DRAFT 12/21/22

MPW PQR4 ToolServer
Release Notes

ToolServer—A Reduced Shell for Background Execution

Introduction

ToolServer, like the MPW Shell, is an application. It is a subset of the MPW Shell, designed to execute time consuming, non-interactive tools or scripts in the background, permitting the user to run the MPW Shell (or any other application) interactively in another MultiFinder partition. ToolServer's I/O is to files; it has no awareness of windows and, in fact, cannot reference the contents of a window before the window has been saved. In general, the subset of the MPW commands which are supported by ToolServer are those which are not restricted to interactive application. The list of commands supported and not supported by ToolServer appears in the **Script Execution** section of this Appendix.

When running under System 7, a running ToolServer can send messages to or receive messages from the MPW Shell via Apple Events.

Launching

Like any application, ToolServer can be launched from the Finder or from a running MPW Shell. Under System 7, it can also be launched by the command `RShell` (see **Apple Events** below). As a non-interactive facility, ToolServer must execute a script if it is to do anything useful. There are two ways under System 6 to specify one or more scripts to be run; under System 7, `RShell` provides a third way. When launching from the MPW Shell, the scripts to be run can be specified on the command line by simply naming them, i.e. by using the syntax

```
ToolServer [ ScriptName ... ]
```

If a running ToolServer is brought to the front in MultiFinder, a menu item called “Execute Script...” (Command-E) will be found under the File menu.Choosing this item will open a dialog for selecting a script to be executed. This item is available regardless of the launching method which had been used.

ToolServer, immediately after launching, executes the supplied script `StartupTS`. The latter executes, in turn, `UserStartupTS` and any files that may be present whose names start with `UserStartupTS•.` The organization of these startup files parallels that of the corresponding MPW files. The file `StartupTS` differs from the MPW file `Startup` in that it executes `UserStartupTS` and `UserStartupTS•≈`. The assumption is made that ToolServer and all the startup files live in the same directory as the MPW Shell. If the user wishes to change this, it will be necessary to perform editing of the `StartupTS` script. In the current release, `UserStartupTS` is empty; it serves as a place holder for customizing the startup.

Script Execution

Before executing a script, ToolServer sets the working directory to be the one containing the script. Two new variables, `BackgroundOut` and `BackgroundErr` name the files to which standard output and standard error are to be redirected. For each of these that is set to Dev:Null, the corresponding file will not be generated. For each of these that is left undefined, the corresponding file will receive a default name. These are, respectively, `<scriptname>.Out` and `<scriptname>.Err.` If there is a naming conflict caused by an open window of the desired name, a new unique name is generated by appending to the name an ordinal, e.g.: “-0”, “-1”, etc.

Scripts, as MPW documents, normally have the creator 'MPS '; double-clicking on them launches MPW. If the user wishes ToolServer to be launched instead, SetFile can be used to change the creator to 'MPSX', the designated creator name for ToolServer.

While ToolServer is executing a script (and if it's in the foreground), the beach ball cursor is displayed. It is expected that most users would run ToolServer in the background. ToolServer does not display windows or interact with the user in any other way. Errors will usually be reported to standard error. However errors that occur in attempting to open standard error or during initialization may result in an alert. This alert will be displayed through the Notification Manager. Users can interrupt ToolServer by using MultiFinder to bring it to the foreground, and entering “Command-.” from the keyboard. If a user desires notification when a script is complete, the script could contain the MPW command "Alert". This command will use the Notification Manager to display the message specified by the alert.

ToolServer will set the shell variable `BackgroundShell` to 1, so that scripts can determine if they are running under the ToolServer or MPW Shell.

ToolServer is persistent and will remain in the background waiting for scripts to run until either Quit is selected from the file menu, a script containing the `Quit` command is run, or the "Quit" Apple Event is received (under 7.0).

Scripts may call any MPW tool and may use any of the following built-in commands:

```
Begin...End          Catenate             Alias
Break                Delete               Date
Continue             Directory            Echo
Exit                 Duplicate            Evaluate
For..                Eject                Execute
If...                Equal                Export
Loop...End           Erase                Flush
                     Exists               Help
Alert                Files                Parameters
Beep                 Mount                Quit
Confirm              Move                 Quote
Request              Newer                Set
                     NewFolder            Shift
                     Rename               ShutDown
                     SetFile              Unalias
                     Unmount              Unexport
                     Version              Unset
                     Volumes
                     Which
```

Scripts may further include variables, aliases, substitution characters, quotes, wildcard operators, and redirection commands.

Projector and editor commands are not supported, and attempts to execute them will set the *Status* variable to 1. These commands include:

```
Add/DeleteMenu       Browser              Adjust
Add/DeletePane       CheckIn              Align
Close                CheckOut             Clear
MoveWindow           CheckOutDir          Copy
New                  DeleteRevisions      Cut
Open                 ModifyReadOnly       Find
RotateWindow         MountProject         Format
Save                 NameRevisions        Mark
SaveOnClose          NewProject           Markers
```



```
ShowSelection          Project                Paste
SizeWindows            ProjectInfo            Position
StackWindows           UnmountProject         Replace
Target                                        Revert
TileWindows                                   Undo
Windows                                       Unmark
ZoomWindows
```

An example of a script for building a tool is:

```
Directory 'HD:MPW:Demo:'
Begin
      echo "Beginning Build"
      date
      make -f Demo.make > domake.cmd
      set done 1
      set echo 1
      domake.cmd ||begin; alert 'Make failed!!!'; unset done; end;
      set echo 0
      date
End ∑ 'Hd:MPW:Demo:domake.results'
If {done}
      Alert "Build is complete.∂nSee HD:MPW:Demo:domake.results"
Else
      Alert "Build had errors.∂nSee HD:MPW:Demo:domake.results"
End
Quit
```

Apple Events

Communication between an MPW Shell and ToolServer is provided by Apple Events. This can be exploited either by using the new Shell command `RShell`, or by direct use of either of two MPW specific Apple Events.

