Getting More of the Good Stuff

When One BBj Session Just Isn't Enough!

By Jon Bradley

The `BBjAPI().newBBjSession()` provides a BBj® programmer with an explicit way to create a new BBj interpreter session. Previously, the programmer used a command such as `a = SCALL("BBj newProgram.bbj &")` to start a new session. This methodology contains implicit (do-what-I-*mean*) behavior. A system that attempts to 'do-what-I-*mean*' makes assumptions about the general-use case, which may differ significantly from a specific developer's application. In many cases, programmers prefer a more explicit (do-what-I-*say*) option. The results of a system without implicit behavior are easier to predict and configure. The BBj programmer can use the `BBjAPI().newBBjSession()`, along with the BBjCommandLineObject, to gain explicit control of configuring the new BBj session.

When using SCALL, a command line terminated with an "&" will not wait for the new process to exit. Developers commonly refer to this type of call as an asynchronous or non-blocking call. Omission of the "&" causes a synchronous or blocking SCALL to wait for the termination of the new process. Similarly, an invocation of `BBjAPI().newBBjSession()` does not wait for the termination of the new BBj session. Invoking `BBjAPI().newSynchBBjSession()` provides the appropriate syntax for a blocking instantiation of a new BBj session.

The BBjCommandLineObject provides an object-oriented way to specify how the developer configures a new BBj session. Configuring an SCALL required assembling a string of command line flags, configuration directives, and program arguments. The BBjCommandLineObject provides methods for setting and retrieving these command line properties. The BBjConfig object provides several methods to create a BBjCommandLineObject. Some examples include:

`BBjConfig.getCommandLineObject()`– returns a blank BBjCommandLineObject

`BBjConfig.getCommandLineObject(String p_args)`– returns a BBjCommandLineObject initialized to represent the command line arguments of p_args.

`BBjConfig.getCurrentCommandLineObject()`– returns a BBjCommandLineObject initialized to represent a new session much like the current session. The `getCurrentCommandLineObject()` method does not initialize the `programName` and `programArguments` properties, as it is unlikely the developer would want to use the sameprogram arguments in the new session.

The usage of the `BBjAPI().newBBjSession` and the BBjCommandLineObject typically look like this:

```
REM - get a command line object to configure
cmd! = BBjAPI().getConfig().getCurrentCommandLineObject()
REM - configure the cmd! object
cmd!.setInterpreterUser("sampleuser")
cmd!.setProgramName("newProgram.bbj")
cmd!.setProgramArgs("arg1 arg2 arg3")
REM - create new session via newBBjSession
BBjAPI().newBBjSession(cmd!)
```

The principle advantages of the `BBjAPI().newBBjSession()` method over the SCALL verb is `newBBjSession` does not introduce implicit behavior and the ability to configure the new session without performing the string manipulations required by SCALL. Additionally, if the developer must start multiple similar new sessions, it is easy to reuse the BBjCommandLine object as an argument to multiple calls to `BBjAPI().newBBjSession()`.