TouchSend Tools

Custom Embedded Controls
Event Functions - Time and Media
Direct Viewer links to other Windows Applications
Baggage Services
INI Read and Write Functions and more

TouchSend Services

Print Catalog to Viewer Conversions
Electronic Publishing Design and Training
Systems for Converting Massive Amounts of Text
Automating Link and Jump Regenerations

{ewc TSTOOLS,Tsbutton, "OverView"[Macro=ji(qchPath,`overview')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Buttons"[Macro=ji(qchPath,`buttons')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Functions"[Macro=ji(qchPath,`functionmenu')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Order/Exit"[Macro=ji(qchPath,`About')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Index"[Macro=ji(qchPath,`showindex')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Help"[Macro=TsHelpContext(`tshelp.hlp',2)][Font="Arial" /S11/B4/3-] /W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Read Me"[Macro=ji(qchPath,`readme')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Licence"[Macro=ji(qchPath,`support')][Font="Arial" /S11/B4/3-]/W61/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"TsToolsW Online"[Macro=ji(qchPath,`dtsdocs')][Font="Arial" /S11/B4/3-]/W125/H18/B1/D2} {ewc TSTOOLS, Tsbutton,"Timer/MCI demo"[Macro=pi(`tstools.mvb',`caution_re_vid>mcipopup')][Font="Arial" /S11/B4/3-]/W125/H18/B1/D2}

.

TouchSend Tools

{ewc MVMCI2, ViewerMCI, [device AVIVideo][name ts1][noframe]!tsopen.avi}

Custom
Embedded Controls

Time and Multimedia Event Management

Hierarchical. Index

And more...

..

Events and time

Can be varied

..

Programatically

{ewc TSTOOLS, Tsbutton,"+"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');TsExec(`pbrush.exe',`',`Paint',-1,1,1);TsExecPos(`Paint',320,0,319,480);JumpID(`tstools.mvb>paint', `paintdemo')][Font="Arial"/S10/B7/3-] /W16/H17/B1/D2} **Run Paintbrush...** {ewc TSTOOLS, Tsbutton,"+"[Macro=PI(qchPath, `buttonpopup3>Popup3');PI(qchPath, `buttonpopup1>popup1');PI(qchPath, `buttonpopup4>Popup4');PI(qchPath, `buttonpopup2>Popup2')][Font="Arial"/S10/B7/3-] /W16/H17/B1/D2} **Popups**

{ewc TSTOOLS, Tsbutton, "Height 70" [Font="Arial"/S11/B4/A900] [Macro = ClosePane(`main',`graphbut');pi(qchPath,`buttonh70>button')]/H70/W20/x1/y55/B1/N/D2} {ewc TSTOOLS,Tsbutton, "Click Me" [Macro=ClosePane(`main',`graphbut');TsInfoBox(4,`TouchSend Textbuttons',`',`This command was executed from a TouchSend Text Button.',`',0,255,0,`')] [Font="Arial" /S9/B4/3-]/H20/w100/B1/D2}

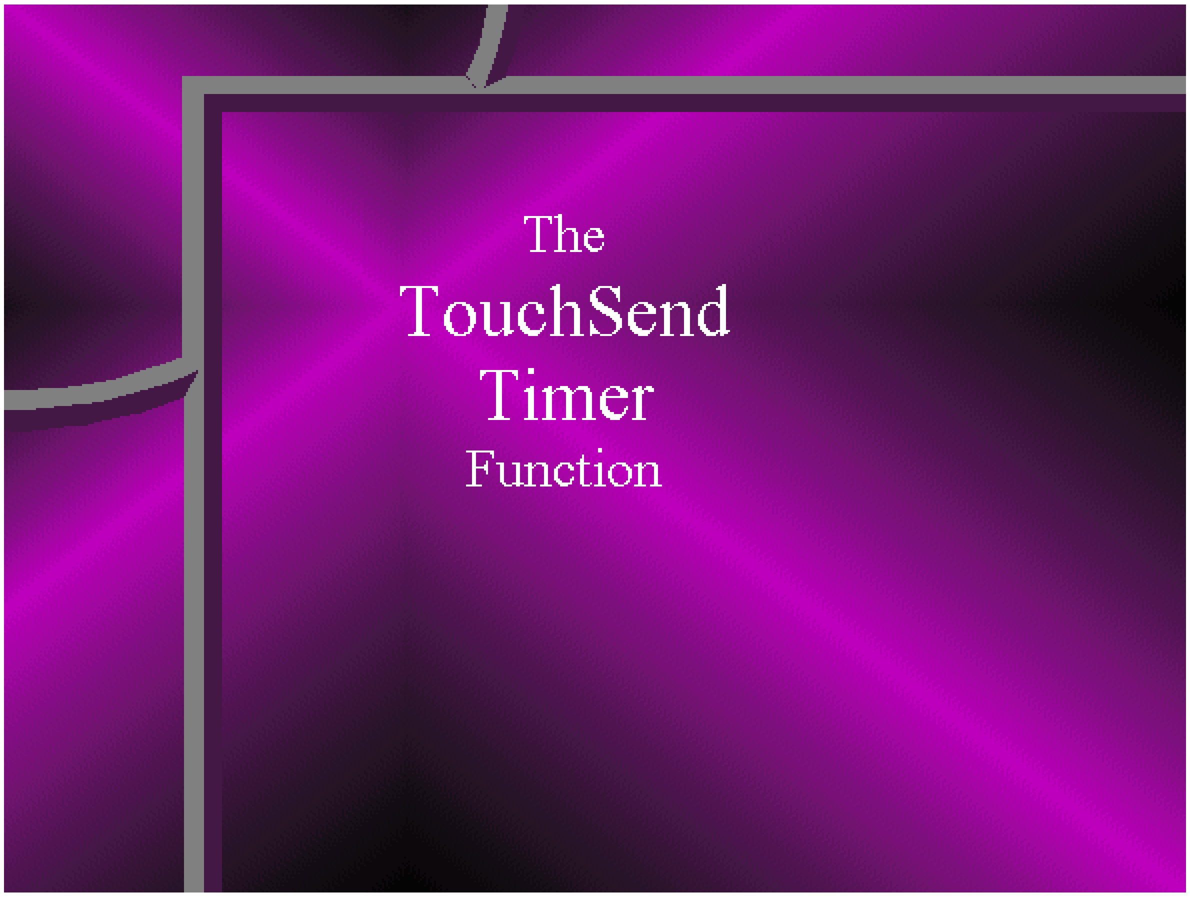
{ewc TSTOOLS, Tsbutton,"Tiny"[Macro=TsInfoBox(4,`TouchSend Textbuttons',`This command was executed from a TouchSend Text Button',`Make sure to press the donmissutton found in',` TsToolsW Online',0,255,0,`')][Font="Arial" /S8/B4/3-] /W23/H17/B1/D2} {ewc TSTOOLS, Tsbutton,"Little"[Macro=TsInfoBox(1,`About TouchSend',`TouchSend is a registered trademark of',`TouchSend Corporation',`904-668-6180',255,0,0,`')][Font=F"Arial" /S9/B4/U1] W37 /H26/B1/D2} {ewc TSTOOLS, Tsbutton,"&Medium"[Macro=TsInfoBox(3,`About TouchSend',`Do you want to do a title in Viewer but..',`need some help through the learning curve and design process ?',`Call TouchSend at 904-668-6180. Save months of work...',0,255,0,`')][Name=med][Font="Arial" /S20/B4]W78/H78/B1/D2} {ewc TSTOOLS, Tsbutton,"Big"[Macro=TsInfoBox(5,`About TouchSend',`The TsTimer Function',`submits Viewer commands',`after a time duration.',192,192,192,`')][Font="Courier New" /S46/B8/3+/A450][Name=BIG] W168 /H116/B1/D3/N/x30/y70}

{ewc TSTOOLS, Tsbutton,"Baby Pictures"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');PI(qchPath, `bevan>Bevan')] [Font="Times New Roman"/S11/B4]W356/H17/B1/D2}

{ewc TSTOOLS, Tsbutton,"White"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 255,255,255)][Font="Arial" /S11/B4/3-]W43 /H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Yellow"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 255,255,0)][Font="Arial" /S11/B4/3-]W43/H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Black"[Macro=ClosePane(`main',`graphbut');MasterNSRColor(`main', 0,0,0);PaneID(qchPath,`tsbuttonpic>tsbutton',0)][Font="Arial" /S11/B4/3-]W43/H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Red"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 128,0,0)][Font="Arial" /S11/B4/3-]W43/H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Blue"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 0,0,128)][Font="Arial" /S11/B4/3-]W43 /H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Grey"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 192,192,192)][Font="Arial" /S11/B4/3-]W43 /H30/B1/D2}{ewc TSTOOLS, Tsbutton,"Green"[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main', 0,128,0)][Font="Arial" /S11/B4/3-]W43 /H30/B1/D2}

{ewc TSTOOLS, Tsbutton,"Button Info..."[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');PI(qchPath,`tsbuttons>tsbutton')] [Font="Arial" /S11/B4/3+] W100/H32/B2/D2}{ewc TSTOOLS, Tsbutton,"Accelerator Keys"[Macro=ClosePane(`main',`tsbutton');AddAccelerator(0x4D, 4, `TsPressButton(`med',150)');PaneID(qchPath,`tskeys>graphbut',0)][Font="Arial" /S11/B4/3+] /W100/H32/B2/D2} {ewc TSTOOLS, Tsbutton,"Button Details..."[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');JI(qchPath,`tsbutton')][Font="Arial" /S11/B4/3+] W100/H32/B2/D2}

{ewc TSTOOLS, Tsbutton,"Return"[Macro=JI(qchPath,`TSFull')][Font="Arial" /S13/B4/3-] /W87 /H32/B1/D2}



The TouchSend Timer Function

```
{ewc TSTOOLS, Tsbutton, "" [Name=OnOff][Graphic=`!switchup.dib',`!switchdn.dib',`!switchdi.dib']
[Macro=ClosePane(`main',`graphbut');ClosePane(`main',`tsbutton');MasterNSRColor(`main',
128,128,0);PaneID(qchPath,`switchon>swonoff',0);PaneID(qchPath,`tsbuttongraphic>graphbut',0)]/B1/
D0/N}
```

{ewc TSTOOLS, Tsbutton, "" [Name=OnOff][Graphic=`!switchdn.dib',`!switchup.dib',`!switchdi.dib'][Macro=ClosePane(`main',`graphbut');MasterNSRColor(`main', 0,0,0);PaneID(qchPath,`switchoff>swonoff',0);TsEnableButton(`OnOff');PaneID(qchPath,`tsbuttonpic>tsbutton',0)]/B1/D0/N} {ewc TSTOOLS,TsButton,"Close~Paintbrush."/AR/ML30R3/H40/W130/B2/D1/N/2 [name=closepb][graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC][macro=TsExecKill(`paint');CloseWindow(`paint')][Font="Times New Roman"/S13/B4/3-]}

The TouchSend MCI Function

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][noframe][name vid1]!tsmcia.avi}

The TouchSend MCI Function

Of course TouchSend Buttons Can Control Popups

Of course Can Control Multiple Popups

Of course TouchSend Buttons Can Multiple Control Popups

Of course TouchSend Buttons Can Control Multiple Popups

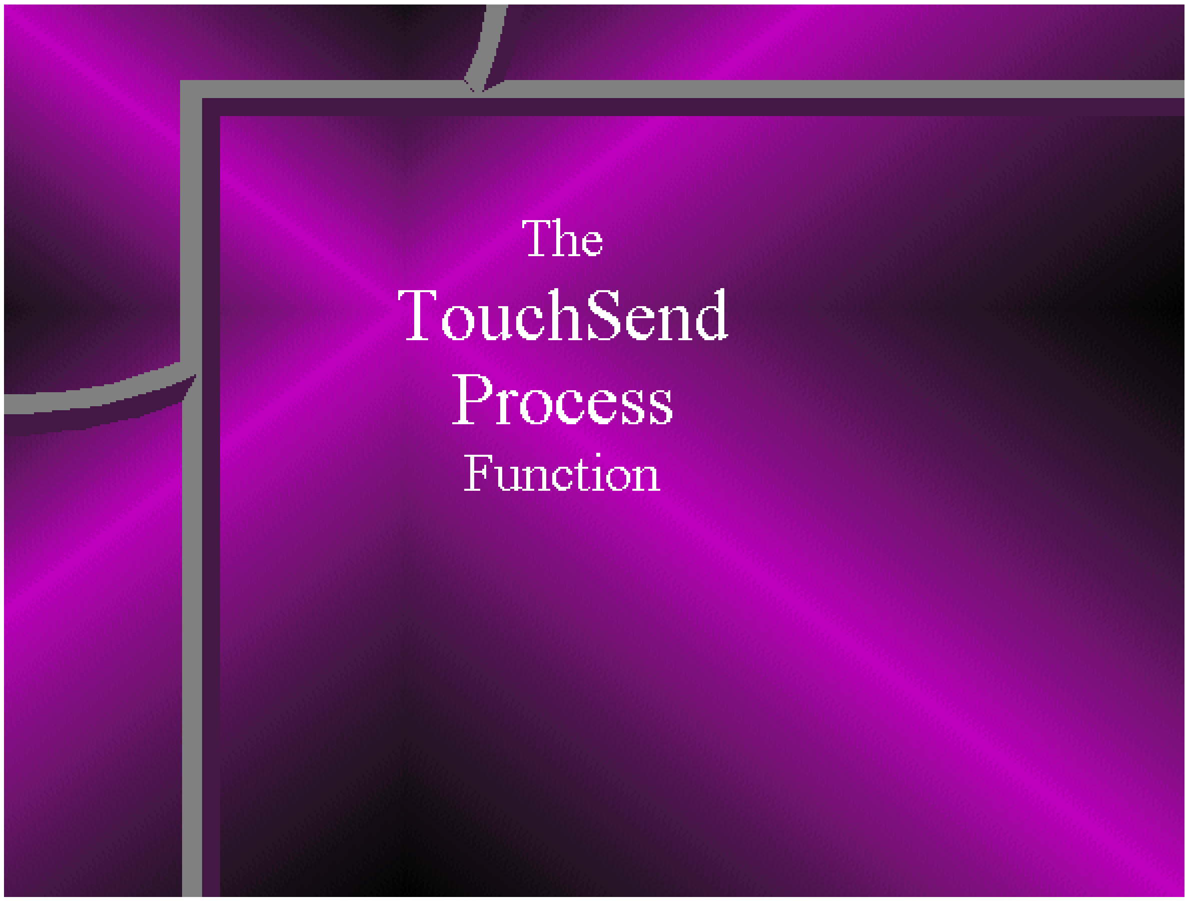
{ewc MVMCI2, ViewerMCI, [device AVIVideo][name tsvid1][autostart][stdcontrol]!tsmcib.avi} {ewc MVMCI2, ViewerMCI, [device AVIVideo][name tsvid2][autostart][stdcontrol]!tsmcic.avi}

```
{ewc TSTOOLS, tsbutton,"Continue"[Macro=ji(qchPath,`functionmenu')][Font="Arial" /S12/B4] /W100 /H80}
```

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][noframe]!tsmcia.avi}

Hello

{ewc TSTOOLS, Tsbutton,"About"[Macro=about()] [Font="Arial" /S10/B4] /W100 /H64/B2/D2/N}



The TouchSend Process Function

TouchSend Tools

```
{ewc TSTOOLS, Tsbutton,"Graphics demo"[Macro=ClosePane(`main', `arrowpan');JI(`tstools>Second', `chartdemo')][Font="Arial" /S8/B4]
/W100 /H16/B1/D2/N}
{ewc TSTOOLS, Tsbutton,"Close Write and Continue"[Macro=TsWriteKill();PositionWindow(0, 0, 1024,1024, 0,
`main');ji(qchPath,`functionmenu')][Font="Arial" /S8/B4] /W100 /H16/B1/D2/N}
```

Functions

Functions

TsWrite

TsTimer

TsMCI

TsHelp

More...

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][looping][noframe][share AVI]!bpmore.avi}

Return

TsClose

TsClose

TsVars

TsVars

TsSnd

TsSnd

TsIni

TsIni

TsInfoBox

TsInfoBox

TsPrint

TsPrint

TsMacro

TsMacro

TsYesNo

TsYesNo

TsWinStyle

TsWinStyle

TsPane

TsPane

TsSaveBag

TsSaveBag

TsExitTopic

TsExitTopic

TsWave

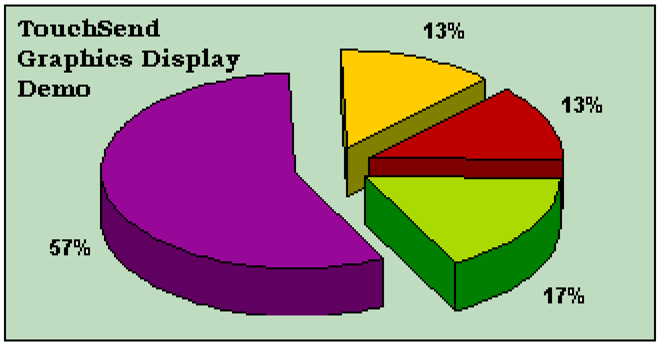
TsWave

More...

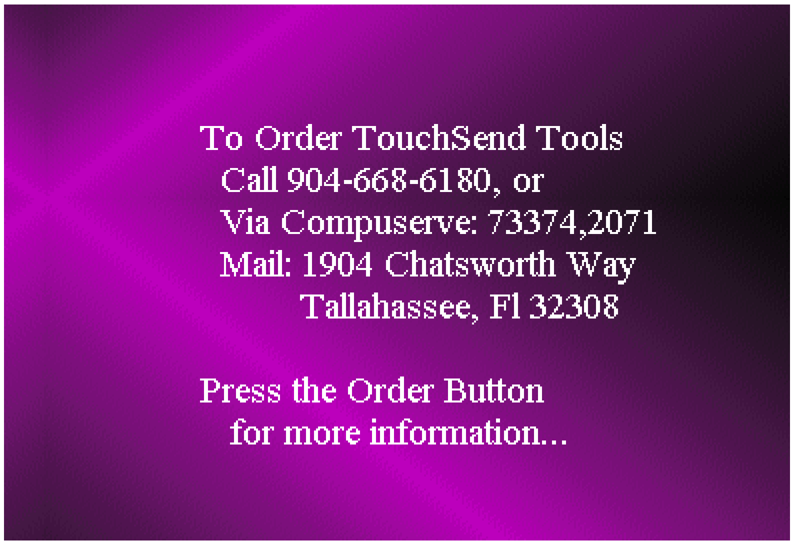
More...

