C

Appendix C PRESENTATION SCRIPT COMMAND SUMMARY

Linear vs. Hierarchical Presentations

Proof provides for two kinds of presentations, linear and hierarchical. Linear presentations comprise a list of commands that is executed from top to bottom. A user can jump ahead or jump back in a linear presentation, but only one step at a time. In hierarchical presentations, commands are organized as a group (tree), with optional subgroups (subtrees), allowing the user quickly navigate through a presentation, homing in on areas of interest.

Linear Presentations

If a presentation script file has no group or subgroups, it is by definition a *linear* presentation. Such a presentation is simply an ordered list of commands. Each command is executed exactly once, in sequence, unless a user intervenes.

Hierarchical Presentations

The structure of a hierarchical presentation is shown by Proof in a custom-built Windows Tree View that allows a user to quickly select presentation steps, and reveal or hide the presentation's subgroups

Presentation Navigation Windows and Navigation Trees

When a group statement is executed, a navigation window, whose title is specified in the group command, appears on the screen. Within the navigation window, a navigation tree is displayed. The tree contains items whose names are specified by item commands. The tree may also contain subgroups whose names are specified by subgroup commands. Subgroups are preceded by a small box with a "+" or "-" in it. Clicking on a box which contains a "+" causes the items and lower-level subgroups of the given subgroup to be shown. Clicking on a "-", which is shown for an expanded subgroup, causes the subgroup to be shrunk back to just its title.

The navigation window is hidden when animations are running or slides are being shown, but reappears automatically when needed. The navigation window can be manually hidden or shown by clicking on the navigation tree icon in the toolbar of Proof's main window.

The use of group…endgroup has the following advantages:

- The structure of a presentation is visible.
- By clicking on items in the navigation tree, a viewer can quickly move around in a presentation.
- The use of subgroups allows construction of a hierarchical presentation. By expanding subgroups, a viewer can zoom in on topics of interest. By shrinking subgroups back to their titles, a viewer can zoom out to see the higher-level structure of a presentation.

Presentation scripts without group…endgroup are allowed, but we discourage the use of such scripts, for the following reasons:

- Such scripts offer no visual cues as to their structure.
- One can step back or ahead one step at a time in such scripts, but one cannot jump to arbitrary points in the script.
- Such scripts are inherently non-hierarchical.

Compatibility with Earlier Versions of Proof

Earlier versions of Proof did not provide for hierarchical presentations. In the current version of Proof, the item command has replaced the former case command, group has replaced menu, and endgroup has replaced endmenu. The old command names are still accepted by the current version of Proof, assuring 100% compatibility.

Command Overview

The commands that can be used in a presentation script (.PSF) file fall into two groups: commands that specify the structure of a presentation, and commands that specify actions, e.g., run an animation, show a slide, play a sound file, etc.

The commands that specify actions are as follows:

Command	Action
run	runs an animation or portion of an animation
slide	displays a .BMP, .RTF, or .PCX slide
play	plays a sound (.WAV) file
syscall	invokes another program from within Proof
wait	specifies time delays or waits for user OKs to proceed
wipe	clears the screen (fades, dissolves, etc.)
exit	exits a presentation (can “chain” to another presentation)

The commands that specify the structure of a presentation are as follows:

Command	Function Performed
item	identifies actions performed for a menu selection
group	displays a menu of items and subgroups
endgroup	terminates a top-level group of items
subgroup	identifies the start of a lower-level group of items
endsubgroup	terminates a lower-level group of items
title	places a title in the title bar of Proof’s main window
end	terminates a .PSF file
*	prefixes a comment line which has no effect

The Structure of a Presentation Script

A typical presentation has the following form:

```
title  Presentation Title
[ introductory actions ]
group  Menu Title
            item  Animation 1 Name
                actions
            subgroup  Subgroup 2 Name
                item  Animation 2.1 Name
                    actions
                item  Animation 2.2 Name
                    actions
                subgroup  Subgroup 2.3 Name
                    item  Animation 2.3.1 Name
                        actions
                    item  Animation 2.3.2 Name
                        actions
                    …
                endsubgroup
                …
            endsubgroup
            …
            subgroup  Subgroup 3 Name
            …
            endsubgroup

            item  Exit Presentation
            exit system

endgroup
end
```

Note that while indentation of the commands shown above is not required, it greatly increases the readability of a presentation script. Further note that there is no escape from group…endgroup; i.e., execution of a script does not fall through to the next command when an endgroup command is executed. Execution of an endgroup command forces a presentation's navigation window (described below) to be displayed if it has been temporarily suppressed.

