' +
        'wsx=' + WirtualDesktop_1.style.y + ',' +
        'wss=' + WirtualDesktop_1.scrollTop + ',' +
        'wdtop=' + WirtualDesktop_1_Content.style.top +',' +
        'wdx=' + WirtualDesktop_1_Content.style.y +',' +
        'wds=' + WirtualDesktop_1_Content.scrollTop + ','
    );
    WD_Left_1.value = WirtualDesktop_1.scrollLeft;
    WD_Top_1.value = WirtualDesktop_1.scrollTop;
}
WirtualDesktop_1.addEventListener('scroll',showScrollPosition);
WirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);


if( false ){
w = 1000 + 'px';
dt.style.width = w;
dt.style.height = "300px";
dt.style.resize = 'both';
dt.style.borderWidth = 50 + 'px';
dt.style.borderRadius = 25 + 'px';
console.log('--2---------- #'+dt.id+' style=' +dt.style);
console.log('-------------- #'+dt.id+' width=' +dt.style.width);
console.log('-------------- #'+dt.id+' left='  +dt.style.left);
console.log('-------------- #'+dt.id+' border='+dt.style.border);
}

function onDTResize(e){
    dt = e.target;
    h = parseInt(dt.style.height);
    dt.style.borderWidth = (h * 0.075) + 'px';
    console.log('---------------- borderWidgh='+dt.style.borderWidth);
}
WirtualDesktop_1.addEventListener('resize', e => { onDTResize(e); });
//console.log('-------------- #'+dt.id+' borderWidth='+dt.style.getProperty('border-width'));

settleWin(
    WirtualBrowser_1,
    WirtualBrowser_1_Location,
    WirtualBrowser_1_Command,
    WirtualBrowser_1_Frame,
    document.URL,
    500,280,50,20,'#262',1.0);
settleWin(
    WirtualBrowser_2,
    WirtualBrowser_2_Location,
    WirtualBrowser_2_Command,
    WirtualBrowser_2_Frame,
    'https://its-more.jp/ja_jp/',
    500,280,150,100,'#448',1.0);

settleWin(
    WirtualBrowser_3,
    WirtualBrowser_3_Location,
    WirtualBrowser_3_Command,
    WirtualBrowser_3_Frame,
    //'../gshell/gsh.go.html',
        //'http://gshell.org/gshell/gsh.go.html',
    'https://golang.org',
    500,280,250,180,'#444',1.0);
    //1000,720,0,0,'#444',0.125);
    //1200,720,-100,-50,'#444',0.4);

function WD_ClockUpdate(e){
    WirtualDesktop_1_Clock.innerHTML = DateShort();
    WirtualDesktop_1_Calender.innerHTML = DateHourMin();
}
window.setInterval(WD_ClockUpdate,500);
//GJ_Join();

Destroy_WirtualDesktop = function(){
    WirtualDesktop_1.style = "";

    WirtualBrowser_1.removeAttribute('style');
    WirtualBrowser_1_Location.innerHTML = '';
    WirtualBrowser_1_Frame.removeAttribute('src');
    WirtualBrowser_1_Frame.removeAttribute('style');
    WirtualBrowser_1_Frame.style="";

    WirtualBrowser_2.removeAttribute('style');
    WirtualBrowser_2_Location.innerHTML = '';
    WirtualBrowser_2_Frame.removeAttribute('src');
    WirtualBrowser_2_Frame.style="";

    WirtualBrowser_3.removeAttribute('style');
    WirtualBrowser_3_Location.innerHTML = '';
    WirtualBrowser_3_Frame.removeAttribute('src');
    WirtualBrowser_3_Frame.style="";

    GJFactory_1.style = "";
    iframe0.style = "";
    WirtualDesktop_1.style = "";
}

</script>
<!-- WirtualDesktop } -->
</details>
*/ //</span>

//<!-- ---------- Work { ---------- -->
//<span id="SBSidebar_WorkCodeSpan">
/*
<details><summary>SBSidebar</summary>
<!-- ---------- SBSidebar // 2020-0928 SatoxITS { -->
<h2>SBSidebar</h2>
<input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
<input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
<input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
<span id="SBSidebar_WorkCodeView"></span>
<script id="SBSidebar_WorkScript">
function SBSidebar_openWorkCodeView(){
    function SBSidebar_showWorkCode(){
        showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
    }
    SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
}
SBSidebar_openWorkCodeView(); /// should be invoked by an event

console.log('-- SbSlider // 2020-1006-01 SatoxITS --');
function SetSidebar(){
    sidebar = document.getElementById('secondary');
//  console.log('primary='+primary+' + 'secondary='+sidebar+' + 'main='+main+' ');
    wrap = sidebar.parentNode;
    console.log('-- SbSlider parent is '+wrap+', #'+wrap.id+' .'+wrap.class);
//wwrap = wrap.parentNode;
//console.log('-- SbSlider parent is '+wwrap+', #'+wwrap.id+' .'+wwrap.class);
//wwwrap = wwrap.parentNode;
//console.log('-- SbSlider parent is '+wwwrap+', #'+wwwrap.id+' .'+wwwrap.class);
//nsb = sidebar.cloneNode();
    nsb = sidebar;
    nsb.style.width = '100%';
    slider = document.createElement('div');
    slider.id = 'SbSlider';
    slider.appendChild(nsb);
    slider.setAttribute('class','SbSlider');
    nsb.style.position = 'relative';
    slider.style.position = 'fixed';
    slider.style.display = 'block'; //'inline';
    slider.style.zIndex = 100000;
    // nsb.style.zIndex = 200000;
    nsb.style.position = 'absolute';
    nsb.style.minWidth = '80px';
    nsb.style.left = '0px';
    nsb.style.top = '0px';