Functions

Functions



{ewc TSTOOLS,tsbutton,"Help"[Macro=TsHelpContext(`tshelp1.hlp',1)][Font="Arial" /S11/B4/3-]W62/H16/B1/D2/N/P-}{ewc TSTOOLS,tsbutton,"Print"[Macro=Print()][Font="Arial" /S11/B4/3-]W62/H16/B1/D2/N/P-}{ewc TSTOOLS,tsbutton,"Close"[Macro=PaneID(`tstools>main', `arrowg>arrowgrn',0);CloseWindow(`second')][Font="Arial" /S11/B4/3-]W62/H16/B1/D2/N/P-}



To Order TouchSend Tools
Call 904-668-6180, or
Via Compuserve: 73374,2071
Mail: 1904 Chatsworth Way
Tallahassee, Fl 32308

Press the Order Button
for more information...

{ewc TSTOOLS,tsbutton,"Order..."[Macro =JumpID(`tstools>touchsnd', `orderform1')][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N} Order Information

{ewc TSTOOLS,tsbutton,"Restart"[Macro=SaveMark(`nowow');Back()][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N} To the Main Screen

{ewc TSTOOLS,tsbutton,"TouchSend"[Macro =JumpID(`tstools>touchsnd', `consult')][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N} Consulting Services

{ewc TSTOOLS,tsbutton,"Credits"[Macro=PopupId("tstools.mvb","creditlist>credits")][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N} Credits

{ewc TSTOOLS,tsbutton,"Exit"[Macro=TsYN(1,`Thanks for Viewing  the TouchSend Demo',`Do you want to Order TouchSend Tools ',`  or the TouchSend Index ?',`',128,128,0,`JumpID(`tstools>touchsnd', `orderform')',`Exit()')][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N} Exit {ewc MVMCI2, ViewerMCI, [device AVIVideo][noframe][name credits][autostart]!tsanim.avi}

The chart graphic should have pasted into Write.   If Write is not on your system in the path, this part of the demonstration will not work.   As well, issuing a command from a secondary window is documented as not consistently dependable.

It is possible that the chart graphic may not have pasted correctly on your system.   However, it is now on the clipboard and can be pasted into any application that will accept a graphic.   This is a very good technique for illustrations in textbooks.

TsVar Functions

The **TsVar Functions** allow the author to set a variable and then increment or decrement a counter and submit a command string to Viewer on reaching the count, if less than the count, more than the count etc. The count can be displayed in an embedded TsPane.

{ewc TSTOOLS, Tsbutton,"Return"[Macro=back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsInfoBox

The **TsInfoBox Function** allows the author to create a custom "about" or dialog box. The author has control of the title and three lines of text as well as the color of the infobox. There are several styles of dialog box. See also: **TSYesNo**.

{ewc TSTOOLS, Tsbutton,"Example"[Macro=TsInfoBox(4,`TouchSend Infobox',`This is the first line of a TsInfoBox ',`There can be three lines of 50 characters',`   or more with certain styles...',0,255,0,`')][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Return"[Macro=back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsIni Functions

The **TsIni Functions** allow the author to write and read settings and restore marks from disk.   As an example, an author may use an ini file to set a reading level (ie beginner, intermediate and advanced), or to set the"state" of the title upon re-entry, or with the **TsVar** functions.

{ewc TSTOOLS, Tsbutton,"Return"[Macro=back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsSnd Function

The **TsSnd Function** tests for the presence of a sound device.   The author has the option of either submitting a wave file or executing some other command if the sound device is not present.

```
{ewc TSTOOLS, Tsbutton,"Demo"[Macro=TsSnd(`MCICommand(hwndContext, qchPath, `!alarm.wav', `play');PI(qchPath,`sndyes>snd');TsTimer(`FocusWindow(`main')',3)',`PI(qchPath,`sndno>snd');TsTimer(`FocusWindow(`main')',3)')][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}
{ewc TSTOOLS, Tsbutton,"Return"[Macro=back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}
```

TsClose

The **TsClose Function** allows the author to let a user restart at the exact location displayed at the time the title is closed   This is particularly useful in titles such as textbooks, policy/procedure manuals or catalogs.

{ewc TSTOOLS, Tsbutton,"Return"[Macro=back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsClose Demo Page 1

Browse to any one of the close demo screens.   Then exit the title.   Then restart the title.   It will return to the screen you exited from.   Upon your return, you will be able to return to the functions menu.

{ewc TSTOOLS, Tsbutton,"<<"[Font="Arial" /S11/B8]/W62 /H16/B1/D2/N/-}{ewc TSTOOLS, Tsbutton,">>"[Macro=Next()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}
{ewc TSTOOLS, Tsbutton,"Exit the Title"[Macro=IfThenElse(ismark(`closedemo'),`TsInfoBox(5,`TsClose Demo',`Press the RETURN button in order to ',`properly exit this part of the demo.',`   TsClose is a Useful Function !!!',192,192,192,`')',`Exit()')][Font="Arial" /S11/B4]/W124 /H16/B1/D2/N}

TsClose Demo Page 2

Browse to any one of the close demo screens.   Then exit the title.   Then restart the title.   It will return to the screen you exited from.   Upon your return, you will be able to return to the functions menu.

{ewc TSTOOLS, Tsbutton,"<<"[Macro=Prev()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}{ewc TSTOOLS, Tsbutton,">>"[Macro=Next()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Exit the Title"[Macro=IfThenElse(ismark(`closedemo'),`TsInfoBox(5,`TsClose Demo',`Press the RETURN button in order to ',`properly exit this part of the demo.',`   TsClose is a Useful Function !!!',192,192,192,`')',`Exit()')][Font="Arial" /S11/B4]/W124 /H16/B1/D2/N}

TsClose   Demo Page 3

Browse to any one of the close demo screens.   Then exit the title.   Then restart the title.   It will return to the screen you exited from.   Upon your return, you will be able to return to the functions menu.

{ewc TSTOOLS, Tsbutton,"<<"[Macro=Prev()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}{ewc TSTOOLS, Tsbutton,">>"[Macro=Next()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}
{ewc TSTOOLS, Tsbutton,"Exit the Title"[Macro=IfThenElse(ismark(`closedemo'),`TsInfoBox(5,`TsClose Demo',`Press the RETURN button in order to ',`properly exit this part of the demo.',`   TsClose is a Useful Function !!!',192,192,192,`')',`Exit()')][Font="Arial" /S11/B4]/W124 /H16/B1/D2/N}

TsClose Demo Page 4

Browse to any one of the close demo screens.   Then exit the title.   Then restart the title.   It will return to the screen you exited from.   Upon your return, you will be able to return to the functions menu.

{ewc TSTOOLS, Tsbutton,"<<"[Macro=Prev()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}{ewc TSTOOLS, Tsbutton,">>"[Macro=Next()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Exit the Title"[Macro=IfThenElse(ismark(`closedemo'),`TsInfoBox(5,`TsClose Demo',`Press the RETURN button in order to ',`properly exit this part of the demo.',`   TsClose is a Useful Function !!!',192,192,192,`')',`Exit()')][Font="Arial" /S11/B4]/W124 /H16/B1/D2/N}

TsClose Demo Page 5

Browse to any one of the close demo screens.   Then exit the title.   Then restart the title.   It will return to the screen you exited from.   Upon your return, you will be able to return to the functions menu.

{ewc TSTOOLS, Tsbutton,"<<"[Macro=Prev()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}{ewc TSTOOLS, Tsbutton,">>"[Font="Arial" /S11/B8]/W62 /H16/B1/D2/N/-} {ewc TSTOOLS, Tsbutton,"Exit the Title"[Macro=IfThenElse(ismark(`closedemo'),`TsInfoBox(5,`TsClose Demo',`Press the RETURN button in order to ',`properly exit this part of the demo.',`   TsClose is a Useful Function !!!',192,192,192,`')',`Exit()')][Font="Arial" /S11/B4]/W124 /H16/B1/D2/N}

```
{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][noframe][looping][share AVI]!arrow.avi}
```

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][noframe][looping][share AVI]!arrowg.avi}

TsPrint Functions

The **TsPrint Functions** overcome limitations of the Viewer Print Function. Viewer will not "wait" until printing is finished before submitting the next command or print., which makes printing more than one topic impossible.

{ewc TSTOOLS, Tsbutton,"More..."[Macro=Ji(qchPath,`PrintFunctions')][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsMacro Function

The **TsMacro Function** acts like a subroutine. It submits a string of macros of up to 512 characters without having to enter a topic or a group. This function effectively overcomes the 512 character limitation of a topic entry macro or a group macro.

{ewc TSTOOLS, Tsbutton,"Example"[Macro=TsMacro(`!tsdemo1.txt',`')][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsYesNo

The **TsYesNo Function** allows the author to create a custom message box. The author has control of the title and three lines of text as well as the color of the infobox and can submit a different command for each of the "Yes" or "No" answers

{ewc TSTOOLS, Tsbutton,"Example"[Macro=TsYN(1,`TsYesNo Example',`Do you want to print an order form',`for TouchSend Tools ?',` ',192,192,192,`JI(qchPath,`orderform')',`')][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsWinStyle

The **TsWinStyle Function** allows the author to easily change the window style of the main or a secondary window, such as removing minimize box, maximize box,caption, title bar.

{ewc TSTOOLS, Tsbutton,"Demo"[Macro=ji(`tstools.mvb>third',`tswindowdemo')]]
[Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}{ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()]
[Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsSaveBaggage Function

The **TsSaveBaggage Function** allows the author to extract a file from baggage. An example would be extracting an Excel worksheet and then launching Excel to load it.

{ewc TSTOOLS, Tsbutton,"Demo"[Macro=TsSaveBaggage(`order.txt',255,0,0)][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N} {ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsExitTopic Function

The **TsExitTopic Function** allows the author to execute a macro on exiting a topic. With certain limitations, this function works on a topic in the same way that the group exit script works upon exiting a group. Can be combined with **TsMacro**

{ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TsWave

The **TsWave Function** allows the author to submit a wave file that will survive jumps between topics. Normally Viewer will stop any wave file being played in the event of a topic jump.

```
{ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}
```

TsPane

TsPane allows the author to keep track of and display information in an embedded pane. It can be used to keep score, tracking progress etc.

{ewc TSTOOLS, TsPane, "demopane"[Name=calcdemo][Font="Times New Roman"/S12/B4][Text=`This is a variable number: ',`demopane',`.'][Color=255,255,128,255,0,0][Macro=TsInfobox(3,`TsPane Demo',`',` TsPane allows the author to send information',` to embedded panes in the title such as scoring or progress.',255,255,128,`')]/w170/h18}

Click here to increment count by 3
Click here to decrement count by 2

{ewc TSTOOLS, Tsbutton,"Return"[Macro=Back()][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

Using *TsTimer* for Auto/Manual Demos

This is the first of *six screens* in a *"simulated"* demo. It will run automatically unless the *"interactive"* button is selected. Then it will not jump until the browse button is pressed, or *"self running"* button is selected.

(let it run through once first !)

Create Award Winning Displays

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][looping][noframe][share AVI]!ribbon.avi}

This is the third Timer Demo Screen

Slide Shows/Kiosks

{ewc MVMCI2, ViewerMCI, [device AVIVideo][autostart][looping][noframe][share AVI]!glight.avi}

Screen5

TouchSend Consulting Services
will fast track your title development.

904-668-6180
Compuserve 76064, 3410

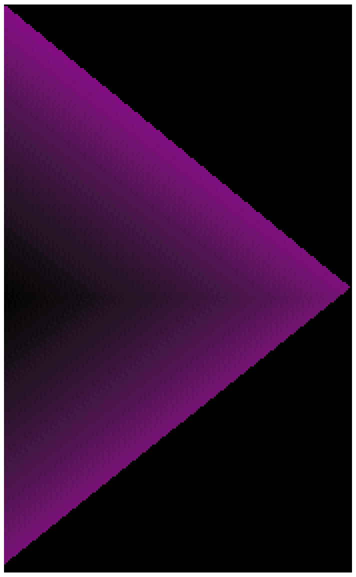
{ewc TSTOOLS, Tsbutton,"Quit Timer
Demo"[Macro=ji(qchPath,`FunctionMenu')][Font="Arial"/S11/B4]/W126/H16/B1/D1}{ewc TSTOOLS,
Tsbutton,"Restart as Interactive"[Macro=DeleteMark(`auto');ji(qchPath,`TsTimerDemo1')]
[Font="Arial"/S11/B4]/W126/H16/B1/D1}{ewc TSTOOLS, Tsbutton,"Restart as Self
Running"[Macro=ji(qchPath,`TsTimerDemo1');SaveMark(`auto')][Font="Arial"/S11/B4]/W126/H16/
B1/D1}

```
{ewc TSTOOLS, Tsbutton,"Change to
Interactive"[Macro=TsTimer(`',0);DeleteMark(`auto');ClosePane(`main',`demobutt');PaneID(qchPath,
`dbm>demobutt', 0)][Font="Arial"/S11/B4]/W126/H16/B1/D1}
```

```
{ewc TSTOOLS, Tsbutton,">>"[Macro=TsTimer(`',0);next()][Font="Arial"/S11/B4]/W126/H16/B1/D1}
{ewc TSTOOLS, Tsbutton,"Change to Self
Running"[Macro=TsTimer(`',0);SaveMark(`auto');ClosePane(`main',`demobutt');PaneID(qchPath,
`dba>demobutt', 0);Next()][Font="Arial"/S11/B4]/W126/H16/B1/D1}
```

Tsbutton

The **Tsbutton** is an embedded clickable button which when clicked, submits a command or string of commands to Viewer. The author has complete control over the **Placement, Size, Border, Highlighting, Font** and **Text** of the Button. Other **TouchSend Buttons** can display graphics and mixed text and graphics.



Pane 1 - TsMacro Demo

Pane 2 - TsMacro Demo

Pane 3 - TsMacro Demo

Pane 4 - TsMacro Demo

Pane 5 - TsMacro Demo

Pane 6 - TsMacro Demo

Pane 7 - TsMacro Demo

Pane 8 - TsMacro Demo

Pane 9 - TsMacro Demo

{ewc TSTOOLS, tsbutton,"Continue"[Macro=TsMacro(`!tsdemo2.txt',`')][Font="Arial"
/S11/B5/3-]W113/H18/B1/D2/N}

Pane 11-TsMacro Demo

Pane 12-TsMacro Demo

Pane 13-TsMacro Demo

Pane 14-TsMacro Demo

Pane 15-TsMacro Demo

Pane 16-TsMacro Demo

Pane 17-TsMacro Demo

Pane 18-TsMacro Demo

Pane 19-TsMacro Demo

```
{ewc TSTOOLS, Tsbutton,"Continue"[Macro=TsMacro(`!tsdemoa.txt',`');TsTimer(`TsMacro(`!
tsdemo4.txt',`');TsTimer(`TsMacro(`!tsdemo7.txt',`');TsTimer(`TsMacro(`!tsdemo8.txt',`');TsTimer(`TsMacro(`!
tsdemo9.txt',`');TsTimer(`TsMacro(`!tsdemo10.txt',`')',1)',1)',1)',1)',1)][Font="Arial"
/S11/B5/3-]W111/H24/B1/D2/N}
```

Jeff: this works

TsMacro is submitting commands in conjunction with TsTimer

Demo Pane Displayed
During TsMacro Demo

TsMacro Finale

You have just been party to the submission of 2266 characters to the Viewer Command processor from a single topic entry macro of 221 characters. TsMacro allows the author to create extensive subroutines which when used Group Entry Macros as well as the Topic Entry Macros, provide utmost flexibility as well as saving time and allowing for reusable code.

{ewc TSTOOLS, tsbutton,"Return"[Macro=JI(qchPath,`FunctionMenu2')][Font="Arial" /S15/B4/3-]W100/H36/B1/D2/N}

1. Stop Smoking...

2. Exercise more vigorously...

3. Lose weight...

TouchSend Tools order form

Phone	**904-668-6180**
Fax	**904-668-5352**
CompuServe	**73374,2071**
Address	**1904 Chatsworth Way, Tallahassee, Fl 32308**

Yes - Send me TouchSend Tools and/or TouchSend Index:

Name:	____________________
	____________________
Attention:	____________________
Address:	____________________
City:	____________________
State:	__________ **Postal Code:** __________
Phone:	____________________
Date :	____________________

I have enclosed a cheque for:

TouchSend Tools ____ copies x $279.00	__________
TouchSend Index ____ copies x $129.00	__________
Shipping and Handling	**$ 7.95**
Florida Residents add 7%	__________
Total	__________

___ Send Me More Information on Touchsend VbTools, The Visual Basic Version of TsTools

TouchSend is a registered Trademark of TouchSend Corporation used under licence by TouchSend Management Consulting Inc.

TouchSend Tools are comprised of the tools demonstrated here. TouchSend Index is a Hierarchical Index Listbox. The Index and the Tools are sold separately. Pricing for TouchSend tools is $279 per development machine, and $279 for each commercial title. Commercial titles that are updated require a further payment but not more than once per year. (For example, catalogs that are updated monthly pay no additional fees). Except for the title fee, there are no runtime charges and no royalties. There are no additional charges for inhouse use or for noncommerical titles. There are no additional charges for networks or wide area networks. Please include $7.95 for shipping and handling for each order. Florida residents add 7% sales tax.

If you see this message, your computer did not correctly display the video on the opening screen.   Press the continue button to proceed.

{ewc TSTOOLS, tsbutton,"Continue"[Macro=ji(qchPath,`tsfull')][Font="Arial" /S11/B4/3-]W62/H16/B1/D2/N}

{ewc TSTOOLS, tsbutton,"Print"[Macro=Print()][Font="Arial" /S12/B4/3-]W100/H17/B1/D2/N/P-}
{ewc TSTOOLS, tsbutton,"Close"[Macro=CloseWindow(`Touchsnd')][Font="Arial" /S12/B4/3-]W100/H17/B1/D2/N/P-}

TouchSend Tools Order Form

TouchSend Tools are comprised of the tools demonstrated here. TouchSend Index is a Hierarchical Index Listbox. The Index and the Tools are sold separately. Pricing for TouchSend tools is $279 per development machine, and $279 for each commercial title. Commercial titles that are updated require a further payment but not more than once per year. (For example, catalogs that are updated monthly pay no additional fees). Except for the title fee, there are no runtime charges and no royalties. There are no additional charges for inhouse use or for noncommerical titles. There are no additional charges for networks or wide area networks. Please include $7.95 for shipping and handling for each order. Florida residents add 7% sales tax.