RShell

The provisional version of MPW that is in the PQR folder of ETO 4 supports a new command named `RShell`. This command, usable only when running under System 7, sends commands from one shell to another via Apple Events. The syntax is:

```
RShell [ command | -q | -q command ] [-c id] [-k id]
      [ -f | -b | [-r [[zone:] server:] application] ]
```

The parameter meanings are:

```
command     Send the given command to the target application.
-q          Quit.  If combined with a command, the command
              is executed first.
-f          Send the command to the foreground (MPW Shell).
-b          Send the command to the background (ToolServer).
-c id       Close the file associated with the request
```


whose transaction id is *id*.

-k *id* Abort the script whose transaction id is *id.*

-r *pathpath* is the network path to a remote application.

A `command` is any string that could have been executed in the current context by the MPW Shell had it been typed in a window on a line by itself and activated by pressing "Enter."

If neither `-f`, `-b,` nor `-r` are on the command line, a dialog appears (PPC Browser) enabling interactive selection of an application to receive the command. In the *`path`* parameter for the `-r` option, the terminal application name is case sensitive. If `-f` or `-b` is specified, and the target shell is not running, the `RShell` command will launch it.

The `-k` option will abort a script if the `id` parameter matches the script in progress and the command is sent from the same shell that initiated the script. Otherwise, the event is treated as a no-op.

The `RShell` command can also send strings (commands) to applications on remote machines, but cannot cause the remote applications to be launched.

Any material sent by ToolServer to Dev:console will be sent back to the MPW Shell that issued the `RShell` command. For example:

```
RShell 'Echo blap > dev:console' -b >> foo
```

will send the string "blap" back to the MPW Shell, which will in turn append it to the contents of file "foo". If redirection, e.g. to "foo" is not specified, the output will appear in the window from which the `RShell` command was issued. Standard Output and Standard Errors are defaulted to the files MPW.Script.out and MPW.Script.err. If it is desired to have either or both appear in an MPW window, one can set the corresponding variables `BackgroundOut` and `BackgroundErr` to Dev:Console. This can be done in the `UserStartupTS` script or by running:

```
RShell 'Set BackgroundOut Dev:Console' -b
RShell 'Set BackgroundErr Dev:Console' -b
```

The ability to redirect to Dev:Console is restricted to commands sent to ToolServer by the `RShell` command. Any attempt to redirect to Dev:Console in a ToolServer script invoked by the "Execute Script" menu item, or a script invoked by a Finder "Open," is an error.

The `-c` option is available so that the Shell can close the local output file if a previous RShell request fails to terminate normally.. If `-c` *`id`* is combined with other options, the action associated with `-c` is taken first.

`RShell` with no parameters will cause a numbered display of the active requests. The numbers given in this display are those that would be used with the `-c` or `-k` options.

Direct Use of Apple Events

- This section is intended only for those users whose intent is the direct generation from an application of Apple Events directed to Tool Server. For further details, see Chapter 6, AppleEvents, in Volume 6 of *Inside Macintosh*..

This event communicates an MPW command line to be executed by ToolServer. The event is summarized as follows:

Message class: `'MPS '`

Message ID: `'scpt'`

Parameter Keyword: `'----'`

Parameter Type: `'TEXT'`

Parameter Data: string containing command line to execute

This event is used to send to the requester any output which ToolServer has redirected to Dev:Console. It is summarized as follows:

Message class: `'MPS '`

Message ID: `'outp'`

Parameter Keyword: `'----'`

Parameter Type: `'TEXT'`

Parameter Data: console output

This event obtains the current status of an executing script. It is summarized as follows:

Message class: `'MPS '`

Message ID: `'stat'`

If the transaction id specified in the event matches those of the sender and the script, or if the value `kAnyTransactionID` is specified, then the reply `'errn'` is set to `noErr (0)`, and four additional parameters are returned:

`'what'` `typeChar`, the file name of the script in progress
`'pos '` `typeLongInteger`, the number of characters read from the file
`'size'` `typeLongInteger`, the size of the file
`'who '` `typeChar`, the name of the current command or tool (as in the status box)

If the transaction id does not match, then:

If the Shell is busy, `'errn' == ProcNotFound`
If the Shell is not busy, `'errn' == noErr`.

This event causes a script to be aborted. It is summarized as follows:

Message class: `'MPS '`

Message ID: `'abrt'`

As for `'stat'`, this event requires a transaction id match, or a transaction id of `kAnyTransactionID`. Otherwise, the event is a no-op.

This event is defined in the Apple Events Registry. It is similar to `'scpt'`, but instread of generating `'outp'` events, it causes the script output to be buffered by the ToolServer and returned as the direct parameter of the reply. It is summarized as follows:

Message class: `'misc'`

Message ID: `'dosc'`

Parameter Keyword: `'----'`

Parameter Type: `'TEXT'`

Parameter Data: string containing command line to execute

Finally, Tool Server supports the four required AppleEvents. They are:

`'oapp'` which opens an application,

`'odoc'` which executes a script,

`'pdoc'` which prints a document, and

`'quit'` which aborts an executing tool and terminates the ToolServer.

Size

ToolServer is still under development, but it is expected that its size will not exceed 250K. This can be compared to the size of the MPW Shell, which is currently over 500K.