Presentation Script Command Summary

The collection of presentation script commands is described in alphabetical order in the sections that follow.

end

Processing of the presentation script file ends when the end command is encountered. Every presentation script must be terminated by an end command.

The syntax of the end command is as follows:

end [*filename*]

In a presentation with no menu, i.e., no group…endgroup, execution of the end command terminates the current presentation script. If a *filename* is specified, Proof "chains" to the script file specified by the *filename*. If no extension is given as part of the *filename*, .PSF is assumed.

In a presentation with a menu, the end command must appear after the endgroup command. In this case, the end command can never be reached, because it is impossible to jump out of group…endgroup. However, the end command is still required.

endgroup

The endgroup command marks the end of a group of actions and/or subgroups comprising a menu-based presentation.

The syntax of the endgroup command is as follows:

endgroup

endsubgroup

The endsubgroup command marks the end of a subgroup of actions and/or subgroups in a menu-based presentation. A subgroup cannot be used on a freestanding basis; a subgroup must be contained within a group or within another subgroup.

The syntax of the endsubgroup command is as follows:

```
endsubgroup
```

exit

The exit command can appear anywhere in a script file. Its syntax is as follows:

exit [*filename* | system]

A exit command with no *filename* or system keyword terminates the current presentation, but Proof remains active. If exit system is specified, execution of Proof is terminated, and the Proof window disappears. If exit *filename* is specified, Proof "chains" to the script file specified by the *filename* and begins processing the new presentation script. If no extension is given as part of the *filename*, .PSF is assumed.

group

The group command marks the start of a collection of items and/or subgroups. The collection must be terminated by an endgroup command. Execution of the group command causes a navigation window to appear. Items and/or subgroups within the group are shown as entries in a navigation tree.

A presentation script file can have at most one group command. If a presentation script contains no group command, the presentation will execute sequentially until an end or exit command is executed. The syntax of the group command is as follows:

group *text_string*

The *text_string* is used as the title of the presentation's navigation window. The *text_string* begins with the second character following group. The first character after group must be a blank. Quotation marks are neither required nor treated specially. The *text_string* ends with the last nonblank character before the end of the command line. If no window title is specified, "Please Select Next Step" will be used.

Normally, a group command is the first command in a presentation script, or the second command if a title command is used. However, any number of "action" commands, e.g., a run command, can precede the group command. Such commands form a preamble, which is executed exactly once, before the menu appears. Once the menu has appeared, the preamble cannot be executed again, because it is impossible to step out of a menu. We recommend that such preambles be kept to a bare minimum.

The use of a single slide or a short introductory animation is a nice way to start a presentation. If you need to provide instructions for viewing a presentation, the best approach is to label the first item in the group something like "How to view this presentation (click here)."

item

The item command identifies the start of a sequence of actions. An item command can be used only within the context of a group…endgroup or subgroup…endsubgroup.

The syntax for the item command is as follows:

item *text_string*

The *text_string* appears in the presentation's navigation tree. It can contain blank characters and can be of arbitrary length. The *text_string* begins with the second character following the item command name. The first character after item must be a blank. Quotation marks are neither required nor treated specially. The text_string ends with the last nonblank character before the end of the item command line.

Here are two examples of the item statement:

```
item Introduction
item 120 arrivals per hour
```

play

The play command is used to play .WAV files. The play command can be used to add sound effects, music, or narration to an animation. Note that there is also a play command for use in trace files. The syntax of the play command is as follows:

play [asynchronous] *filename*

The specified *filename* must be a .WAV file. File names which contain embedded blanks must be enclosed in quotes. The asynchronous keyword indicates that the playing of *filename* is to be overlapped with ongoing execution of an animation. If the asynchronous option is not used, execution of the presntation is suspended until *filename* has been played in its entirety.