Yes - Send me TouchSend Tools and/or TouchSend Index:

Name:	______________________________
	______________________________
Attention:	______________________________
Address:	______________________________
City:	______________________________
State:	________ **Postal Code:** ________
Phone:	______________________________
Date :	______________________________

I have enclosed

TouchSend Tools	____ **copies x $279.00**	________
TouchSend Index	____ **copies x $129.00**	________
Shipping and Handling		**$ 7.95**
Florida Residents add 7%		________
	Total	________

___ Send Me More Information on Touchsend VbTools, The Visual Basic Version of TsTools

Phone	**904-668-6180**
Fax	**904-668-5352**
Address	**1904 Chatsworth Way, Tallahassee, Fl 32308**
CompuServe	**73374,2071**

TouchSend® is a registered trademark of TouchSend Corporation used under licence by TouchSend Management Consulting Inc. TouchSend Tools are provided in a dynamic link library running under Windows 3.1 or greater.

TsMacro Function

The **TsMacro Function** acts like a subroutine.   It submits a string of macros of up to 512 characters without having to enter a topic or a group.   This function effectively overcomes the 512 character limitation of a topic entry macro or a group macro.

TsPrintAfterJump	jumps to a topic, prints the topic and then submits another command to Viewer after the printing is complete.
TsPrintFromList	prints a list of topics. The list can be in baggage or on the disk. Each topic is submitted to the printer, and then upon completion, jumping to the next topic.
TsPrintGroup	prints all of the topics in a Viewer defined group.
TsPrintThenCmd	prints the current screen and then submits a command to Viewer after the printing is complete..

{ewc TSTOOLS, Tsbutton,"Return"[Macro=JI(qchPath,`FunctionMenu2')][Font="Arial" /S11/B4]/W62 /H16/B1/D1/N}

The following Screen was initially the opening screen for this demo of the TouchSend Tools. It opens 4 panes one at a time using the TsTimer Command and then plays a video using the TsMCI Command and jumps to what is now the opening screen. These are capabilities that are not in "vanilla Viewer".

It was a very exciting opening but the video was built as an RLE encoded video which was crashing on all sorts of ATI cards. It is now in Video1 but to play it safe it is now optional. If your machine crashes, don't despair, just start up again and avoid this button.

TsMCI & TsTimer Demo
Skip Demo: Return to Main Screen

```
{ewc TSTOOLS, TsPane, "status1"[text=`status1'][graphic=`!status.dib'][macro=TsVCopyS(`status1',`You
clicked~the status bar !!!');TsTimer(`TsVCopyS(`status1',`TouchSend~Status Bar~Cleared
');TsTimer(`TsVCopyS(`status1',`A TsPane~Status Bar')',3)',3)][Color=192,192,192,128,0,0]
[Font="Arial" /S11/B5/]/H60/w115/B1/ML6T10/D2/2}
```

TsWinStyle Demo

To properly use this function in a Main Window, make sure the Menu and Buttonbar are turned off or this technique is unreliable.   Tsbuttons can be used in place if the Viewer Buttonbar

{ewc TSTOOLS, Tsbutton, "Thick Frame" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,0,0,0,0,0,1)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "Caption Only " [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,1,0,0,0,0,1)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "No System Menu" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,1,1,0,1,1,0)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "Thin Frame" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,0,0,0,0,0,0)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "No Styles" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,0,0,0,0,0,0)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "Min/Max Off" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,1,1,1,0,0,0)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "All Styles On" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,1,1,1,1,1,0)]/H16/w100/B2/D1/N}

{ewc TSTOOLS, Tsbutton, "Close this Window" [Font="Times New Roman"/S11/B3/3-] [Macro = closewindow(`third')]/H16/W100/B2/D1/N}

Ts Buttons - Graphic Options

TsButtons Graphic Options allow the display of three bitmaps: up, down and disabled.   Click the disable switch button then look at the Yellow Switch.

{ewc TSTOOLS,Tsbutton,"Disable Switch"[Macro=TsDisableButton(`OnOff');MasterNSRColor(`main', 128,0,0)][Font="Arial" /S11/B4]/W84/H20/B1/D2/N}{ewc TSTOOLS,Tsbutton,"Enable Switch"[Macro=TsEnableButton(`OnOff');MasterNSRColor(`main', 128,128,0)][Font="Arial" /S11/B4]/W84/H20/B1/D2/N}
{ewc TSTOOLS,Tsbutton,"Close "[Macro=ClosePane(`main',`graphbut');PaneID(qchPath,`switchoff>swonoff',0);MasterNSRColor(`main', 0,0,0);TsEnableButton(`OnOff')][Font="Arial" /S11/B4]/W168/H20/B1/D2/N}

The Commercial Version

In the commercial version of TsTools, the author can cause a button "click event" from an accelerator key. To activate the "Medium" button without using the mouse, try using the "Alt M" key.

```
{ewc TSTOOLS,Tsbutton,"Close"[Macro=ClosePane(`main',`graphbut')][Font="Arial"
/S11/B4/3-]/W100/H18/B1/D2/N}
```

{ewc TSTOOLS, tsbutton,"Return to Menu"[Macro=ji(qchPath,`functionmenu1')][Font="Arial"
/S11/B4/3-]W100/H36/B1/D2/N}

{ewc TSTOOLS, TsButton, "The TouchSend Index Demo"[graphic=`!tsrrup.dib',`!tsrrdn.dib',`!tsrrdi.dib'/AL][macro=JumpID(`tstools>indexdem', `indexdem')] [Font="Times New Roman" /S9/B4]/H20/w140/B1/D1/AR/MR5} {ewc TSTOOLS, TsButton, "Return"[graphic=`!tsrrup.dib',`!tsrrdn.dib',`!tsrrdi.dib'/AL][macro=MasterSRColor(`main', 0,0,0);CloseWindow(`indexdem');JumpID(`tstools>main', `tsfull')] [Font="Times New Roman" /S9/B4]/H20/w140/B1/D1/AR/MR5}

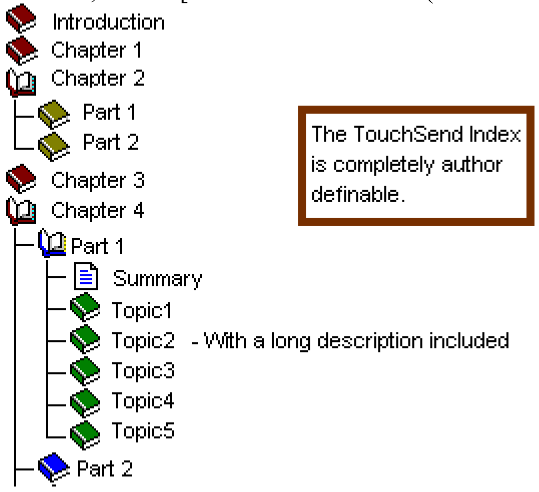
The TouchSend Index

The Index comes with the *TouchSend Builder*. The *TouchSend Builder* will parse all of a project's rich text format files as well as the compiled title and then create the table lookups necessary to build and store the index for a title. The index is completely author definable. Press the The TouchSend Index Demo Button for an example of the *TouchSend Index.*

No title should be without one.

TouchSend Index is a Hierarchical Index Listbox. The Index and the Tools are sold separately. There are no runtime charges. **For Commercial use,** each copy of TouchSend Tools and TouchSend Index is licenced for a single title. For Publishers with significant numbers of titles, volume discounts are available. **For Non Commercial Use**, each copy of TouchSend Tools and TouchSend Index is licenced to a authoring/development single machine (just like your wordprocessor etc.) but the DLL can be used for any number of non commercial titles and inhouse publications with no further charges.

```
{ewc TSTOOLS, tsbutton,"Search"[Macro=PopupID(qchPath, `indexmsg')][Font="Arial"
/S11/B4/3-]W100/H16/B1/D2/N}{ewc TSTOOLS, tsbutton,"Sync"[Macro=PopupID(qchPath, `indexmsg1')]
[Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N}{ewc TSTOOLS,
tsbutton,"Close"[Macro=CloseWindow(`indexdem')][Font="Arial" /S11/B4/3-]W100/H16/B1/D2/N}
```



- Introduction
- Chapter 1
- Chapter 2
  - Part 1
  - Part 2
- Chapter 3
- Chapter 4
  - Part 1
    - Summary
    - Topic1
    - Topic2 - With a long description included
    - Topic3
    - Topic4
    - Topic5
  - Part 2

The TouchSend Index is completely author definable.

TouchSend Consulting Services
904-668-6180   Compuserve 76064,3410

TouchSend specializes in the following:

1. Training developers/authors in innovative ways to use Viewer 2.0 to create satisfying titles, including electronic publishing, textbooks, catalogues, interactive demonstrations, tutorials, corporate policies & procedures, statutes & regulations, and educational material.
2. Developing migration and conversion strategies for existing hard copy and electronic media.   For example, converting existing textbooks from desktop publishing format to Viewer format on an automated cost effective basis or converting banking/insurance corporate policy/procedure manuals to electronic format.
3. Designing databases that will automatically create, maintain and update material as well as autogenerate and update hypertext links and jumps.
4. Custom programming of Viewer and Winhelp extensions (such as the TouchSend Tools) and interfacing of Viewer data with other Applications (both Windows and non-Windows).
5. Developing corporate strategies for electronic publishing, electronic sales and presentation tools, as well as kiosks, electronic catalogues and CD Rom Development.

{ewc TSTOOLS, tsbutton,"Print"[Macro=Print()][Font="Arial" /S11/B4/3-]W100/H18/B1/D2/N/P-}   {ewc TSTOOLS, tsbutton,"Close"[Macro=CloseWindow(`Touchsnd')] [Font="Arial" /S11/B4/3-]W100/H18/B1/D2/N/P-}

LAST
{ewc TSTOOLS, Tsbutton,"Demo"[Macro=][Font="Arial" /S11/B4]/W62 /H16/B1/D2/N}

TouchSend Tools

Functions included in TstoolsW.dll:
(licenced for noncommercial use only) - click each for documentation

Note: TsToolsW.mvb and TsTools.mvb cannot be run concurrently. Also remember to use **rr.txt**

Order information

TouchSend Write Functions

Write Functions included in TstoolsW.dll:

(licenced for noncommercial use only) - click each for documentation

Licence & Warranty

TsToolsW.DLL is licenced only for noncommerical use. **Use in any commercial software of any kind is expressly prohibited.**

There is no warranty express or implied by or TouchSend Management Consulting Inc. with respect to this software or any of its contents. The entire and exclusive liability of any nature whatsoever arising from or in any way related to use of this software shall be limited to a refund of price paid for this software, and shall not include or extend to any claim for or right to recover any other damages, including but not limited to loss of profit, loss of any claim , or special, incidental or consequential damages or other similar claims.

ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

This version of TsToolsW.DLL is not officially supported. Use it at your own risk. However if you have questions, please feel free to send email to Compuserve 73374,2071 and we will do the best we can to help you out.

{ewc TSTOOLS, Tsbutton,"Return"[Macro=MasterSRColor(`main', 0,0,0);CloseWindow(`returnrm');Back()] [Font="Arial" /S12/B4/3-] W100/H32/B1/D2/N}

TsTools Demo Credits:

TsTools Design & TsTools Demo	Jeff Kovitz	904-668-6180
Programming	David Stidolph	
Voice	John Summers	
Voice Audio	Green Vine Media	904-574-3400
Animation	Jeff Strickland	904-421-1878
Visual Basic	JLK Technology	904-893-8469

You pressed Yes !

You pressed No !

```
{ewc TSTOOLS, Tsbutton, "Hug Me" [graphic=`!babyup.dib',`!babydn.dib',`!babyup.dib'][name=hugme]
[macro=tshidebutton(`hugme',0,0,0)][Font="Times New Roman" /S10/B4/I]/B0/D0/MT60}
```

Ts Tools ReadMe

{ewc TSTOOLS, Tsbutton,"Print"[Macro=Print()][Font="Arial" /S12/B5] /W80 /H20/B2/D1}{ewc TSTOOLS, Tsbutton,"Return"[Macro=MasterSRColor(`main', 0,0,0);JI(qchPath,`TSFull')][Font="Arial" /S12/B5] /W80 /H20/B2/D1}

The TouchSend Tools directory on this CD ROM consists of the following:

tstools.mvb	This demo of TouchSend Tools
tstools.dll	The DLL to run the demo (it will only run with the demo)
tshelp.hlp	A help file used in the demo
tshelp1.hlp	A help file used in the demo
tstoolsw.mvb	The help and how to Viewer Title for the tstoolsw.dll
tstoolsw.dll	The TouchSend Tools working sampler DLL provided with this Book.
readme.txt	A DOS text version of this screen.
rr.txt	The function declarations to be inserted in the configuration section of the MVP file for the TsToolsW.DLL functions. Because these declarations are so "fussy" *ie "i" is not the same as "I" and "u" is not the same as "U",* it is recommended that the entire file be pasted into the config section even if you aren't using some of the functions. It can save hours of frustration trying to figure out why something doesn't work. Click here to view rr.txt
order.txt	A DOS text order form for TsTools. Click here to view order.txt

==

Last minute notes:

We have also noted that if two buttons are created that autosize to text and one has lower case and one has upper and lower case, they will not be the same height. The fix (it will be fixed in the next build of the DLL) for TsToolsW.DLL is to ensure that all buttons are the same case mix or to author specify the width and height.

We have also found that running in 1024x768 or 800x600 with large fonts caused the menu buttons on the opening screen to be pushed out of view. Please change to normal font or 640x480.

==

The tstoolsw.dll enclosed is a full working version of the following functions:

TsToolsInit	must be the first function called in the configuration script of the MVP file ***or none of the other functions will work***. TsToolsInit initializes the TsToolsW.DLL and creates an ini file in the Windows directory that is used by TsToolsW.DLL. The author controls the name of the ini file. It is **strongly** recommended that each title have a different ini name.
TsAbsolute	Viewer device-independent measurements map position values into a 1,024-by-1,024 grid. At run time, these measurements are converted to device-specific coordinates using a scaling ratio appropriate to the display device. For example, in 640-by-480 video mode, a device-independent measurement of 512 (specified for the X or Width parameters) would be converted to a pixel measurement of 320 (512 x (640/1024)). For the Y and Height parameters, the same value would produce a pixel measurement of 240 (512 x (480/1024)). **TsAbsolute(0)** will cause TsButtons to be mapped to a device independent 1024x1024 grid. **TsAbsolute(1)** [the default if not called] will ensure that whatever pixel coordinates are specified will be reproduced on the screen.
TsCopyString	copies a specified string to the clipboard.
TsHelpContext	calls context sensitive help from Viewer.
TsInfoBox	creates a 3 line "about" or dialog box. The author has control over the text in the

	title bar, 3 lines of text in the dialog box and the color of the dialog box.
TsSaveBaggage	copies a specified file from baggage to a file on the hard disk. Storing files in baggage is extremely useful and this function provides a way to retrieve stored files. By making careful use of this command together with marks, an author can extract a file that is appropriate for the current user. As an example, if the author wants to illustrate a point using a spreadsheet file, the function can extract a Lotus, Excel or Quattro Pro file dependent on the user's selected spreadsheet preference and the others never clutter the users disk or create conversion/confusion problems.
TsWrite	calls Write.exe and using the other Write functions, allows the author to manipulate write from Viewer.
TsWriteCopy	calls Write.exe and allows the user copie highlighted material to the clipboard.
TsWriteKill	will close Write after opened with TsWrite.
TsWritePaste	will paste the contents of the clipboard into Write
TsWritePos	will set the screen position of Write after it has been opened using TsWrite
TsWriteSetZ	will set the screen "z order" of Write after opened with TsWrite.
TsYN	creates an about box with yes and no buttons. It has all of the same capabilities as TsInfoBox but as well will submit a command to Viewer based on the button pressed.

Note: Clicking on each function name will cause a jump to the syntax screen for that function.

The TsWrite function set is a Write.exe specific group of functions that emulates the TsExec functions (which do the same thing for virtually any Windows application) but limited to work just with Write. The TsWrite functions illustrate one of the challenges of hypetext multimedia delivery systems, namely: ***once an author hands data to a user: he or she wants to be able to do something with it***. ***These tools and the TsExec functions are designed to fulfill those needs.***

Windows allows an author to manage and use multiple applications to manage and manipulate information. This will become more apparent and even easier with the release of Windows 4.0 (now code named "Chicago" by Microsoft).

The TsToolsW.DLL also includes a subset of the functions available in the commercial version of the TsButton embedded pane.

TsButton allows the author to create buttons in embedded panes in Viewer. This tool provides the author with ultimate flexibility in creating exciting titles with a custom look and feel.

The following TsButton capabilites are provided in TSTOOLSW.DLL:

autosizing to text
use of any font available on the users system at any allowable pixel size
control over height and width
control over border and depth
` *control of italics, underline, bold and overstrike text display*
control over text alignment and margins
notching
angled text
xy start location of text
attaching Viewer command strings to any button

The following functions available in the commercial version of TsButton are not included in TSTOOLSW.DLL:

the ability to disable the button
3 dimensional looking fonts
the ability to inhibit printing of the button

the ability to hide/restore the button
multiline text buttons
the ability to place graphics on the button
the ability to click the button by function call.
mouse enter and mouse leave events.

A full description of TsTools can be found by selecting the TsToolsW Online section of this title, and then selecting "TsTools Commercial Version". All of the functions (at press time) are fully described. We did not repeat any of the information relating to TsButtons which are clearly documented under the heading "TsButtons"

Note: If you don't have time to click through the entire demo, please make sure you select the "don't miss" button on the button bar in the TsToolsW Online section.

Special note regarding TsButtons

Viewer creates embedded panes by looking for {ewx....} as a structure in a rich text format file (where the x is either an "l","c" or an "r"). If there is an extra curly bracket in the line, usually Viewer will crash. Don't despair. Also because of the myriad of authoring control provided with TsButton, it is easy to make a mistake in the switch syntax. TouchSend has put extensive error checking in to deal with those issues, but occasionally Viewer will crash if your syntax is wrong, or alternatvely the button won't draw as you have intended. Check your button syntax carefully (not always as easy to do as to say !!).

The commercial version of TsTools has an authoring tool to create buttons and manage the syntax.

If you have any questions we will be happy to answer them, but only via Compuserve 73374,2071.

Orders for the commercial version of TsTools can be placed via Compuserve or via phone 904-668-6180 or by fax at 904-668-5352. Payment must be by cheque in advance of delivery.
to TouchSend Tools c/o 1904 Chatsworth Way Tallahassee, Florida 32308. The TsTools set offer through this book also contains the source files for this demo (not the bitmaps or avi's though) which exhaustively show the techniques used to create this demo. *Make sure you mention that you want the Waite Group version to ensure that the source code is sent.* In order to examine the mvp file used to create this demo, click here.

Pricing for TouchSend tools is $279 per development machine, and $279 for each commercial title. Commercial titles that are updated require a further payment but not more than once per year. Except for the title fee, there are no runtime charges and no royalties. There are no additional charges for inhouse use or for noncommerical titles. There are no additional charges for networks or wide area networks. Please include $7.95 for shipping and handling for each order. Florida residents add 7% sales tax.

