WorkstationJ Case Randomization Keywords
(Updated September 30, 2010)

Keywords to randomize the presentation of cases can be added to script files. The "RND$" tag is used to indicate that the line is a randomization specification is. Keywords should be separated with semicolons. The available keywords are:

1. `fixed` : an optional keyword used to indicate the number of cases at the beginning of the presentation that are to be presented in the order specified in the script file (i.e., not randomized). If not specified, a value of "`fixed=0`" is the default.

   Example: "`fixed=3`" would be used to indicate that there is a title case, demo case, and practice case at the beginning of the script that should be presented in the order they appear in the script

2. `blocks` : an optional keyword used to indicate the number of blocks into which the non-fixed cases should be divided. Cases are randomized within blocks. If not specified, a value of "`blocks =1`" is the default.

   Example: "`blocks=2`" would be used to indicate that the remaining (total-fixed) cases should be divided into two blocks.

3. `seed` : an optional keyword used to specify the seed to use for the pseudorandom number generator (must be greater than zero). If a `seed` is specified, the `prompt` keyword is ignored. It would be convenient to specify the seed in the script if you want the same seed to be used for all readers. Valid values for `seed` are 1 to 9,223,372,036,854,775,807 (inclusive). Specifying a value of 0 causes the seed to be generated for the system clock. If not specified, a value of "`seed =0`" is the default.

4. `prompt` : an optional keyword used to indicate whether a dialog should appear prompting for a seed to use. If not specified, a value of "`prompt =0`" is the default unless a value of `seed` was provided (i.e., `seed` takes precedence over `prompt`).

   Valid entries:
   "`prompt=0`" : do not prompt—seed is automatically generated
   "`prompt=1`" : prompt for seed

5. `runlimit`: an optional keyword used to indicate how many trials of a particular type can appear consecutively. If the number of consecutive cases of a particular type exceeds `runlimit`, the generated sequence is considered discarded, a new seed is computed, and another sequence is generated. This process of discarding the sequence, computing a new seed, and generating a new sequence is repeated until the `runlimit` specification is satisfied. If `runlimit` is specified, then the `TAG` keyword (see *WorkstationJ Script File Preparation.pdf*) must be used to specify the `casetype` for each case. If the TAG keyword is not used, all the cases are assumed to be of the same type by default.

   Example: "`runlimit=3`" would be used to indicate that no more than three files of the same type could appear consecutively.

If an invalid value is specified for a keyword, the default for that keyword is assumed. For example, specifying a value of `seed=-1` would result in a warning dialog and the program automatically generating a seed.

If the values specified for fixed and blocks would result in blocks of unequal size, then a warning dialog appears and no randomization takes place. For example, if a script file contains specifications for 32 cases and the user specifies values of `fixed=3` and `blocks=2`, a warning appears and no randomization occurs because the remaining 29 cases cannot be evenly divided into two blocks.

Examples:

1. `RND$ fixed=3; blocks=2; prompt=1`

The first three trials are presented in the order they appear in the script file. The remaining trials are divided into two blocks with each block randomized. The program will prompt the user for the seed to use.

2. `RND$ fixed=1; blocks=2; seed=2007`

The first trial in presented in the order it appears in the script file. The remaining trials are divided into two blocks with each block randomized. The program will use the value of "2007" as the seed for the generator. Because the value of the seed was specified, no prompting occurs.

3. `RND$ fixed=3; blocks=1; prompt=0; seed=20070905`

The first three trials are presented in the order they appear in the script file. The remaining trials are treated as a single block and randomized. The program will not prompt the user for the seed to use. The program will use the value of "20070905" as the seed for the generator.

4. `RND$ fixed=3; blocks=2`

The first three trials are presented in the order they appear in the script file. The remaining trials are divided into two blocks with each block randomized. The program will automatically generate the seed based on the current value of the system clock. The seed is written to the log and data files for future reference.

5. `RND$ fixed=3; blocks=1; prompt=1; seed=20070905`

The first three trials are presented in the order they appear in the script file. The remaining trials are treated as a single block and randomized. The program will use the value of "20070905" as the seed for the generator. The "prompt=1" specification is ignored because a seed was specified.

6. `RND$ fixed=0; blocks=2; prompt=1; seed=0`

None of the trials are presented in the order they appear in the script file. All of the trials are divided into two blocks with each block randomized. The program will automatically generate the seed based on the current value of the system clock. The "prompt=1" specification is ignored because a seed was specified.

7. `RND$ blocks=2`

None of the trials are presented in the order they appear in the script file. All of the trials are divided into two blocks with each block randomized. The program will automatically generate the seed based on the current value of the system clock. The seed is written to the log and data files for future reference.

8. `RND$ fixed=3`

The first three trials are presented in the order they appear in the script file. The remaining trials are treated as a single block and randomized. The program will automatically generate the seed based on the current value of the system clock. The seed is written to the log and data files for future reference.

9. `RND$ prompt=1`

None of the trials are presented in the order they appear in the script file. All of the trials are treated as a single block and randomized. The program will prompt the user for the seed to use.

10. `RND$`

None of the trials are presented in the order they appear in the script file. All of the trials are treated as a single block and randomized. The program will automatically generate the seed based on the current value of the system clock. The seed is written to the log and data files for future reference.

11. `RND$ fixed=3; blocks=0; seed=20070905`

The first three trials are presented in the order they appear in the script file. Because an invalid value was specified for `blocks`, the remaining trials are treated as a single block and randomized. The program will use the value of “20070905” as the seed for the generator. A warning dialog will appear to alert the user that the randomization specification was invalid.

12. `RND$ fixed=3; blocks=2; seed=2007; runlimit=3`

The first three trials are presented in the order they appear in the script file. The remaining trials are divided into two blocks with each block randomized. The program will use the value of "2007" as the seed for the generator. The "runlimit" specifier indicates that no more than three cases of the same type can appear consecutively in the sequence before the sequence is considered faulty.