The following example plays a file named tada.wav:

```
play asynch tada
```

Ongoing, asynchronous .WAV file playing can be terminated as follows:

```
play off
```

run

The run command is used for running all or part of an animation. The syntax of the run command is as follows:

```
run             [ special_effect ]  { layoutfilename [ tracefilename ] } | *
                [ view viewname ] [ speed speed ]
                [ [ startingtime ] endingtime ]
```

The following special effects can be used to control how the animation is brought up ont the screen: checkerboard, dissolve, sunrise, venetian, or vvenetian. See the description of the slide command, below, for descriptions of how these special effects work.

The extension .LAY is assumed for the *layoutfilename*, and .ATF or .PTF is assumed for the optional *tracefilename*. (The Proof Demo Viewer uses .PTF files which are generated from .ATF files by the Proof Demo Maker.) If the trace file you wish to use has the same base name as the *layoutfilename*, it is unnecessary to specify the *tracefilename*. Filenames which include embedded blanks must be enclosed within quotes.

The optional *viewname* determines the named view to be displayed instead of the Home view. If the name of the view contains blank spaces, it must be enclosed in quotes.

The optional *speed* determines the animation speed to be used instead of the speed stored in the layout file.

The optional *endingtime* specifies that this animation is finished when the animation reaches that time. The default is the end of the trace file.

The optional *startingtime* specifies that this animation should begin at the specified animated time instead of at Time 0 (or instead of at the current time, in the case of run *; see below).

There a *startingtime is specified*, an *endingtime* must also be specified; if only one number is specified, it will be interpreted as an *endingtime*.

Note that the use of a *startingtime* implies that Proof must fast forward through a trace file until the starting time is reached. If the starting time is large, time will be wasted performing the fast forward, and space will be wasted by having to store a large trace file. Under such circumstances, it is preferable to generate an abridged trace file which spans exactly the desired time range. “Write Abridged Trace” is an option under “Create Special Files” in Proof’s File menu.

*Using * to specify the animation*

The optional "*" replaces both the *layoutfilename* and the *tracefilename* and stands for "the same layout and trace files as specified in the preceding run command in the presentation script file." If the new *startingtime* is greater than or equal to the *endingtime* of the preceding RUN command, or is omitted, then Proof Animation doesn't need to re-open the files or rewind to time zero before jumping to the new *startingtime*. If the new *startingtime* is omitted, then "the current time" is assumed.

If "*" is used and the *viewname* and/or *speed* is omitted, then the view and/or speed, respectively, are derived from the preceding run command instead of from the layout file.

The following are examples of the run command:

```
run  myfile
run  dissolve  mylayout  mytrace
run  myfile 100 200
run  checkerboard  myfile view "Loading Dock" speed 10  200 300
```

slide

The slide command is used to display graphical (.BMP or .PCX files) information or textual information (.RTF files) developed using another program. The syntax of the slide command is as follows:

slide [checkerboard | dissolve | sunrise | venetian | vvenetian] *filename*

File names which contain embedded blanks must be enclosed in quotes.

The checkerboard effect brings up a slide by expanding tiny squares of the slide image until all the squares converge into a complete image.

The dissolve effect brings up a slide one pixel at a time in randomly selected pixel order.

The sunrise effect brings up a slide by gradually transitioning from a black screen to a full-color screen. If the screen is not already black to begin with, it is quickly faded to black before execution of the sunrise effect.

The venetian effect brings up a slide by expanding horizontal rectangles of the slide image until all rectangles converge. The effect is evocative of venetian blinds.

The vvenetian effect is a vertical version of the venetian effect.

Note that if the color palette of a .BMP or .PCX file differs from the color palette currently in effect at the time the slide command is executed, sunrise is forced. This is done to avoid jarring instantaneous palette changes.

How soon a slide disappears depends on the presentation script file commands that follow the slide command. A slide command is almost always be followed by a wait command; otherwise, the slide may be instantaneously superseded as the result of an action command which follows. For example, if two consecutive slide commands are issued with no intervening wait, the first will be seen as a momentary flash.

The assumed default extension for *filename* is .PCX.

If the image of a .PCX or .BMP file is larger than the space available in the Proof window, only the upper left portion of the image is shown. If the image in a .BMP or .PCX file is smaller than the space available, the image is surrounded with a black band. .BMP files can be stretched (or shrunk) to fit the available space by using the stretch keyword, e.g.,

```
slide stretch dissolve myfile.bmp
```

Proof uses built-in Windows support for displaying .RTF files. The built-in .RTF support does not support all available .RTF formatting options. Basic options, such as font selection, tab stops, and indentation are supported. Standard font colors are supported. .RTF slides are displayed using a grayish blue background.