Click here for licence information and warranty limitations.

```
{ewc TSTOOLS,TsButton,"Return to ~Readme.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returnrm]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=ji(qchPath,`rrreturn');CloseWindow(`returnrm')][Font="Times New Roman"/S13/B4/3-]}
```

```
{ewc TSTOOLS,TsButton,"Return to ~Readme.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returnrm]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=ji(qchPath,`returntm');CloseWindow(`returnrm')][Font="Times New Roman"/S13/B4/3-]}
```

```
{ewc TSTOOLS,TsButton,"Return to ~Readme.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returnrm]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=ji(qchPath,`buttonrtn2');CloseWindow(`returnrm')][Font="Times New Roman"/S13/B4/3-]}
```

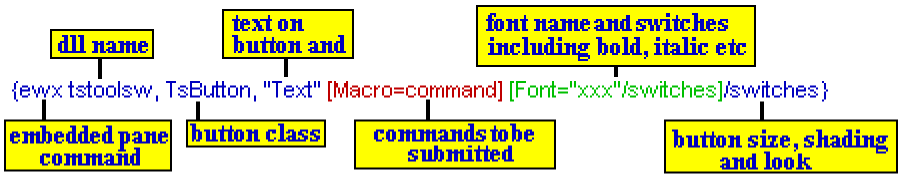
```
{ewc TSTOOLS,TsButton,"Return to ~Readme.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returnrm]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=ji(qchPath,`buttonrtn1');CloseWindow(`returnrm')][Font="Times New Roman"/S13/B4/3-]}
```

```
{ewc TSTOOLS,TsButton,"Return to ~Readme.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returnrm]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=ji(qchPath,`buttonrtn0');CloseWindow(`returnrm')][Font="Times New Roman"/S13/B4/3-]}
```

TsButton

Purpose: Displays a clickable button in an embedded pane. The author has control of the text, size, font, and shading of the button. On clicking the button, it will submit a command or string of commands to Viewer.

Syntax:


dll name
text on button and
font name and switches including bold, italic etc
{ewx tstoolsw, TsButton, "Text" [Macro=command] [Font="xxx"/switches]/switches}
embedded pane command
button class
commands to be submitted
button size, shading and look

This is a typical button:
{ewx TSTOOLS, Tsbutton, "Clic&k Me"[Name=clickme] [Macro=TsInfoBox(2,`TouchSend Textbuttons',` This Dialog Box was created using TsInfoBox.',` This command was executed from a TouchSend Text Button.',` Press each of the buttons on the right for lots of info.',0,255,0,`')] [Font="Arial" /S11/B4]/H24/W100/B1/D2}

Which results in*** :

{ewc TSTOOLS, Tsbutton, "Clic&k Me"[Name=clickme][Macro=TsInfoBox(2,`TouchSend Textbuttons',`This Dialog Box was created using TsInfoBox.',`This command was executed from a TouchSend Text Button.',`Click the buttons to the right for more information.',0,255,0,`')][Font="Arial" /S11/B4]/H24/w100/B1/D2}

Note that the "click me" button can also be
activated by the "Alt K" key by making use of
the **TsPressButton** Function which is attached
to the "Alt K" key with the AddAccelerator() function.

Tsbutton: Embedded Pane

The embedded pane statement has the following syntax:

{ewX DLL-name, window-class, author-data}

The **{ewc}** statement positions the object as if it were the next character in the line, aligning it on the base line and applying the current paragraph properties. The **{ewl}** statement positions the object at the left paragraph margin. The left paragraph indent specifies where the object sits in relation to the border of the master pane, regular pane, or popup. The **{ewr}** statement positions the object at the right paragraph margin. The right paragraph indent specifies where the object sits in relation to the border of the master pane, regular pane, or popup.

Note: The maximum length of this statement (including braces) is 1,023 characters.

Tsbutton DLL Name and   Button Class

The embedded pane statement has the following syntax:

{ewx **DLL Name**, **Button Class**,commands and parameters}

DLL Name:

Purpose: Specifies the name of the DLL that controls the embedded pane. The filename should not include an extension or be fully-qualified, but it can include a relative path. Viewer assumes .DLL or .EXE to be the default extension.   In this case the name is "**TsTools**"

Button Class:

Purpose: The button class is the name of the button called from the DLL.   In this case the button class is "**TsButton"**.

TsButton: Commands

Commands	are strings specifying a command or commands to run if the condition is true/false. To specify multiple commands, insert a semicolon (;) between each command. If the command(s) contain string parameters, use single open quotes (`) and close quotes (') to delimit the string parameters. If the commands contain paths, use double backslashes (\\) or single forward slashes (/) to represent each backslash in the path. The total length of a Viewer command string cannot exceed 512 characters unless TsMacro is used.

Why Pixels for Fonts and not Points ?

Points and pixels approximately the same but not quite.
All TsButtons are created by using pixel measurements.

By using pixels for font size, the author has **exact** control over the look of the button and the size of the text relative to the size of the button.

TsButton: Font Properties

Switches for the Font Properties are:

"Font Name *optionally using & and ~***"/S#/B#/D#/I#/U#/O#/A#/** where

The **Font Name** must be in quotes and must be found on the system.   If the font is not found the font will be helvetica 8 point.   Using the & in the text string will underline the next character.   Multiline text and 3D effects are not included in the TsToolsW.DLL.

S	is the font size in <u>pixels</u>
B	is the font weight (0-9) - with 4 being normal *(some fonts only support weights of 4 and 7)*
I	Italics on (1) or off - no entry or (0)
U	Underline on (1) or off - no entry or (0)
O	Overstrike on (1) or off - no entry or (0)
A	is for angle in 10ths of a degree with 0 straight right, 900 being vertical

See: X and Y switches in button properties re text starting location with angle switch

See: Attribute Examples (buttons on right) for samples.

TsButton: Font Attribute Examples

3d effects are not included in TsToolsW.DLL included with this book.

{ewc TSTOOLS, Tsbutton, "Bold 3" [Font="Arial"/S11/B3] /H30/w80/B2/D2} {ewc TSTOOLS, Tsbutton, "Bold 5" [Font="Arial"/S11/B5] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Bold 7" [Font="Arial"/S11/B7] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Bold 9" [Font="Arial"/S11/B9] /H30/W80/B2/D2}

{ewc TSTOOLS, Tsbutton, "3d Raised" [Font="Arial"/S11/B3/3+]/H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Raised" [Font="Arial"/S11/B5/3+] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Raised" [Font="Arial"/S11/B7/3+]/H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Raised" [Font="Arial"/S11/B9/3+]/H30/W80/B2/D2}

{ewc TSTOOLS, Tsbutton, "3d Inset" [Font="Arial"/S11/B3/3-] /H30/w80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Inset" [Font="Arial"/S11/B5/3-] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Inset" [Font="Arial"/S11/B7/3-] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "3d Inset" [Font="Arial"/S11/B9/3-] /H30/W80/B2/D2}

{ewc TSTOOLS, Tsbutton, "Ital Bold 3" [Font="Arial"/S11/B1/I1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Ital Bold 5" [Font="Arial"/S11/B5/I1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Ital Bold 7" [Font="Arial"/S11/B7/I1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Ital Bold 9" [Font="Arial"/S11/B9/I1] /H30/W80/B2/D2}

{ewc TSTOOLS, Tsbutton, "Underline" [Font="Arial"/S11/B4/U1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Overstrike"[Font="Arial"/S11/B5/O1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Underline" [Font="Arial"/S11/B7/U1/I1] /H30/W80/B2/D2} {ewc TSTOOLS, Tsbutton, "Overstrike" [Font="Arial"/S11/B9/O1/I1] /H30/W80/B2/D2}

{ewc TSTOOLS, Tsbutton, "Angle 450" [Font="Arial"/S11/B3/A450] /H80/W80/x20/y40/B2/N/D2} {ewc TSTOOLS, Tsbutton, "Angle 900" [Font="Arial"/S11/B5/A900] /H80/W80/x34/y60/B2/N/D2} {ewc TSTOOLS, Tsbutton, "Angle 1350" [Font="Arial"/S11/B7/a1350] /H80/W80/x60/y60/B2/N/D2} {ewc TSTOOLS, Tsbutton, "Angle 1800" [Font="Arial"/S11/B9/A1800] /H80/W80/x70/y30/B2/N/D2}

TsButton: Button Properties

Button properties are controlled by the following switches:

/W#/H#/B#????/**D#/N/M**L#T#R#B#**/X#/Y#/A**HV where

W	Button width in pixels	*See Width Examples*
H	Button height in pixels	*See Height Examples*
B	Border width in pixels	*See Border Examples*
	where ???? controls which borders are drawn. All borders will be drawn unless any of **T(**Top);**B(**Bottom);**L(**Left); **R(**Right) are specified.	
D	Depth *(highlight)* width	*See Depth Examples*
N	Notched corners	*See Notch Examples*
M	Specifies the margin from the outside of the border to the start of text. ...only specify the ones that are required.	*See Margin Examples*
XY	Starting X & Y position for text in the button	*See XY Examples*
A	**H**orizontal or **V**ertical alignment of text	*See Alignment examples*

Note: - if no switches are specifed, the button wil autosize to and center the text.

{ewc TSTOOLS, Tsbutton, "   X Y Examples"/AL [Font="Arial"/S11/B4/I1] [Macro = ji(`tstools.mvb>second',`tsbutlookxy')/H16/W110/B1TRBL/N/D2}

TsButton: Width Examples

Width Examples:

Width 11	{ewc TSTOOLS, Tsbutton, "+" [Font="Arial"/S11/B4/I1] /H16/W11/B1/N/D2}
Width 20	{ewc TSTOOLS, Tsbutton, "+" [Font="Arial"/S11/B4/I1] /H16/w20/B1/N/D2}
Width 30	{ewc TSTOOLS, Tsbutton, "30" [Font="Arial"/S11/B4/I1] /H16/w30/B1/N/D2}
Width 31	{ewc TSTOOLS, Tsbutton, "31" [Font="Arial"/S11/B4/I1] /H16/w31/B1/N/D2}
Width 32	{ewc TSTOOLS, Tsbutton, "32" [Font="Arial"/S11/B4/I1] /H16/w32/B1/N/D2}
Width 100	{ewc TSTOOLS, Tsbutton, "Width 100" [Font="Arial"/S11/B4/I1] /H16/w100/B1TRBL/N/D2}
Width 200	{ewc TSTOOLS, Tsbutton, "Width 200" [Font="Arial"/S11/B4/I1] /H16/w200/B1/N/D2}
Width 400	{ewc TSTOOLS, Tsbutton, "Width 400" [Font="Arial"/S11/B4/I1]/H16/w400/B1/N/D2}

Width 50 - Height 50 - Border 2 - Depth 3 Notched

{ewc TSTOOLS, Tsbutton, "50x50" [Font="Times New Roman"/S11/B5] /H50/w50/B2/N/D3}

TsButton:Height Examples

Height Examples:

Height 70 {ewc TSTOOLS, Tsbutton, "Height 70" [Font="Arial"/S11/B4/A900][Macro = pi(qchPath,`buttonh70>button')]/H70/W20/x1/y55/B1/N/D2} Height 200 {ewc TSTOOLS, Tsbutton, "200" [Font="Courier New"/S50/B4] /H200/W150/B1/N/D2}

Note that the text in the Height 70 example is vertical using the /A900 parameter. (Click the button...) The syntax of that button is:

{ewx TSTOOLS, Tsbutton, "Height 70" [Font="Arial"/S11/B4/A900] /H70/W20/x1/y55/B1/N/D2}

TsButton: Border Examples

Border Examples *(All using a Depth of /D1)*:

Border 0 {ewc TSTOOLS, Tsbutton, "Border 0" [Font="Times New Roman"/S11/B5] /H20/w60/B0/D1} Border 1 {ewc TSTOOLS, Tsbutton, "Border 1" [Font="Times New Roman"/S11/B5] /H20/w60/B1/D1}
Border 2 {ewc TSTOOLS, Tsbutton, "Border 2" [Font="Times New Roman"/S11/B5] /H20/w60/B2/D1} Border 3 {ewc TSTOOLS, Tsbutton, "Border 3" [Font="Times New Roman"/S11/B5] /H20/w60/B3/D1}
Border 4 {ewc TSTOOLS, Tsbutton, "Border 4" [Font="Times New Roman"/S11/B5] /H20/w60/B4/D1} Border 5 {ewc TSTOOLS, Tsbutton, "Border 5" [Font="Times New Roman"/S9/B4] /H20/w60/B5/D1}

Controlling Border Sides & Notch *(All using /B2)*

{ewc TSTOOLS, Tsbutton, "Left On" [Font="Times New Roman"/S11/B5] /H25/w60/B2L/D1} {ewc TSTOOLS, Tsbutton, "Right On" [Font="Times New Roman"/S11/B5] /H25/w60/B2R/D1} {ewc TSTOOLS, Tsbutton, "Top On" [Font="Times New Roman"/S11/B5] /H25/w60/B2T/D1} {ewc TSTOOLS, Tsbutton, "Bottom On" [Font="Times New Roman"/S11/B5] /H25/w60/B2B/D1} {ewc TSTOOLS, Tsbutton, "Notch On" [Font="Times New Roman"/S11/B5] /H25/w60/B2/N/D1}

Rule: A border will be assumed to be on all sides, but if you specify any one of left, right, top, bottom, you must specify <u>all</u> the parameters you want on.

Creating a Button Bar *(No L & R on every 2nd Button & /D2)*

{ewc TSTOOLS, Tsbutton, "You can" [Font="Times New Roman"/S11/B5] [Macro = pi(qchPath,`buttonbar')]/H20/W75/B2LRTB/D2} {ewc TSTOOLS, Tsbutton, "make a button" [Font="Times New Roman"/S11/B5] [Macro = pi(qchPath,`buttonbar')]/H20/W75/B2TB/D2}{ewc TSTOOLS, Tsbutton, "bar in a " [Font="Times New Roman"/S11/B5] [Macro = pi(qchPath,`buttonbar')]/H20/W75/B2LRTB/D2}{ewc TSTOOLS, Tsbutton, "non scrolling" [Font="Times New Roman"/S11/B5] [Macro = pi(qchPath,`buttonbar')]/H20/W75/B2TB/D2}{ewc TSTOOLS, Tsbutton, "region" [Font="Times New Roman"/S11/B5] [Macro = pi(qchPath,`buttonbar')]/H20/W75/B2/D2}

TsButton: Depth (Highlight) Examples

Depth (Highlight) Examples *(All using a border of /B1)*:

Depth 0 {ewc TSTOOLS, Tsbutton, "Depth 0" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D0} Depth 1 {ewc TSTOOLS, Tsbutton, "Depth 1" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D1}
Depth 2 {ewc TSTOOLS, Tsbutton, "Depth 2" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D2} Depth 3 {ewc TSTOOLS, Tsbutton, "Depth 3" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D3}
Depth 4 {ewc TSTOOLS, Tsbutton, "Depth 4" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D4} Depth 5 {ewc TSTOOLS, Tsbutton, "Depth 5" [Font="Times New Roman"/S11/B5] /H30/w60/B1LRTB/D5}

Depth (Highlight) Examples *(with corresponding borders)*:

Depth 0 {ewc TSTOOLS, Tsbutton, "Depth 0" [Font="Times New Roman"/S11/B5] /H30/w60/B0/D0} Depth 1 {ewc TSTOOLS, Tsbutton, "Depth 1" [Font="Times New Roman"/S11/B5] /H30/w60/B1/D1}
Depth 2 {ewc TSTOOLS, Tsbutton, "Depth 2" [Font="Times New Roman"/S11/B5] /H30/w60/B2/D2} Depth 3 {ewc TSTOOLS, Tsbutton, "Depth 3" [Font="Times New Roman"/S11/B5] /H30/w60/B3/D3}
Depth 4 {ewc TSTOOLS, Tsbutton, "Depth 4" [Font="Times New Roman"/S11/B5] /H30/w60/B4/D4} Depth 5 {ewc TSTOOLS, Tsbutton, "Depth 5" [Font="Times New Roman"/S9/B4] /H30/w60/B5/D5}

TsButton: Notch Examples

Notch Off Example
{ewc TSTOOLS, Tsbutton, "Depth 0" [Font="Times New Roman"/S11/B5] /H30/w60/B2/D2}
Notch On Example
{ewc TSTOOLS, Tsbutton, "Depth 1" [Font="Times New Roman"/S11/B5] /H30/w60/B2/N/D2}

Notching Samples with equal Depth and Borders

Depth 0 {ewc TSTOOLS, Tsbutton, "Depth 0" [Font="Times New Roman"/S11/B5] /H30/w60/B0/N/D0} Depth 1 {ewc TSTOOLS, Tsbutton, "Depth 1" [Font="Times New Roman"/S11/B5] /H30/w60/B1/N/D1}
Depth 2 {ewc TSTOOLS, Tsbutton, "Depth 2" [Font="Times New Roman"/S11/B5] /H30/w60/B2/N/D2} Depth 3 {ewc TSTOOLS, Tsbutton, "Depth 3" [Font="Times New Roman"/S11/B5] /H30/w60/B3/N/D3}
Depth 4 {ewc TSTOOLS, Tsbutton, "Depth 4" [Font="Times New Roman"/S11/B5] /H30/w60/B4/N/D4} Depth 5 {ewc TSTOOLS, Tsbutton, "Depth 5" [Font="Times New Roman"/S11/B5] /H30/w60/B5LRTB/N/D5}

TsButton: Disabled Text Examples

Disabled (Greyed) Text Examples:

{ewc TSTOOLS, Tsbutton, "Button1" [Name=Button1][Font="Arial"/S11/B4] /H16/w80/B1/D2/-}{ewc TSTOOLS, Tsbutton, "Button2" [Font="Arial"/S11/B4] /H16/w80/B1/D2/-}{ewc TSTOOLS, Tsbutton, "Button3" [Font="Arial"/S11/B4] /H16/w160/B1/D2/-}

Click here to enable Button1.   Click here to disable Button1

The disable switch allows the author to "grey out" and disable a button as its initial state.   In the disabled state the button will not press and any   macro that is attached to the button will not be executed .   The TsEnable Command will enable the button and the TsDisable command will disable the button.   If a button has the disabled switch set, make sure that the TsEnable command is issued after any jump to the topic or the disabled state will be reset by the jump to the topic. *(This would even include a JumpKeyword command used to position the cursor after initial topic entry)*.   *Note: Disable capability is not included in the TsToolsW.DLL included with this book.*

Embedded browse buttons, with the *"previous"* button disabled.   Note that this capability allows the author to emulate the browse buttons capability of Viewer.