    w = window.innerWidth;
    console.log('SliderWidth '+w+': ',(w/3)+'px');
    if( w < 640 ){
        slider.style.setProperty('width',(w/3) + 'px','important');
    }
    main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';

    slider.style.resize = "both";
    slider.draggable = "true";
    wrap.appendChild(slider);
    console.log('-- added SbSlider');
    //nsb.addEventListener('scroll',SbScrolled);

    buttons = document.createElement('div');
    buttons.id = 'NaviButtons';
    buttons.setAttribute('class','NaviButtons');
    buttons.align = "center";
    buttons.innerHTML += '<'+'p><a href="#TopOfPost" draggable="true">TOP<'+'/a><'+'/p>';
    buttons.innerHTML += '<'+'p><a href="#EndOfPost" draggable="true">END<'+'/p>';
    page.appendChild(buttons);
    buttons.style.position = 'fixed';
    buttons.style.zIndex = 30000;
    buttons.style.width = '180%';
    buttons.style.top = '320px';
    buttons.style.left = parseInt(w) * 1.0 + 'px';
    console.log('-- SbSlider installed (^-^)/ SatoxITS');
}
//window.addEventListener('load',SetSidebar); // after load
DestroyNaviButtons = function(){
    nb = document.getElementById('NaviButtons');
    if( nb != null ){
        nb.parentNode.removeChild(NaviButtons);
    }
}
</script>

// 2020-1006 its-more.jp-blog-60000-style.css
<!-- {
<style>
#NaviButtons {
    position:fixed;
    display:block;
    xwidth:100%;
    xxtop:100px;
    xxleft:10px;
    z-index:30000;
    font-size:10pt;
    color:#2ff !important;
    text-align:center;
    background-color:rgba(230,230,230,0.01);
}
#NaviButtons a {
    color:#2a2 !important;
    font-size:20px;
    text-align:center;
    xxtext-shadow:2px 2px #8ff;
    resize:both;
    padding:6px;
    margin:10px;
    border:1px solid #288 !important;
    border-radius:3px;
    background-color:rgba(160,160,160,0.05);
}
#SbSlider {
    overflow:auto;
    resize:both !important;
    xxoverflow-y:hidden !important;
    height:100px !important;
    display:inline !important;
    position:fixed !important;
    left:0px;
    top:0px;
    xxwidth:180px;
    width:24%;
    min-width:80px;
    height:100% !important;
    background-color:rgba(100,100,200,0.1);
}
#secondary {
    position:fixed;
    left:0px;
    top:0px;
    xxxz-index:60000;
    scroll-behavior: overflow !important;
    xxxxxxxxxxxxxxxxxxxxxxxxxxwidth:18% !important;
    padding-left:4pt;
    color:#fff;
    font-size:0.5em;
    background-color:rgba(64,160,64,0.6) !important;
    white-space:nowrap;
}
#secondary a {
    color:#fff !important;
    text-decoration:disable !important;
}
#primary {
    position:relative;
    width:75% !important;
    left:25% !important;
}
#main {
    position:relative;
    width:75% !important;
    left:25% !important;
}
#site-navigation {
    position:relative;
    left:120px;
}
#adswsc_countertext {
    color:#4169e1;
    font-size:16pt !important;
    xxfont-size:10% !important;
    font-weight:bold;
}
#nowTime {
    color:#a0ffa0;
    font-size:16pt !important;
    xxfont-size:10% !important;
    font-weight:bold;
    text-shadow:1px 1px #fff;
}
.navigation-top {
    color:#22a !important;
    border:0px;
    background-color:rgba(220,220,220,0.1);
}

.visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
  display: block;
  xxwidth: 1em;
  xxoverflow: auto;
  xxheight: 1em;
}
.invisible-scrollbar ::-webkit-scrollbar {
  xxdisplay: none;
 width:1px !important;
 height:1px !important;
}
.mostly-customized-scrollbar ::-webkit-scrollbar {
  width: 2px;
  height: 2px;
  xxbackground-color: #aaa; xxx:or add it to the track;
}
.mostly-customized-scrollbar ::-webkit-scrollbar-thumb {
    background: #000;
}
</style>
} -->


</details>
<!-- SBSidebar_WorkCodeSpan } -->
*/ //</span>
//<!-- ---------- Work } ---------- -->




//<!-- ---------- Work { ---------- -->
//<span id="Template_WorkCodeSpan">
/*
<details><summary>Work Template</summary>
<!-- ---------- Template of Work// 2020-0928 SatoxITS { -->
<h2>Template of Work</h2>
<input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="Source">
<input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
<input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
<span id="Template_WorkCodeView"></span>
<script id="Template_WorkScript">
function Template_openWorkCodeView(){
    function Template_showWorkCode(){
        showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
    }
    Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
}
Template_openWorkCodeView(); /// should be invoked by an event
</script>
</details>
<!-- Template_WorkCodeSpan } -->
*/ //</span>
//<!-- ---------- Work } ---------- -->

//<script>Indexer_afterLoaded();</script>

//</div>
//<br></span></html>

```