.RTF support is intended to provide a quick way of incorporating nice looking, but simple textual slides into a presentation. .RTF slides can be generated using Microsoft Word or WordPad.

subgroup

The subgroup command marks the start of a lower-level collection of items and/or subgroups in a hierarchical presentation. The collection must be terminated by an endsubgroup command. Items and/or subgroups within a subgroup are hidden in a navigation tree until the subgroup is expanded by clicking on it. The syntax of the subgroup command is as follows:

subgroup *text_string*

The *text_string* is used as the subgroup's label in the presentation's navigation window. The text_string begins with the second character following subgroup. The first character after subgroup must be a blank. Quotation marks are neither required nor treated specially. The text_string ends with the last nonblank character before the end of the command line.

The use of subgroups is highly recommended for lengthy, complex presentations. The ability to quickly expand or shrink subgroups in a presentation's navigation tree makes it possible for viewers to rapidly home in on topics of interest.

syscall

The syscall command is used for invoking other programs from within a presentation script. Before invoking a specified program, the Proof window is minimized and placed on the taskbar. After the specified program completes its execution, you must reactivate Proof by clicking on icon in the taskbar or by means of Alt-TAB task switching. You must then manually resume the presentation, e.g., select the next step from the navigation tree. The syntax of the syscall command is as follows:

syscall *program name* [*program arguments*]

For example,

```
syscall  d:\slx\se  d:\slx\myprog
```

runs D:\slx\se, passing it “d:\slx\myprog” as an argument.

Program names which contain embedded blanks must be enclosed in quotes.

title

The title command is used to customize the title bar of the Proof window.

The syntax of the title command is as follows:

title *text_string*

The specified *text_string* replaces the standard Proof-provided title.

wait

The wait command is used to delay or pause the presentation. The wait command is most frequently used after the slide and run commands. Three forms of the wait command are available:

wait

wait *time_delay*

wait [*time_delay*] *text_string*

If first form is used, a presentation is paused until the viewer intervenes. The phrase “PRESENTATION PAUSED” appears in the title bar of Proof’s main window.

If the second form is used, *time_delay* specifies the number of seconds to wait before continuing the presentation. This form of the wait command should be used sparingly. You should not attempt to construct a “forced march” presentation, in which you control the holding time for slides by using timed wait commands. It is very difficult to accurately anticipate viewing times which are appropriate for everyone. One common use of timed waits is to freeze the screen for a second or two at the conclusion of an animation, before clearing the screen via dissolve or other special effect.

If the third form is used, *text*_string is shown in a message box. If a *time_delay* is specified, the message box will disappear after the specified time delay has elapsed, and the presntation will continue; otherwise, a user OK is required to proceed.

The following are examples of the wait command:

```
wait
wait 2.5
wait 3 The overload case will be shown next.
```

wipe

The wipe command is used to clear the screen to a specified color by using a special effect. The specified *color* can be F1 through F32, L1 through L32, or Backdrop. The syntax of the wipe command is as follows:

```
    wipe   color   checkerboard
                   curtain up
                   curtain down
                   curtain updown
                   dissolve
                   random
                   fade
                   venetian
                   vvenetian
```

The checkerboard effect transitions to a solid color by expanding tiny squares of color until all the squares converge into a solid color.

The curtain up, curtain down, and curtain updown effects transition to a solid color using a quickly moving wave of color which moves down the screen, up the screen, or in both directions simultaneously.

The dissolve effect obliterates the screen one pixel at a time in randomly selected pixel order.

The random effect randomly chooses one of the other effects. The *color* specification is ignored, and a randomly selected color is used.

The fade effect smoothly reduces the colors of the screen to black over a short interval. The *color* specification takes effect only when the screen is forced back into full-color mode.

The venetian effect transitions to a solid color by expanding horizontal rectangles until all rectangles converge. The effect is evocative of venetian blinds.

The vvenetian effect is a vertical version of the venetian effect.

* Comment Lines

Comment lines can be used to provided non-executable annotations in script files. Such annotations are unseen by viewers of the presentation, but provide helpful reminders to the developer of the presentation. Any line that begins with an asterisk (*) is a comment line

* *comment*