{ewc TSTOOLS, Tsbutton, "<<" [Font="Arial"/S11/B4] /H20/w80/B1/D2/-}{ewc TSTOOLS, Tsbutton, ">>" [Font="Arial"/S11/B4] /H20/w80/B1/D2}

Tsbutton: Margin   Examples

Margins are used to create a "bounding box" for text.   This is imperative for mixed text and graphic buttons and for precisely locating text.

{ewc TSTOOLS, Tsbutton, "Margin is: T15L10R10 No Others Set -all line breaking auto" [Font="Arial"/S11/B5] /H120/W120/B1/N/D2/MT15R10L10/2}   {ewc TSTOOLS, Tsbutton, "Margin is: T30L20R20 No Others Set - all line breaking auto" [Font="Arial"/S11/B5] /H120/W120/B1/N/D2/MT30R20L20/2}   {ewc TSTOOLS, Tsbutton, "Margin is: T50L50R5 No Others Set all line breaking auto" [Font="Arial"/S11/B5] /H120/W120/B1/N/D2/MT50R5L50/2}

{ewc TSTOOLS, Tsbutton, "Margin is: T45L5R10 No Others Set -all line breaking auto" [Font="Arial"/S11/B5] /H120/W120/B1/N/D2/MT45R10L5/2}   {ewc TSTOOLS, Tsbutton, "Margin is: T5L10R50 No Others Set - all line breaking auto" [Font="Arial"/S11/B5] /H120/W120/B1/N/D2/MT5R50L10/2}   {ewc TSTOOLS, Tsbutton, "Margin is: T25L25R25B25 No Others Set -all line breaking auto" [graphic=`!tsmup.dib',`!tsmdn.dib',`!tsmdi.dib'/ALT] [Font="Arial"/S9/B3] /H120/W120/B1/N/D2/MT25L25R25B25/2}

Tsbutton: X & Y Examples

X & Y Examples:

X10 Y10 **X20 Y20** **X30 Y30** **X40 Y40**

{ewc TSTOOLS, Tsbutton, "X:10 Y10" [Font="Arial"/S11/B3] /H80/W70/x10/y10/B1/N/D2}{ewc TSTOOLS, Tsbutton, "X:20 Y20" [Font="Arial"/S11/B3] /H80/W70/x20/y20/B1/N/D2}{ewc TSTOOLS, Tsbutton, "X:30 Y30" [Font="Arial"/S11/B3] /H80/W70/x30/y30/B1/N/D2}{ewc TSTOOLS, Tsbutton, "X:40 Y40" [Font="Arial"/S11/B3] /H80/W70/x40/y40/B1/N/D2}

In the following example: ***/H80/W80/x10/y10/B2N/D2***

X = the number of pixels to the left of the upper left hand corner of the button (10).
Y = the number of pixels down from the upper left hand corner of the button (10).
(*See the 10x 10y button above for an example of 10x10).*

Note: the X & Y switches are necessary with the Angle switch.
(*Press the "attributes examples" button for angle samples).*

Tsbutton: Font Alignment Examples

Purpose:

The /A switch specifies alignment of text on the button. Without the optional alignment switch, text is centered on the button which is the same as **/ACM**

Alignment is via the optional switch **/A**

The alignments are:

H: Horizontal	Left (L), Center (C) or Right (R)
V: Vertical	Top (T), Middle (M) or Bottom(B)

Text and Alignment Examples

"Centered" {ewc TSTOOLS, Tsbutton, "Centered" [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Left"/AL {ewc TSTOOLS, TsButton, "Left"/AL [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}
"Right"/AR {ewc TSTOOLS, Tsbutton, "Right"/AR [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Left Top"/ALT {ewc TSTOOLS, Tsbutton, "Left Top"/ALT [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}
"Right Bottom"/ARB {ewc TSTOOLS, Tsbutton, "Right Bottom"/ARB [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Right Top"/ART {ewc TSTOOLS, Tsbutton, "Right Top"/ART [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}
"Center Bottom"/ACB {ewc TSTOOLS, Tsbutton, "Center Bottom"/ACB [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Center Top"/ACT {ewc TSTOOLS, Tsbutton, "Center Top"/ACT [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}

..

Height 70 Demo Button

syntax for this button:

{ewc
tstools,
Tsbutton,
"Height 70"
[Font="Arial"/S11/B4/A900]
[Macro = pi(qchPath,`buttonh70>button')]
/H70/W20/x1/y55/B1/N/D2}

Where:

S11	= font size of 11 pixel height
B4	= weight of 4
A900	= text angle 90 degrees (vertical)
H70	= height of 70 pixels
W20	= width of 20 pixels
x1	= start text 1 pixel from left side
y55	= start text 55 pixels below top
B1	= border width of 1 pixel
D2	= depth (highlight) of 2 pixels
N	= notched

Switches for the Font Parameters are:

"Font Name"/S#/B#/D#/I#/U#/O#/A# where

The *Font Name* must be in quotes and must be found on the system.
If the font is not found the default font is Helvetica 8 point.

S	is the font size in pixels
B	is the font weight (0-9) - with 4 being normal *(some fonts only support weights of 4 and 7)*
I	Italics on (1) or off - no entry or (0)
U	Underline on (1) or off - no entry or (0)
O	Overstrike on (1) or off - no entry or (0)
A	**is for the text angle in 10ths of a degree eg 0 is normal,900 is vertical 1800 upside down (the angle switch is used in conjunction with the X and Y switches)**

Ts Macro

The **TsMacro Function** acts like a subroutine.
It submits a string of macros of up to 512 characters without having to enter a topic or a group. This function effectively overcomes the 512 character limitation of a topic entry macro or a group macro, but using multiple calls to **TsMacro**

Here is a handy little trick:

To create a button bar at the bottom of a topic:

1. build the {embedded pane tsbuttons}
2. ensure they are in the nonscrolling region
3. set the non scrolling region to "bottom"

{ewc TSTOOLS,tsbutton,""[name=tsbut][graphic=`!tsup.dib',`!tsdn.dib',`!tsdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=filebut][graphic=`!file1up.dib',`!file1dn.dib',`!file1di.dib']} {ewc
TSTOOLS,tsbutton,""[name=printbut][graphic=`!printup.dib',`!printdn.dib',`!printdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=diskbut][graphic=`!diskup.dib',`!diskdn.dib',`!diskdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=cutbut][graphic=`!cutup.dib',`!cutdn.dib',`!cutdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=clipbut][graphic=`!clipup.dib',`!clipdn.dib',`!clipdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=clockbut][graphic=`!clockup.dib',`!clockdn.dib',`!clockdi.dib']} {ewc
TSTOOLS,tsbutton,""[name=facebut][graphic=`!faceup.dib',`!facedn.dib',`!facedi.dib']
[macro=pi(qchpath,`smilemsg')]} {ewc TSTOOLS,tsbutton,""[name=helpbut][graphic=`!help1up.dib',`!
help1dn.dib',`!help1di.dib']}

Tsbutton: Graphics

The Button Bar displayed above is a series of TsButtons displaying graphics. By putting the buttons in the nonscolling region, the author has control over the color and location of the button bar. Each of the following examples demonstrates control and flexibility that TsButtons deliver to the author. *(There are no commands attached to the buttons except the smile button).* *Note: Graphics capability is not included in the TsToolsW.DLL included with this book.*

Move Button Bar to Bottom	Move Button Bar to Top
BBar on Bottom with Separator	BBar on Top with Separator
Black Non Scrolling Region	Dk Grey NScrolling Region
Disable Entire Buttonbar	Enable Entire Buttonbar
Hide Entire ButtonBar	Display Entire ButtonBar
Press Happy Face by Function Call	
Put a Vertical ButtonBar in a Pane	Floating Tool Bar

```
{ewc TSTOOLS,tsbutton,""[name=vfilebut][graphic=`!vfile1up.dib',`!vfile1dn.dib',`!file1di.dib']}
{ewc TSTOOLS,tsbutton,""[name=vprintbut][graphic=`!vprintup.dib',`!vprintdn.dib',`!printdi.dib']}
{ewc TSTOOLS,tsbutton,""[name=vdiskbut][graphic=`!vdiskup.dib',`!vdiskdn.dib',`!diskdi.dib']}
{ewc TSTOOLS,tsbutton,""[name=vcutbut][graphic=`!vcutup.dib',`!vcutdn.dib',`!cutdi.dib']}
{ewc TSTOOLS,tsbutton,""[name=vclipbut][graphic=`!vclipup.dib',`!vclipdn.dib',`!clipdi.dib']}

{ewc TSTOOLS,tsbutton,""[name=vclockbut][graphic=`!vclockup.dib',`!vclockdn.dib',`!clockdi.dib']}
{ewc TSTOOLS,tsbutton,""[name=vfacebut][graphic=`!vfaceup.dib',`!vfacedn.dib',`!facedi.dib']
[macro=pi(qchpath,`smilemsg')]}

{ewc TSTOOLS,tsbutton,""[name=vhelpbut][graphic=`!vhelp1up.dib',`!vhelp1dn.dib',`!vhelp1di.dib']}
```

```
{ewc TSTOOLS,tsbutton,""[name=ffilebut][graphic=`!file1up.dib',`!file1dn.dib',`!file1di.dib']}{ewc
TSTOOLS,tsbutton,""[name=fprintbut][graphic=`!printup.dib',`!printdn.dib',`!printdi.dib']}{ewc TSTOOLS,tsbutton,""[name=fdiskbut]
[graphic=`!diskup.dib',`!diskdn.dib',`!diskdi.dib']}{ewc TSTOOLS,tsbutton,""[name=fcutbut][graphic=`!cutup.dib',`!cutdn.dib',`!
cutdi.dib']}
{ewc TSTOOLS,tsbutton,""[name=fclipbut][graphic=`!clipup.dib',`!clipdn.dib',`!clipdi.dib']}{ewc TSTOOLS,tsbutton,""[name=fclockbut]
[graphic=`!clockup.dib',`!clockdn.dib',`!clockdi.dib']}{ewc TSTOOLS,tsbutton,""[name=ffacebut][graphic=`!faceup.dib',`!facedn.dib',`!
facedi.dib'][macro=pi(qchpath,`smilemsg')]}{ewc TSTOOLS,tsbutton,""[name=fhelpbut][graphic=`!help1up.dib',`!help1dn.dib',`!
help1di.dib']}
{ewc TSTOOLS,tsbutton," About This Toolbar    "[name=closeme][macro=TsInfoBox(5,`TouchSend ToolBar Demo',`The toolbar
displayed is a stay on top secondary window,',`with 10 tsbuttons - 8    are graphic buttons, 2 are text buttons.',`                This floating
toolbar can survive topic jumps.',255,0,0,`')]/w112/h16}
{ewc TSTOOLS,tsbutton," Close This Toolbar    "[name=closeme][macro=CloseWindow(`floater')]/w112/h16}
```

```
{ewc TSTOOLS, TsButton, "Click the ~Smile Face" [Macro=TsPressButton(`vfacebut',500)] [Font="Times New
Roman" /S11/B5]W90/H34/B2/D2/N/2}
{ewc TSTOOLS, TsButton, "Hide 6 buttons"
[Macro=TsHideButton(`vfilebut',192,192,192);TsHideButton(`vprintbut',192,192,192);TsHideButton(`vdiskbut',192,192,
192);TsHideButton(`vcutbut',192,192,192);TsHideButton(`vclipbut',192,192,192);TsHideButton(`vclockbut',192,192,19
2)] [Font="Times New Roman" /S11/B5]w90/H24/B2/D2/N/2}
{ewc TSTOOLS, TsButton, "Redisplay the~6 Buttons"
[Macro=TsShowButton(`vfilebut');TsShowButton(`vprintbut');TsShowButton(`vdiskbut');TsShowButton(`vcutbut');TsSho
wButton(`vclipbut');TsShowButton(`vclockbut')] [Font="Times New Roman" /S11/B5]W90/H34/B2/D2/N/2}
{ewc TSTOOLS, TsButton, "Disable ? Button" [Macro=TsDisableButton(`vhelpbut')] [Font="Times New Roman"
/S11/B5]W90/H24/B2/D2/N}
{ewc TSTOOLS, TsButton, "Enable ? Button" [Macro=TsEnableButton(`vhelpbut')] [Font="Times New Roman"
/S11/B5]W90/H24/B2/D2/N}
{ewc TSTOOLS, TsButton, "Close the ~Vertical ~Button Bar"
[Macro=ClosePane(`main',`vertbutn');ClosePane(`main',`vertcomm');PaneID(`tstools>main',
`tspanedemo>butnpan2',0)] [Font="Times New Roman" /S11/B5]W90/H49/B2/D2/N/2}
```

Tsbutton: Mixed Text and Graphics

Text and graphics can be combined - either side by side, or text on top of graphics. *Note: Multiline text and Graphics capability and TsButton functions are not included in the TsToolsW.DLL included with this book.*

{ewc TSTOOLS,TsButton,"&Printer~Button"/AR/ML24R3T5/H50/W80/B2/D2/N/2 [name=mixprintbut] [graphic=`!yprintup.dib',`!yprintdn.dib',`!yprintdi.dib'/ALC][Font="Times New Roman"/S13/B4/3-]}

Disable Button **Enable Button** **Click Button**

{ewc TSTOOLS,TsButton,"Printer"/H46/W80/B1/D1/AR[name=xprintbut][graphic=`!xprintup.dib',`!xprintdn.dib',`!xprintdi.dib'/ALC][Font="Times New Roman"/S16/B4]} {ewc TSTOOLS,TsButton,"&Test"/ARM/MR3/B1/D2/N[name=showmore][graphic=`!tst1up.dib',`!tst1dn.dib',`!tst1dn.dib'/ALC][Font="Times New Roman"/S16/B8/3+]}

{ewc TSTOOLS,TsButton,"Marble Enabled"/B1/D1/N/[name=marble1][graphic=`!marbleup.dib',`!marbledn.dib',`!marbledi.dib'][Font="Times New Roman"/S11/B6/3-]} {ewc TSTOOLS,TsButton,"Marble Disabled"/B1/D1/N/-[name=marble2][graphic=`!marbleup.dib',`!marbledn.dib',`!marbledi.dib'][Font="Times New Roman"/S11/B6/3-]}

{ewc TSTOOLS,TsButton,"Yes"/B1/D1/N[name=chitz1][graphic=`!chitzup.dib',`!chitzdn.dib',`!chitzdi.dib'][Font="Arial"/S11/B7/3+]}

```
{ewc TSTOOLS, TsButton, "Main Screen" [Macro=Ji(qchPath,`tsfull')] [Font="Arial" /S11/B7/3-]/H20/w115/B1/D2}
{ewc TSTOOLS, TsButton, "TsButton Contents" [Macro=Ji(qchPath,`tsbutton')] [Font="Arial"
/S11/B7/3-]/H20/w115/B1/D2}
{ewc TSTOOLS, TsButton, "Full Summar&y" [Name=summary][Macro=Ji(qchPath,`buttonsummary')] [Font="Arial"
/S11/B7/3-][Font="Arial" /S11/B7/3-]/H24/w115/B1/D2}
{ewc TSTOOLS, TsButton, "Embedded Pane"   [Macro=ji(qchPath,`tsbutewx')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "DLL Name &&~                    Button Class"   [Macro=ji(qchPath,`tsbutdll')][Font="Arial"
/S9/B4/3-]/H28/w115/B1/D2/AL/ML4/B1/2}
{ewc TSTOOLS, TsButton, "Font Properties"   [Macro=ji(qchPath,`tsbutfont')][Font="Arial"
/S11/B7/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Attribute Examples"   [Macro=ji(qchPath,`tsbutfontattributes')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Button Properties"   [Macro=ji(qchPath,`tsbutlook')][Font="Arial"
/S11/B7/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, " Width Examples"    [Macro=ji(qchPath,`tsbutlookwidth')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Height Examples"    [Macro=ji(qchPath,`tsbutlookheight')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Border Examples"   [Macro=ji(qchPath,`tsbutlookborder')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Depth Examples"   [Macro=ji(qchPath,`tsbutlookdepth')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Notch Examples"    [Macro=ji(qchPath,`tsbutlooknotch')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Margin Examples"    [Macro=ji(qchPath,`tsbutlookmargin')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "XY Examples"    [Macro=ji(qchPath,`tsbutlookxy')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Alignment Examples"   [Macro=ji(qchPath,`tsbuttext')][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Disabled Examples" [Macro=ji(qchPath,`tsbutlookgreyed')] [Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Graphics" [name=tfacebut][graphic=`!tfaceup.dib',`!tfacedn.dib',`!tfacedi.dib'/ALC]
[Macro=closepane(`main',`butnpane');PaneID(`tstools>main',
`smallbuttonbar>butnpan1',0) ;ji(qchPath,`graphicdemo')][Font="Arial" /S11/B7/3-]/H24/w115/B1/D2/ARM/MR10}
{ewc TSTOOLS, TsButton, "Text          Graphics"[graphic=`!zapup.dib',`!zapdn.dib',`!zapdi.dib'/ALC]
[Macro=ji(qchPath,`mixeddemo')][Font="Arial" /S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Button Functions" [Macro=Ji(qchPath,`tsbutfunctions')] [Font="Arial" /S11/B7/3-]
[Font="Arial" /S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
```

```
{ewc TSTOOLS, TsButton, "TsButton Contents" [Macro=Ji(qchPath,`tsbutton')] [Font="Arial"
/S11/B7/3-]/H20/w115/B1/D2}
{ewc TSTOOLS, TsButton, "Graphics" [name=tfacebut][graphic=`!tfaceup.dib',`!tfacedn.dib',`!tfacedi.dib'/ALC]
[Macro=JI(qchPath,`graphicdemo')][Font="Arial" /S11/B5/3-]/H24/w115/B1/D2/ARM/MR10}
{ewc TSTOOLS, TsButton, "Bitmap Syntax" [Macro=Ji(qchPath,`bitmapsyntax')] [Font="Arial"
/S11/B5/3-]/H24/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Bitmap Placement" [Macro=Ji(qchPath,`bitmapplacement')] [Font="Arial"
/S11/B5/3-]/H24/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Text          Graphics"[graphic=`!zapup.dib',`!zapdn.dib',`!zapdi.dib'/ALC]
[Macro=ji(qchPath,`mixeddemo')][Font="Arial" /S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
{ewc TSTOOLS, TsButton, "Mouse Move"[enter=TsVCopyS(`status1',`Mouse ENTER event~ triggered
this~message')][Macro=ji(qchPath,`mousemove');TsKillTimer();TsVCopyS(`status1',`Clicking Mouse
Move~Button triggered~ this message');TsTimer(`TsVCopyS(`status1',`~')',3)][Font="Arial"
/S11/B5/3-]/H20/w115/B1/D2/AL/ML4}
```

TsVCopyS(`varname',`string')

Tsbutton: Bitmap Syntax

Purpose:

Specifies the bitmaps to be displayed on the face of the button. *Note: Graphics capability is not included in the TsToolsW.DLL included with this book.*

A graphics button will display the following:

upbitmap	= the bitmap displayed in the initial enabled state
downbitmap	= bitmap displayed when the button is clicked
disablebitmap	= bitmap displayed when the button is disabled

Note: It is imperative that the palettes for all of the bitmaps in all of the states and buttons be the same or the buttons will not display properly. This includes any other bitmaps that will be displayed simultaneously.

Without the optional alignment switch, text is centered on the button. If there are no height and width specifications, the button will autosize to the upbitmap.

Tsbutton: Graphic Alignment

The **/A** switch used with graphics specifies alignment of a graphic on the button. Without the optional alignment switch, the graphic is centered on the button which is the same as **/ACM**. If size is not specified, the button autosizes to the up graphic. **Alignment** is via the optional switch **/AHV** where

H: Horizontal	Left (L), Center (C) or Right (R)
V: Vertical	Top (T), Middle (M) or Bottom(B)

Graphic Placement can also be managed precisely by adding background grey coloring to the graphic to properly position the acutal displayable graphic precisely. This is especially useful when creating buttons that have Mixed Text and Graphics.

"Centered" {ewc TSTOOLS, Tsbutton, "" [graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'] [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Left"/AL {ewc TSTOOLS, TsButton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/AL] [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}
"Right"/AR {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/AR] [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Left Top"/ALT {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/ALT] [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1}
"Right Bottom"/ARB {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/ARB] [Font="Times New Roman" /S9/B4]/H30/w70/B1/D1} "Right Top"/ART {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/ART] /H30/w70/B1/D1}
"Center Bottom"/ACB {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/ACB]/H30/w70/B1/D1} "Center Top"/ACT {ewc TSTOOLS, Tsbutton, ""[graphic=`!tsgup.dib',`!tsgdn.dib',`!tsgdi.dib'/ACT]/H30/w70/B1/D1}

Tsbutton: Ancillary Functions

The following button control functions are available in the commercial version of TsTools but are not included in the TsToolsW.DLL included with this book.

TsPressButton:	Allows the author to programmatically "click" any TsButton.
TsEnableButton:	Will enable a button that has been disabled with the TsDisableButton function**.**
TsDisableButton:	Disables a button.
TsHideButton:	Hides a button.
TsShowButton:	Make a button visible that has been hidden with the TsHideButton command.

Have a Great Day !!

CreateButton(`btn_dtsdocs', `TsToolsW',`JI(qchPath,`dtsdocs')');CreateButton(`btn_back', `Go Back', `Back()');CreateButton(`btn_previous', `<<', `Prev()');CreateButton(`btn_next', `>>', `Next()');CreateButton(`btn_return', `Return to Demo', `JI(qchPath,`tsfull');DestroyButton(`btn_dtsdocs');DestroyButton(`btn_back');DestroyButton(`btn_previous');DestroyButton(`btn_next');DestroyButton(`btn_return');hidebuttonbar();DeleteMark(`been here')');Showbuttonbar()

TsButton: Full Syntax Summary

{ewc TSTOOLS, TsButton, "Print this Summary" [Macro=Print()] [Font="Arial" /S9/B4/3-]/H20/w100/B1/D2}{ewc TSTOOLS, TsButton, "TsButton" [Macro=Ji(qchPath,`tsbutton');Showbuttonbar()] [Font="Arial" /S11/B5/3-]/H20/w100/B1/D2}

Font Properties:

Text Font properties that can be authored include Size, Weight, Italic, Underline, Overstrike, Three Dimensional Look, and Text Angle. Use the & to underline a character and && to have the `&' character show. *3D buttons are not included in TsToolsW.DLL*

Switches for the Font Parameters are:

"Font Name"/S#/B#/D#/I#/U#/O#/A# where

The *Font Name* must be in quotes and must be found on the system. If the font is not found the font will be helvetica 8 point.

S	is the font size in pixels
B	is the font weight (0-9) - with 4 being normal *(some fonts only support weights of 4 and 7)*
I	Italics on (1) or off - no entry or (0)
U	Underline on (1) or off - no entry or (0)
O	Overstrike on (1) or off - no entry or (0)
A	is for angle in 10ths of a degree with 0 straight right *(see XY switches in button attributes section)*

Macro(Command) Properties:

The commands that are submitted when the button is clicked include all standard Viewer commands and any properly registered routines, including all of the TsTools commands. The syntax is as follows:

[Macro = "command;command;command etc"]

There is a limit of 512 characters unless TsMacro is used.

Name Property: (not included in the TSTOOLSW.DLL)

TsButtons can be given a name which will allow the button to be addressed for the purpose of programmatically clicking it, hiding it, or disabling it (and thereafter unhiding it or enabling it).The syntax is as follows:

Graphics Properties: (not included in the TSTOOLSW.DLL)

TsButtons can display 3 graphic states, up, down and disabled, where each of *upgraphic,downgraphic* and *disabledgraphic* is the file name of a bitmap file.

Graphics can be displayed alone, or in conjuction with Text. If size parameters are not used, the button will size to the size of the *upgraphic*. It is the author's responsibility to ensure that the bitmap sizes are the same and that the techniques for offsetting the "down" bitmap with the appropriate changes in border shading are implemented if "movement" on clicking is intended.

The disabledgraphic bitmap will only be displayed by calling the TsDisableButton function or starting in the disabled state.

Graphics can be optionally aligned using the alignment switch /AHL where H - Horizontal alignment can be L (Left) C (Center) or R (Right) and V - Vertical alignment can be L (Left), M (Middle) or R (Right), The default value is CC if the alignment switch is not used.

Button Attributes:

/W#/H#/M????**/B#**????/**N/D#/A**??**/X#/Y#**where

W Button width in pixels

H Button height in pixels

M Text Margin in pixels

where ???? controls which text margins are implemented
no ???? means no margin management. Otherwise specify:
T = Top; **B**=Bottom
L = Left; **R**=Right

Examples "/MR10 will mean the text is inset 10 pixels from the right
Examples "/ML10T10 will mean the text is inset 10 pixels from the left and 10 pixels from the top of the button. Margins are especially useful for multi-line text buttons using the /2 switch with or without the ~ as a line break manager.

B Border width in pixels
where ???? controls which borders are drawn
no ???? means all 4 will be drawn. Otherwise specify:
T = Top; **B**=Bottom
L = Left; **R**=Right

N Notched corners

D Depth *(highlight)* width

A alignment of text on the button
orizontal alignment may be Left, Right or Centered
Vertical alignment may be Top Middle or Bottom

Example: /ALC will left align the text and center it vertically

X starting X position for text in the button
Y starting Y position for text in the button
(these two are necessary if using the /A for angle switch under in the fonts section of the string to create an approriate start location)

Disabled buttons, disabled button printing, multiline buttons, graphic buttons and mixed text and graphic buttons are not included in TsToolsW.DLL

Button Functions: *(not included with TsToolsW.DLL)*

TsPressButton:	Allows the author to programmatically "click" any TsButton.
TsEnableButton:	Will enable a button that has been disabled with the TsDisableButton function or started disabled.
TsDisableButton:	Disables a button.
TsHideButton:	Hides a button.
TsShowButton:	Make a button visible.

Tsbutton: Mouse Move

The following **TsButton** and **TsPane** mouse move functions are available in the commercial version of TsTools but are not included in the TsToolsW.DLL included with this book.

Mouse Enter:	Submits commands to Viewer upon the mouse moving onto a TsButton or TsPane. This can be used with most TsFunctions and Viewer Functions. The status bar to the left makes use of the Mouse Enter Function on the *"mouse move"* button. Move the mouse over the mouse move button. Then click it. Then click the status bar.
Mouse Leave:	Submits commands to Viewer upon the mouse moving off of a TsButton or TsPane if the TsButton or TsPane are not clicked. This can be used with most TsFunctions and Viewer Functions.

Watch the area below the picture and the status bar to the left as the mouse enters and leaves the picture. Then click the picture.

```
{ewc TSTOOLS, TsPane, "skypic"[graphic=`!winlogo.dib'][enter=TsVCopyS(`skymsg',`This is part of the
Windows Logo.');TsKillTimer();TsVCopyS(`status1',`Windows Logo Message~in 2 places')]
[leave=TsVCopyS(`skymsg',`~');TsVCopyS(`status1',`The TouchSend~Status Bar')]
[Macro=TsVCopyS(`skymsg',`~');TsVCopyS(`status1',`~');TsInfoBox(2,`TouchSend Mouse Move',`
Mouse move allows the author to send',`     commands when a mouse enters or leaves',`     a TsPane or
TsButton.',192,192,192,`')][Color=192,192,192,0,0,0][Font="Arial" /S11/B5/]/B0}
{ewc TSTOOLS, TsPane, "skymsg"[text=`skymsg'][Color=255,255,255,0,0,128][Font="Times New
Roman" /S12/B6/]/H23/W250/2}
```

Purpose: Manages TouchSend functions to ensure that coordinates are either relative or absolute. For example the function determines whethe a TsButton specified to be Height 40 and Width 100 is actually 40x100 pixels no matter what the resolution or 40x100 relative to the display resolution.

Viewer device-independent measurements map position values into a 1,024-by-1,024 grid. At run time, these measurements are converted to device-specific coordinates using a scaling ratio appropriate to the display device. For example, in 640-by-480 video mode, a device-independent measurement of 512 (specified for the X or Width parameters) would be converted to a pixel measurement of 320 (512 x (640/1024)). For the Y and Height parameters, the same value would produce a pixel measurement of 240 (512 x (480/1024)). TsAbsolute defaults to TsAbsolute(1) unless TsAbsolute(0) is called. It is recommended that the author stick to one case for an entire title and that the call to TsAbsolute be made in the configuration script after a call to **TsToolsInit**

Syntax: **TsAbsolute(`Integer')**

Parameters: **0** will cause TsButtons to be mapped to a device independent 1024x1024 grid.

1 will ensure that whatever pixel coordinates are specified will be faithfully reproduced on the screen regardless of the coordinates.

To Register: **RegisterRoutine(`TSTOOLSW',`TsAbsolute',`v=i')**
(press the register routine button above to copy the RR string to the clipboard)

Purpose:	Copies the string enclosed in the function to the clipboard. This function is particularly useful in order to select information and have it available for other applications through the clipboard. **Click here for a quick demo of TsCopyString(`Hello Friends')**
Syntax:	**TsCopyString(`String')**
Parameters:	**String** is the string to be passed to the clipboard.
To Register:	**RegisterRoutine("TSTOOLSW","TsCopyString","v=S")** *(press the register routine button above to copy the RR string to the clipboard)*

Purpose:	A function to call and manage Windows Help from inside Viewer. It calls a Windows Help file, opening at a context id number that has been defined in the help [MAP] section.
Syntax:	**TsHelpContext(HelpFileName,HelpContextID)**
To Register:	**RegisterRoutine(`TsHelpContext',`v=SU')** *(press the register routine button above to copy the RR string to the clipboard)*

Purpose: Displays a dialog box with author information. The author has control of the title bar of up to 31 characters), 3 lines of text of up to 50 characters, color control of the dialog box. There are five available dialog boxes. Authors with resource editors can customize the size, shape, location and icons in each box.*(Make sure that the id's all stay the same).* Some of the syles appear the same, but are at different xy locations or have different frames.

Syntax: **TsInfoBox(StyleNo,Title,Text1,Text2,Text3,Color1,Color2,Color3,Commands1)**

Parameters:

StyleNo:	A number from 1 to 5 specifying the dialog box style. (See examples)
Title	The title bar (up to 31 characters) - may be left blank (a null string)
Text1	The first line of text (up to 50 characters) - may be left blank (a null string)
Text2	The first line of text (up to 50 characters) - may be left blank (a null string)
Text3	The first line of text (up to 50 characters) - may be left blank (a null string)
Colors 1,2,3	A digit from 0 to 255
Commands1	String specifying commands to run after clicking "OK" (it may be left blank)

eg: StyleNo 1: **TsInfoBox(1,`TouchSend InfoBox Style 1',`This is the first line of text',`This is the second line of text',`This is another line of textbut much longer than the first two',192,192,192,`')**

{ewc TSTOOLS, Tsbutton, "StyleNo 1" [Macro=TsInfobox(1,`TouchSend StyleNo Style 1',`This is the first line of text',`This is the second line of text',`This is another line of text but much longer than the first two',192,192,192,`')] [Font="Arial" /S9/B4]/H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 2" [Macro=TsInfobox(2,`TouchSend StyleNo - Style 2',`TouchSend Variables allow the author',`to provide feedback to a user including interactive learning,',`testing, scoring, and other useful messages.',0,128,0,`')] [Font="Arial" /S9/B4]/H20/W60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 3" [Macro=TsInfobox(3,`TouchSend StyleNo - Style 3',`TouchSend Task Functions can turn any Windows Application.',`into a task of Viewer - and provides a user with',`the tools to make use of information delivered in a Viewer title.',0,255,255,`')] [Font="Arial" /S9/B4]/H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 4" [Macro=TsInfobox(4,`TouchSend StyleNo - Style 4',`The TsInfobox Function enables the author to',`submit a command to Viewer after the message.',`    In this case the About() command will be submitted.',255,255,0,`About()')] [Font="Arial" /S9/B4]/H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 5" [Macro=TsInfobox(5,`TouchSend StyleNo - Style 5',`This TsInfoBox is for rent.',`Call 904-668-6180 for further information.',`TouchSend Consulting can help you fast track your title.',0,255,0,`')] [Font="Arial" /S9/B4]/H20/w60/B2/D2}

To Register: **RegisterRoutine("TSTOOLSW","TsInfoBox","v=iSSSSuuuS")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Extracts files from baggage, brings up a common dialog box and allows the user to save the file to the disk with the name and at the location specified. The author has control over the color displayed in the common dialog box.

Syntax: **TsSaveBaggage(`String',RNumber,GNumber,BNumber)**

Parameters:

String is the name of the baggage file.

Note: Baggage is case sensitive and the ***case and name must be exact***.

****Remember this, as it is a common error to overlook it.****

RNumber is an integer between 0 and 255
GNumber is an integer between 0 and 255
BNumber is an integer between 0 and 255

To Register: **RegisterRoutine("TSTOOLSW","TsSaveBaggage","v=Suuu")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Initializes the TsToolsW.DLL and creates an ini file specified by the author, or if the ini file already exists, makes use of it. **If TsToolsInit is not called as the first function in the configuration script after all of the register routines, TsToolsW will not work.**. Make sure that each title authored has a different Ini file name or the title and operation of the TsFunctions will not be reliable.

TSTools makes use of the created ini file for several actions related to the functions. It is created and maintained in the Windows directory. TsToolsW.DLL may not need to create and ini file because of it's limited functionality, but it must be declared in the config call or the DLL will not work. Don't be alarmed if no ini file is actually created.

Syntax: **TsToolsInit(qchPath,"IniName")**

Parameters:

qchPath is the Viewer internal variable referencing the title. The variable is NOT surrounded by quotes and must be spelled exactly as set out above.

IniName is the a string that is the name of an ini file. The string must follow DOS conventions, namely not more than 8 characters and the suffix "ini".

To Register: **RegisterRoutine("TSTOOLSW","TsToolsInit","v=SS")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Allows the user to execute a copy of Write.exe and pass information from Viewer to Write via the clipboard. After write is launched using the function, each subsequent call to TsWrite will bring forward the existing copy of Write.exe rather than launching a new copy as would be the case with the Viewer ExecProgram function. Focus can be passed to Write or stay with Viewer. Write can be repositioned using the **TsWritePos** command.

Syntax: **TsWrite(`String', Zorder, Dependency, Focus)**

Parameters:

String is the name of a write file and is optional. If.the file is not a "wri" file, write will ask if you want to convert the file. This will cause write to reposition. It can be moved back to the default tiled position or any other position using the **TsWritePos** command.

Z order is an integer specifying the Z order of Write. See **TsWriteSetZ**.

Dependency is an integer specifying whether or not write is "dependent" on Viewer or is an independent application. If an application is dependent, it will maximize and minimize with Viewer and will close when the Viewer title that called it closes. If it is independent, while Viewer is aware of and can interact with the application, it will not minimize, restore or close in concert with Viewer. The value **0** is used for an independent application and the value **1** is used for a dependent application.

Focus is an integer specifying whether or not Write or Viewer obtains focus after TsWrite is called. Specifying **1** will set focus to Write. Specifying **0** will set focus to Viewer.

See Also **TsWriteCopy, TsWritePaste, TsWriteKill, TsWritePos, TsWriteSetZ,**

To Register: **RegisterRoutine(`TsToolsW',`TsWrite',`v=Siii')**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Sends the keystrokes to Write that are the equivalent of *Edit Copy*. Write will copy to the clipboard any text that is highlighted or selected.

Syntax: **TsWriteCopy()**

To Register: **RegisterRoutine("TSTOOLSW","TsWriteCopy","v=")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose:	Closes Write after it has been called using the TSWrite function.
Syntax:	**TsWriteKill()**
To Register:	**RegisterRoutine("TSTOOLSW","TsWriteKill","v=")** *(press the register routine button above to copy the RR string to the clipboard)*

Purpose: Sends the keystrokes to Write that are the equivalent of *Edit Paste*. Write will paste any text that is in the clipboard to the Write instance called by the TsWrite function.

Syntax: **TsWritePaste()**

To Register: **RegisterRoutine("TSTOOLSW","TsWritePaste","v=")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Sets the screen position of Write after calling it with TsWrite. It sets the upper left hand corner of the Write window as well as the width and height.

Syntax: **TsWritePos(integerX,integerY,integerW,integerH)**

Parameters:

integerX	is the x position of the upper left hand corner.
integerY	is the y position of the upper left hand corner.
integerW	is the width.
integerH	is the height.

Note: by passing(-1,-1) for either the XY or the WH pairs, they will remain unchanged from the previous position.

The resultant size will depend on whether **TsAbsolute** is set relative or absolute.

To Register: **RegisterRoutine("TSTOOLSW","TsWritePos","v=iiii")**
(press the register routine button above to copy the RR string to the clipboard)

Purpose: Sets the Z order of Write after it has been executed using TsWrite. There are 4 cases that can be set using this function:

1 places window at bottom of the z order

0 places at top of z order - keeps its current status (ie if write is set as a top most type it stays a topmost type and vice versa.

-1 places window above all topmost windows.

-2 places window at the top of all but "topmost windows".

The Z order is akin to the place of a card in a deck of cards. "Topmost" windows are stay on top windows. That is they will stay on top of other applications even if they do not have focus. If there are multiple stay on top Windows, This is the same functionality as is used by Windows applications to stay on top of other applications.

Syntax: **TsWriteSetZ(integer)**

To Register: **RegisterRoutine("TSTOOLSW","TsWriteSetZ","v=i")**
(press the register routine button above to copy the RR string to the clipboard)

In order to illustrate Z order differences, try each of the example buttons below:

Example 1: Launch Write and set as a topmost window

{ewc TSTOOLS, TsButton, "Example 1"[Macro=TsWriteKill();TsWrite(`',-1,1,1);TsAbsolute(1);TsWritePos(300,20,300,200)] [Font="Times New Roman" /S9/B4]/H20/w80/B1/D1}

Click your mouse on the title, not Write. Note that even when Write does not have focus (ie the titlebar is not highlighted) that it does not disappear. This setting makes it extremely useful to use in conjuction with a title. Then close write. It can be closed using the **TsWriteKill() Function** or by closing it from within Write.

Example 2: Launch Write and but do not set as a topmost window

{ewc TSTOOLS, TsButton, "Example 2"[Macro=TsWriteKill();TsWrite(`',0,1,1);TsWritePos(0,20,200,200)] [Font="Times New Roman" /S9/B4]/H20/w80/B1/D1}

Click your mouse on the title, not Write. Note that Write becomes lower in the Z order (ie is below the title) and disappears from View. Then close write. It can be closed using the **TsWriteKill() Function** or by closing it from within Write.

Purpose: Displays a dialog box with author information. Commands will be submitted to Viewer based upon the button pressed. The author has control of the title bar of up to 31 characters), 3 lines of text of up to 50 characters, color control of the dialog box. There are five available dialog boxes. Authors with resource editors can customize the size, shape, location and icons in each box. *(Make sure that the id's all stay the same)* Some of the styles appear the same, but are at different xy locations or have different frames. *(Note: Only style no 1 below responds to the yes and no buttons)*

Syntax: **TsYN(StyleNo,Title,Text1,Text2,Text3,Color1,Color2,Color3,Commands1,Commands2)**

Parameters:

StyleNo:	A number from 1 to 5 specifying the dialog box style. (See examples)
Title	The title bar (up to 31 characters) - may be left blank (a null string)
Text1	The first line of text (up to 50 characters) - may be left blank (a null string)
Text2	The first line of text (up to 50 characters) - may be left blank (a null string)
Text3	The first line of text (up to 50 characters) - may be left blank (a null string)
Colors 1,2,3	A digit from 0 to 255
Commands1	String specifying commands to run after clicking "Yes" (it may be left blank)
Commands2	String specifying commands to run after clicking "No" (it may be left blank)

eg: StyleNo 1: *TsYN(1,`TouchSend YN Style 1',`This is the first line of text',`This is the second line of text',`This is another line of textbut much longer than the first two',192,192,192,`pi(qchpath,`yes')',pi(qchpath,`no')')*

{ewc TSTOOLS, TsButton, "StyleNo 1" [Macro=TsYN(1,`TouchSend YN Style 1',` This is the first line of TsYN Text',` You can press Yes or No',` Please do so now... !',192,192,192,`pi(qchpath,`yes')',`pi(qchpath,`no')')] [Font="Arial" /S9/B4/3-]/H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 2" [Macro=TsYN(2,`TouchSend YN Style 2',` This is the first line of TsYNText',` You can press Yes or No',` Please do so now... !',192,192,192,`',`')] [Font="Arial" /S9/B4/3-]/H20/W60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 3"[Macro=TsYN(3,`TouchSend YN Style 3',` This is the first line of TsYN Text',` You can press Yes or No',` Please do so now... !',192,192,192,`',`')] [Font="Arial" /S9/B4/3-]/H20H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 4" [Macro=TsYN(4,`TouchSend YN Style 4',` This is the first line of TsYNBox Text',` You can press Yes or No',` Please do so now... !',192,192,192,`',`')] [Font="Arial" /S9/B4/3-]/H20/w60/B2/D2} {ewc TSTOOLS, Tsbutton, "StyleNo 5" [Macro=TsYN(5,`TouchSend YN Style 5',` This is the first line of TsYN Text',` You can press Yes or No',` Please do so now... !',192,192,192,`',`')] [Font="Arial" /S9/B4/3-]/H20/w60/B2/D2}

To Register: **RegisterRoutine("TSTOOLSW","TsYN","v=iSSSSuuuSS")**
(press the register routine button above to copy the RR string to the clipboard)

Crash Protection & Other Problems

Viewer is very sensitive to the syntax used in embedded panes. If you have authored a TsButton with an extra ***"{"*** or and extra ***"}"*** , Viewer tends to get confused and may crash. Don't get mad - just go back and read your embedded pane string. More often than not it is a typing or syntax error. If your button does not look like you thought it should, make sure that your switches are correct.

If you get the message "cannot display" where you have authored a TsButton, most likely it is because TsToolsInit has not been called. **If TsToolsInit is not called as the first function in the configuration script after the registration routines, TsToolsW will not work.** Make sure that each title authored has a different Ini file name or the title and operation of the TsFunctions will not be reliable. TsToolsW may not create an ini file but it must still be declared in the function.

If nothing works or some function(s) don't work: check the Register Routines If the syntax is not exactly precise, the functions/buttons will not work. eg an "i" is not the same as an "I". A "u" is not the same as a "U". If a function specifies "v=iiii" and you have "v="ii" , it will not work. It is suggested that the **rr.txt** file be pasted into your mvp file using Notepad.exe. Then is must be reloaded into the Project Editor. Another common error is to make changes to an mvp file and then forgetting to reload it into the Project Editor resulting in the changes being overwritten the next time the Project Editor is saved. Then make sure that **TsToolsInit** is called. (see comments above re TsToolsInit)

If you get a message stating that TsToolsW.DLL is damaged: you are probably running the TsTools.MVB at the same time. Unload it and start again. If you are not running TsTools.mvb, copy a new copy of TsToolsW.DLL from the CD-ROM to your Windows\System directory.

Still Stumped ? Send email to TsTools on Compuserve: ID 73374,2071 and we will do the best we can to help you out. Make sure to include your mvp file and the "offending code"**.**

rr.txt

rr.txt is a file on the CD ROM that includes all of the function declarations to be inserted in the configuration section of the MVP file for the TsToolsW.DLL functions.   Because these declarations are so "fussy" ie   "i" is not the same as "I" and "u" is not the same as "U", it is recommended that the entire file be pasted into the config section of your mvp file even if you aren't using some of the functions.   It can save hours of frustration trying to figure out why something doesn't work.

A copy of RR.txt is also included in baggage of this help file.

Click here to extract a copy of RR.txt to your hard disk

Commercial Version Functions

```
{ewc TSTOOLS, TsButton, "" [name=m2][macro=ji(qchPath,`ContentManagement')]
[graphic=`m2up.dib',`m2dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m3][macro=ji(qchPath,`DialogManagement')]
[graphic=`m3up.dib',`m3dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m4][macro=ji(qchPath,`HelpManagement')]
[graphic=`m4up.dib',`m4dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m5][macro=ji(qchPath,`IniManagement')]
[graphic=`m5up.dib',`m5dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m6][macro=ji(qchPath,`HelpManagement')]
[graphic=`m6up.dib',`m6dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m7][macro=ji(qchPath,`HelpManagement')]
[graphic=`m7up.dib',`m7dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m8][macro=ji(qchPath,`HelpManagement')]
[graphic=`m8up.dib',`m8dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m9][macro=ji(qchPath,`HelpManagement')]
[graphic=`m9up.dib',`m9dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m10][macro=ji(qchPath,`HelpManagement')]
[graphic=`m10up.dib',`m10dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=m11][macro=ji(qchPath,`HelpManagement')]
[graphic=`m11up.dib',`m11dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=y12][macro=ji(qchPath,`HelpManagement')]
[graphic=`m12up.dib',`m12dn.dib',`y1up.dib']/B0/D0}
{ewc TSTOOLS, TsButton, "" [name=y13][macro=ji(qchPath,`HelpManagement')]
[graphic=`m13up.dib',`m13dn.dib',`y1up.dib']/B0/D0}
```

Not written

Commands	are strings specifying a command or commands to run if the condition is true/false. To specify multiple commands, insert a semicolon (;) between each command. If the command(s) contain string parameters, use single open quotes (`) and close quotes (') to delimit the string parameters. If the commands contain paths, use double backslashes (\\) or single forward slashes (/) to represent each backslash in the path. The total length of a Viewer command string cannot exceed 512 characters unless TsMacro is used.

TouchSend Tools - Content Management

TsAppendString	copies and appends a specified string to an existing file.See also TsSaveString.
TsCopyBaggage	copies a specified file from baggage to the clipboard.
TsCopyString	copies a specified string to the clipboard. Useful to provide syntax to the user. used in TsTools documentation to copy the current function syntax or the current function register routine to the clipboard.
TsSaveBaggage	extracts a file from baggage and saves it to disk using a file name specified by the author. **Click here for a demo**
TsSaveBaggageAs	extracts a file from baggage and opens the save dialog box to allow saving of the baggage file to disk at a location and with a name as specified by the user.
TsSaveString	copies a specified string to a file. Useful to easy access to programs that are using Viewer as a lookup facility. For example - a parts program running in Foxpro could use Viewer for context sensitive parts lookup. The user would then navigate through a Viewer Catalog to get a part number. Click on the part number would save it to a file. Upon returning to the Foxpro program, it would read the parts in from the file. The file name is author definable and the text saved can be appended or overwritten at the author's option.
TsSaveTopic	copies the current topic to a file.

TouchSend Tools - Dialog Management   Functions

TSInfoBox	is a dialog box (select one of 5 styles/size/location) with up to 3 lines of text of 50 characters, an author defined titlebar, and the color of the box definable by the author.
TsYN	is a dialog box with all of the same properties as tsinfobox but with two buttons (yes & no) which submits commands dependent on the button pushed.
TsSelectFile	brings up the common dialog box and allows the user to select a file and save the file and path to an ini entry. It is particularly useful in allowing the user to select a wordprocessor. **Click here for a demo of TsSelectFile**

```
 {ewc TSTOOLS, Tsbutton, "Click Here to see the results~of the TsSelectFile Demo"
[Name=clickdemo][Macro=TsExec(`Notepad.exe',`tstools.ini',`demoselect',-
1,1,1);TsExecPos(`demoselect',321,70,310,410);TsDisableButton(`clickdemo')][Font="Arial"/
S11/B4]/B1/D2/-/2}
```

TouchSend Tools - Help Functions

The Help Management Functions call context sensitive help and include:

TsHelpContents	calls a help file and opens at the contents screen.
TsHelpContext	calls a help file, opening at a context id number that has been defined in the help [MAP] section.
TsHelpQuit	quits the help file unless it has been called by some other application.
TsHelpKeyword	displays the topic in the keyword list that matches the keyword passed if there is an exact match. If there is more than one match, it displays the first topic found. If there is no match, an error message is displayed.
TsHelpSearch	displays the topic in the keyword list that matches the keyword passed. If there is more than one match, it displays the Search dialog box with the topics that match. If there is no match it displays the Help search dialog box.
TsHelpPopup	displays in a popup window the topic identified by a specific context string. The main Help window is not displayed.

TouchSend Tools - IF Management Functions

The TsIf Functions incorporate the **INI Management** capabilities and simplify the authoring of *IfThen* and *IfThenElse* commands by incorporating the *IsMark* and *Not* commands. In addition, by using TsGotoMark and TsDeleteMark in place of the built in commands the warning messages when using undefined marks (or marks that have been deleted) are eliminated.

TsIfMark	This function is the equivalent of IfThen(IsMark(\`markname'),\`Command') but is shortened and simplified to TsIfMark(\`markname',\`command').
TsIfNotMark	This function is the equivalent of IfThen(Not(IsMark(\`markname')),\`command') but is shortened and simplified to TsIfNotMark(\`markname',\`command').
TsIfMarkElse	This function is the equivalent of IfThenElse(IsMark(\`markname'),\`command',\`command') but is shortened and simplified to TsIfMarkElse(\`markname',\`command',\`command').
TsIfNotMarkElse	This function is the equivalent of IfThenElse(Not(IsMark(\`markname')),\`command',\`command') but is shortened and simplified to TsIfNotMarkElse(\`markname',\`command',\`command').

TouchSend Tools INI Management Functions

The TsIni Functions are designed to allow titles to be "customized" by the user and to save that information and the "state" of the title between sessions. As well the ini capabilites are used with TsVariable management. These functions extend the Viewer mark commands to offer persistance of data between sessions (using the INI file).

TsToolsInit	initializes the TsTools DLL and sets the name of the INI file that is used with the title. If the INI file exists, it will be used, otherwise it will be created. **If TsToolsInit is not called, the entire dll will not work.**
TsCreateIni	creates an ini file.
TsDeleteIni	deletes an ini file.
TsDeleteIniEntry	deletes an entry from an ini file.
TsDeleteIniSection	deletes a section from an ini file.
TsWriteIni	writes a TsVariable to an ini file.
TsReadIni	reads a TsVariable from an ini file.
TsWriteIniString	writes a string to an ini file.
TsWriteIniNumber	writes a number to an ini file.

TouchSend Tools - Mark Management Functions

Mark management allows the author to manage marks in the same manner as the Viewer Marks commands but the information is also saved to the INI file created as part of the TsInit Function. These marks will automatically be restored upon the title restarting unless they are cleared with the TsClearMarks function. Thus, the state of the title can be saved and restored. See also the **TsIf Functions**.

TsClearMarks	clears all marks from the ini file.
TsGotoMark	goes to a mark saved in the ini file. Must be used to go to a location after the file is exited and restarted. The Viewer GotoMark function will not find the mark as it is not restored to its original location.
TsIsMark	tests to see if a mark is in the ini file and in memory.
TsDeleteMark	deletes a mark from the ini file and from memory.
TsSaveMark	saves a mark to the ini file and to memory.

The **TsSaveMark Demo** saves the mark **"This is a demo mark"** to the Tstools.INI file. The **TsDeleteMark Demo** then deletes the mark. In each case the Tstools.INI file is displayed in Notepad. Notepad is called, positioned and displayed using the **TsExec** and **TsExecPos** Commands. **Click here to restart Mark Demo**

The **TsSaveMark Demo** saves the mark **"This is a demo mark"** to the Tstools.INI file. The **TsDeleteMark Demo** then deletes the mark. In each case the Tstools.INI file is displayed in Notepad. Notepad is called, positioned and displayed using the **TsExec** and **TsExecPos** Commands.

{ewc TSTOOLS,TsButton,"TsSaveMark~Demo."/AR/ML30R3/H40/W130/B2/D1/N/2 [name=markdemo] [graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC][macro=TsClearMarks();PositionWindow(0, 0, 512, 1024,0, "Main");TsSaveMark(`This is a demo mark');TsExecKill(`demo1');TsExec(`Notepad.exe',`tstools.ini',`demo',0,1,1);TsExecPos(`demo',321,75,310,405);TsEnableButton(`markdemo1');TsDisableButton(`markdemo');pi(qchPath,`markdemopopup>markpop')][Font="Times New Roman"/S13/B4/3-]}{ewc TSTOOLS,TsButton,"TsDeleteMark~Demo."/AR/ML30R3/H40/W130/B2/D2/N/2/-[name=markdemo1] [macro=TsDeleteMark(`This is a demo mark');TsExecKill(`demo');TsExec(`Notepad.exe',`tstools.ini',`demo1',0,1,1);TsExecPos(`demo1',321,75,310,405);TsEnableButton(`markdemo');TsDisableButton(`markdemo1')][graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC][Font="Times New Roman"/S13/B4/3-]}{ewc TSTOOLS,TsButton,"End TsMark~Demo"/AR/ML30R3/H40/W130/B2/D2/N/2 [name=markdemo2][graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC][macro=TsDeleteMark(`This is a demo mark');TsExecKill(`demo1');TsExecKill(`demo');PositionWindow(0, 0, 1024, 1024, 1, "Main");CloseWindow(`markdemo');IfThen(ismark(`online'),JumpID(`tstools.mvb>returnrm', `returntobtn2')');deletemark(`online')][Font="Times New Roman"/S13/B4/3-]}

TouchSend Tools - MCI Management Functions

TsMci	submits a command upon the completion of an mci event. Without this function, Viewer has no facility for acting upon completion of a wave file or a video. As an example, the author may present an opening "splash screen" with a vivid graphic and music and then upon completion of the music desire to jump to a table of contents. This function could get lost in the mob of descriptions but is extremely important. This function changes the way Viewer works because it does some thing that most authors want at one time or another but Viewer does not do, which is respond at the end of a sound or a video. Another must remember function is **TsTimer**
TsSnd	tests for a sound device, and then submits a command if a valid sound device is present, and optionally a different command if a valid sound device is not present. Without this function Viewer will display an error message if a wave file is submitted but there is no sound device. As well this function provides the author with an opportunity to provide an alternate means to communicate if a message was being delivered by sound. As an example, a voice saying "*press any key to continue"* could be replaced with a pane presenting the same information. This function has been a life saver on a few occasions.
TsWave	submits a wave file and will continue to play it until it is completed or until it is stopped by the author. Normally Viewer will stop a wave file if the user jumps to a different topic.

TouchSend Tools - Miscellaneous Functions

TsAbsolute	sets the state for drawing TsButtons and TsPanes. The state will either be 640 by 480 (absolute size) or relative on a 1024x1024 grid. By default TsAbsolute is on.
TsClose	sets a flag that will ensure that the topic.present when a title is exited will be the startup topic upon reentering the title. This function allows a user to restart at the exact topic he or she was using at the time of exit.
TsDebug	turns TsTools error message on or off.
TsExitTopic	submits commands upon exiting a topic. However, Viewer will not reliably execute all of these commands before executing topic entry commands at the topic jumped to so care must be taken with authoring to ensure there is no conflict of commands if Viewer submits them out of order.
TsGlobalMacro	executes a command or set of commands upon entering every topic until turned off by calling TsGlobal Macro with a null string. There can only be one TsGlobal Macro command string active at one time.
TsHideCursor	will hide the cursor. This function must be used with care or the cursor will disappear for the balance of the title. It is often used to remove a cursor from a large video display to avoid blinking. When used all methods of exit from the topic must used the TsShowCursor command.
TsMacro	The TsMacro Function acts like a subroutine. It submits a string of macros of up to 512 characters without having to enter a topic or a group. This function effectively overcomes the 512 character limitation of a topic entry macro or a group macro, by using multiple calls to TsMacro. The macros to be submitted are placed in "script files" that are located in baggage or on the hard disk or CDRom.
TsShowCursor	will restore a hidden cursor.
TsResources	when called displays the available windows resources. Useful for debugging during title authoring. It is also helpful if support for TsTools is required as it will tell support the "state of the title" at a problem location. Click here for TsResources Demo
TsToolsInit	initializes the TsTools DLL and sets the name of the INI file that is used with the title. If the INI file exists, it will be used, otherwise it will be created. If the function is not called as the first function in the configuration script, none of the TsFunctions, TsButtons or TsPanes will operate properly.
TsWinStyle	allows the author to change the style of a window. This function works with both the main and secondary windows. Click here for a WinStyle Demo

TsWinStyle Demo

To properly use this function in a Main Window, make sure the Menu and Buttonbar are turned off or this technique is unreliable. If a button bar is desirable, Tsbuttons in the nonscrolling region can be used in place of the Viewer Buttonbar. Note that "hwndContext" and **NOT** "hwndApp" must be passed to the function.

{ewc TSTOOLS, Tsbutton, "Thick Frame" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,0,0,0,0,0,1)]/H20/w100/B1/D1/N} {ewc TSTOOLS, Tsbutton, "Caption Only " [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,1,0,0,0,0,1)]/H20/w100/B1/D1/N}

{ewc TSTOOLS, Tsbutton, "No System Menu" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,1,1,0,1,1,0)]/H20/w100/B1/D1/N} {ewc TSTOOLS, Tsbutton, "Thin Frame" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,0,0,0,0,0,0)]/H20/w100/B1/D1/N}

{ewc TSTOOLS, Tsbutton, "No Styles" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,0,0,0,0,0,0,0)]/H20/w100/B1/D1/N} {ewc TSTOOLS, Tsbutton, "Min/Max Off" [Font="Times New Roman"/S11/B3/3-] [Macro = TsWinStyle(hwndContext,1,1,1,1,0,0,0)]/H20/w100/B1/D1/N}

{ewc TSTOOLS, Tsbutton, "All Styles On" [Font="Times New Roman"/S11/B7/3-] [Macro = TsWinStyle(hwndContext,1,1,1,1,1,1,0)]/H20/w100/B1/D1/N} {ewc TSTOOLS, Tsbutton, "Exit" [Font="Times New Roman"/S11/B7/3-] [Macro = closewindow(`third')]/H20/W100/B1/D1/N/3+}

TouchSend Tools - Print Management Functions

Viewer doesn't handle printing very well. Due to the inherent limitations of the Viewer print engine, after a print command is submitted, if the author issues a jump command, more often than not, Viewer prints the page jumped to rather than the page on which the print command is submitted. **TouchSend Tools** provides the following functions to give author complete control over printing from a title. In addition the baggage extraction functions allow an author to deliver files to the user for printing through standard Windows applications such as a wordprocessor.

TsPrintAfterJump	jumps to a topic, prints the topic and then submits another command to Viewer after the printing is complete. Click Here for a TsPrintAfterJump Demo
TsPrintFromList	prints a list of topics the list being either in baggage or on the disk, submitting each to the printer and when complete jumps to and prints the next topic in the list until all topics in the list are completed.
TsPrintGroup	prints all of the topics in a Viewer defined group.
TsPrintThenCmd	prints the current screen and then submits a command after the printing is complete.
TsKillPrint	this function allows the author to kill a printing that has been launched with either TsPrintFromList or TsPrintGroup.
TsSync	tests to see if the printer is currently printing. When executed the function will prevent any other commands in the macro command string from being submitted to the Viewer engine until the printing of the current topic is completed.

TouchSend Tools - Print Management   Functions Demo

This page was printed using the TsPrintAfterJump Function Call.

This command is a very convenient way to print any kind of a form, or example page that is separate from the current topic.

TouchSend Tools - Task Management Functions

The **TsExec** task management functions allow for the launching other Windows applications and if specified, monitor their state. A program can be launched without knowledge of its location or path. The launched program can be **dependent** (ie a child of the Viewer title - will minimize or restore with Viewer and terminate upon termination of the Viewer title) or **independent** (will reside independent of the Viewer title). With some limitiations, keystrokes can be submitted to the other application for pasting, executing macros etc. This suite of commands is extremely useful for managing Visual Basic applications that provide additional functionality to Viewer but are not designed to be a "Viewer Shell" such as listboxes etc. It is also very useful for using Word Processors or Spreadsheets in concert with a Viewer title to extract data "on the fly" optionally with formatting instructions via pasting and then the implementation of macros. Each application launched by TsExec has a "name" and as such the user can manage and work with multiple launched applications concurrently.

Click here for a demo of the TsExec Functions

TsExec	Executes any Windows Exe file that is available on the system. The function will search the active path and the [Programs] section of Win.ini to find the executable if it is not in the current working directory. If the program is still not identified, a dialog box will pop up to allow the user to select the appropriate executable. TsExec will also execute a program that has been defined by name in the [Exec] section of the INI file associated with the TsToolsInit function for that title.
TsExecKill	Will shut down an application that has been launched using TsExec.
TsExecState	Sets the "state" of any application that has been launched using TsExec. The states available are Normal, Minimized, Maximized, Show Not Active; Show in its pre minimized and maximized state and make it the active application; minimized and active;show minimized;show not active and restore.
TsExecPos	Sets the size and position of the executed application. This function acts in a manner similar to the Viewer PositionWindow command.
TsExecPaste	Will paste the current contents of the clipboard into the application launched by TsExec. This function is not reliable with some applications including Word for Windows and Ami Pro.
TsExecSendKeys	Submits keystrokes to the named application. This function is not reliable with some applications including Word for Windows and Ami Pro.
TsExecSetFocus	Sets focus to an application launched by TsExec.
TsExecSetZ	Manages the Z order of any application launched with TsExec. An application can be set as any one of bottom of the Z order, not a topmost window, a top window or a topmost window. The Z order can be changed by a further call to this function.
TsExecV	Executes any Windows Exe file that is available on the system and has been defined with a variable alias in an ini file. The variable alias is completely controlled by the author. For example, using **TsSelectFile**, an author may have a user chose a default wordprocessor. TsExecV will then execute the variable "wordprocessor" which in the ini file would have the following format: *wordprocessor=path\executable.exe.*
TsViewerSetFocus	Returns focus to the Viewer title it has been set to a launched application using TsExecSetFocus.

TouchSend Tools - Task Management Demo

Each of the below listed commands will execute when clicked. They demonstrate some of the capabilities of the TsExec task functions. **Note:** Before killing the calculator, minimize the title and then restore it - the calculator has been called as a dependent or child application and will minimize when Viewer does. The **TsCalls Pane** just counts the number of times any of these demo functions are accessed. This count can be saved between sessions altho is is not saved in this demo.

{ewc TSTOOLS, TsPane, "calcvar"[Name=calcdemo][Font="Times New Roman"/S12/B4][Text=`The calculator number is: ',`calcvar',`.'][macro=TsInfobox(4,`TsPane Demo',` This TsPane is initially set to a value of 0',` It is then updated by calling TsVPaste which pastes whatever ',` variable is on the clipboard to TsPane.',255,255,255,`')][graphic=`!panedbig.dib'][Color=192,192,192,128,0,0]/w200/h30/b1}

{ewc TSTOOLS, TsPane, "countvar"[Name=countdemo][Font="Times New Roman"/S12/B4][Text=`Ts Calls: ',`countvar',`.'][macro=TsInfobox(3,`TsTask Counter',`This TsPane is initially set to a value of 0',`It is then updated by any time any of the',`Hot Spots on this page are clicked.',255,255,255,`')][graphic=`!panedemo.dib'][Color=192,192,192,0,0,128]/w78/h30/b0}

TsExec(`calc.exe',`',`Calculator',-1,1,1) Bring up the calculator as a stay on top app.
TsExecPos(`Calculator',369,287,-1,-1) Change the Position of the Calculator.
TsExecSendKeys(`Calculator',`35*40*52=')Send some keystrokes to the calculator to calculate a number.
TsExecCopy(`Calculator') Copy the calculator result to the clipboard.
TsVPaste(`calcvar') Paste the contents of the clipboard to `calcvar' in the Tspane shown above.

(Feel free to change the numbers on the calculator by using it directly -then click the TsExecCopy and TsVPaste examples to update the TsPane. The calculator is a "stay on top" application - see Z order description in TsSetWriteZ

TsExecState("Calculator",2) Minimize Calculator.
TsExecState("Calculator",1) Restore Calculator.
TsExecSendKeys(`Calculator',`%vs') Set Scientific mode.
TsExecSendKeys(`Calculator',`%vt') Set Standard mode.
TsExecKill("Calculator") Kill the Calculator. Paste the Number of Calls to Notepad

TouchSend Tools - Timer Management

TsTimer	Submits a command after the passage of time. TsTimer will be killed in the event of a jump. Tstimer can be used to do self running demos, or presentations that can be converted into interactive presentations at any time. They are also great for causing an action to take place if a user doesn't do something for a given time interval. (Example:- if a user doesn't select anything for two minutes pop up a menu). Timers can be embedded to create multiple timer sequences. Like **TsMCI** it is easy for this function to get lost in the list. This function changes the way Viewer works because it does some thing that most authors want at one time or another but Viewer does not do, which is submit commands to the Viewer engine over time. The beginning sequence of this title with the 4 panes opening was done with TsTimer. See also the TsTimer demo in the functions section of this title which illustrates how to use TsTimer to create time sequenced demos.
TsKillTimer	Immediately kills any TsTimers that are currently running.

TouchSend Tools - TsButton Management

Each of the below listed functions offer programatical control of any **TsButton**. These functions give the author complete flexibility and control of the *"look and feel"* of their Title. As well, TsPressButton completely overcomes the impression in the Viewer Manual that an embedded pane cannot be attached to an Accelerator key.

TsPressButton:	Allows the author to programmatically "click" any **TsButton**.
TsEnableButton:	Will enable a button that has been disabled with the TsDisableButton function.
TsDisableButton:	Will disable a button. On a text button the text will appear light grey and the button will not click nor will any attached macro be executed. On a graphics button, the specified graphic for the disabled state will displayed, the button will not press and any attached macro will not be executed. On a mixed text and graphics buttons, the combined effect is implemented.
TsHideButton:	Will hide a button.
TsShowButton:	Will make a button that has been hidden with TsHideButton, visible.

TouchSend Tools - Variable Management

The TsPane embedded window provides text and variable output within a topic page.   Using the TsVariables capability, the author can keep track of any number of counts, including scores, "first trys", lookups, and the results can be reported concurrently in the topic page, in a secondary window, popup or pane and can be updated continuously or as an event.   The TsVariables allow for incrementing and decrementing counters. The **TsIni** functions allow for the variables to be saved between sessions.   The **TsMarks** functions allow the author to set and maintain states between sessions (example - beginner,intermediate, expert and have the title respond accordlingly).   See **TsPanes** for details on display capabilites.   See also **TsExecV**.

The functions are:

TsVAdd	increments a TsVariable by an author defined amount.
TsVAppend	copies and appends the value of a TsVariable to a file.
TsVCatS	concatenates string with the current named TsVariable
TsVCatV	concatenates variable with the current named TsVariable and sets total to first named TsVariable.
TsVCopy	copies value of a named TsVariable to the clipboard as text.
TsVCopyS	copies a string into a named TsVariable.
TsVCopyV	copies a value into a named TsVariable.
TsVIfEQ	submits a command if a named TsVariable equals the specified value. (This function does not have automatic like TsVOnCount it only acts if the function is called and the variable is equal to the specified value at that time).
TsVIfGT	submits a command if named TsVariable is greater than specified value (This function does not have automatic like TsVOnCount it only acts if the function is called and the variable is greater than the specified value at that time).
TsVIfLT	submits a command if named TsVariable is less than specified value (This function does not have automatic like TsVOnCount it only acts if the function is called and the variable is less than the specified value at that time).
TsVIfWithin	submits a command to Viewer if within a "range" of the specifiec value (This function does not have automatic like TsVOnCount it only acts if the function is called and the variable is within the specified range of the specified value at that time).
TsVOnCount	submits a command upon reaching the counter as set without any further function call. This function "sets" an action to take place when the TsVariable value reaches the count value by whatever means.
TsVPaste	copies text on the clipboard to a named TsVariable.
TsVRandom	creates a TsVariable that is a random number.
TsVSave	copies the value of a TsVariable to a text file.
TsVSet	sets a named TsVariable to a value.
TsVSub	decrements a named TsVariable by an author defined amount.

TouchSend Tools - Pane Display of Variables

This is an embedded TsPane:

```
{ewc TSTOOLS, TsPane, "demopane"[Name=calcdemo][graphic=`!panestat.dib'][Font="Arial"/S11/B3]
[Text=`This is a status pane - Score: ',`demopane',`.'][Color=192,192,192,0,0,0][Macro=TsInfobox(3,`TsPane
Demo',`',`     TsPane allows the author to send information',`     to embedded panes in the title such as scoring or
progress.',192,192,192,`') ]/w200/h40}
```

Click here to increment count
Click here to decrement count

The author has control over the following items in creating and managing TsPanes: *font, multiple lines, color of pane and color of text, graphic display in the pane, text to be displayed in the panes and variables that can be displayed in the panes*. The author also has control of *height, width, margins, text starting location, text alignment and whether or not the pane will print*. As well a macro can be attached to the pane which will execute when the pane is clicked. Click it and see.

The Ancillary Pane Functions are:

TsHidePane:	Will hide a pane. Click here to hide the pane
TsShowPane:	Will make visible a pane that has been hidden with the TsHidePane command. Click here after hiding the pane to restore it
TsUpdatePane:	Forces an update of the current display pane or panes.

TsClose Demo

TsClose has been turned on.
Now close the title from any topic.

Upon restarting this title it will
open at the location it closed.

Note;

It is worth taking the time to click through every **TouchSend Function**,   but it can be a bit overwhelming at first.

The **TouchSend Functions** will make the most sense after you have read the book and tried the examples.

For a quick look at some of the interesting extensions of TsTools, click the **Don't Miss** button on the button bar.

{ewc TSTOOLS, Tsbutton, "Hug Me" [graphic=`!babyup.dib',`!babydn.dib',`!babyup.dib'][Font="Times New Roman" /S10/B4/I]/B0/D0/MT60}

Because the entire demo is quite extensive this popup provides quick access to selected highlights to illustrate some of the extended capabilities of Viewer using the TouchSend Tools

Graphics Buttons to simulate a toolbar
Mixed Text & Graphics Buttons
Mouse Move
Pane Management
Selecting a default Wordprocessor
TsExec Demo
WinStyle Demo

Selecting a Wordprocessor

Often an author would like a user to be able to select a wordprocessor to use in conjunction with a title and have the user be able to call it up from thereafter.   Here is how to do it with TsTools:

(click each of the steps below)

1.   **Call TsSelectFile**   to select a wordprocessor of choice.   For simplicity try selecting Write or Notepad, although actually any executable file will do.   The selected file will be saved with its path in the ini file "Tstools.ini" as "wordprocessor = path\file.exe"

2. **Launch the Selected Wordprocessor** using the TsExecV command (which executes a program using its TsVariable name rather than its program name.

3.   **Kill the Word Processor** using the TsExecKill Function.

```
{ewc TSTOOLS,TsButton,"Return to ~TsExec Demo.     "/AR/ML30R3/H40/W110/B2/D1/N/2 [name=returncalc]
[graphic=`!tsblgoup.dib',`!tsblgodn.dib',`!tsblgodi.dib'/ALC]
[macro=IfThenElse(ismark(`readme'),`JumpID(`tstools.mvb>returnrm',
`returntoreadme');ji(qchPath,`ExecDemo');DeleteMark(`readme')',`DeleteMark(`readme');ji(qchPath,`ExecDemo');C
loseWindow(`returnrm')')][Font="Times New Roman"/S13/B4/3-]}
```

{ewc TSTOOLS, Tsbutton, "" /b0/d0[graphic=`!babyup.dib',`!babydn.dib',`!babydi.dib']

[name=hugme][macro=tsdisablebutton(`hugme')]}

Viewer is a wonderful tool

However...

When I first rolled up my sleeves to do a title I had the following challenges:

1. I had to learn the Viewer "language".
2. I wanted my titles to be as good as or better than Microsoft's Encarta.

After struggling through the inevitable learning curve, I at first, felt that I was making magic, but as my skills grew, I wanted to do more than Viewer would let me, and I wanted to do some of the things I had seen in Encarta, Cinemania and Dinosaurs (to name a few) as well as a few creative ideas of my own.

I felt I had 3 choices

1. I could look for another authoring environment. (The alternative choices weren't that appealing and were expensive, and constrained).

2. I could put a Visual Basic "wrapper" around every title. I am fluent in Visual Basic but for a number of the functions, it seemed to be a lot of work and for others, such as embedded panes like TsButton it was impossible (*But, while I enjoy programming, when I was being creative and authoring titles, I didn't want to be writing software, I wanted to create great titles without the clutter of programming*.)

3. I could develop a set of custom functions that would "flesh out Viewer".

I decided to build a few tools. That soon became a lot of tools.

I am happy to report that to date, virtually every serious Viewer developer that has seen the pre-release version of the TouchSend Tools has ordered a set.

In fact, today I found out that Microsoft has purchased a set of the TouchSend Tools through an independent Viewer developer to be used in building the Microsoft University Course Catalog as a Viewer title. *(That made me **VERY HAPPY** !!!).*

To quote one experienced Viewer developer: "*TouchSend Tools allow Viewer Authors to do extensive customization without intensive programming".*

My challenge today is how to explain to new inductees to Viewer just what the tools deliver because new users wouldn't know what Viewer could or could not do. Harking back to my learning curve challenges I decided to provide a fast look at some of the more interesting functions.

I created the "don't miss" button on the TsToolsW Online Section. With those quick access demos, and keeping TsTimer, and TsMCI in mind one gets a fairly good sense of what the Tools can let you do. Suffice it to say, the 100 or so functions I have provided are not for the most part otherwise directly available in Viewer.

With the TouchSend Tools any creative title author (most of who have no interest in being programmers) can do what multimedia delivery systems are designed for, namely: to make exciting, artistic, satisfying and informative titles whether they are for entertainment, education or reference. If you are a Visual Basic programmer, consider the tools just like getting a whole set of VBX's. They add additional functionality to your arsenal. *As well we are in the process of building a set of functions to be called directly through **Visual Basic** called the **TouchSend VBTools** which will allow any Visual Basic programmer to directly access the functionality of the TouchSend Tools from Visual Basic without the need to write and manage several thousand lines of code.*

I would like to thank **Steve Pruitt** for his inspiration and encouragement throughout the last many months. Steve thought it would be great if his readers could have a very flexible embedded button to enjoy and a custom "About" box. I have added some other freebies because they are fun and as well to provide a sense of how useful Viewer extensions can be.

We have built things that we were told could not be done. I hope you enjoy them. If you like to program and build anything that you think is really fantastic, please contact me. I would love to see your work.

I would also like to thank my wife **Leah** who kept telling me we could make it happen. She was and is right. **So thanks Leah, and Happy Valentines day !**

Jeff Kovitz
Valentines Day, 1994

topic

topic1