Easily Install Your Apps With the BCI

By Brian Hipple
Quality Assurance Supervisor

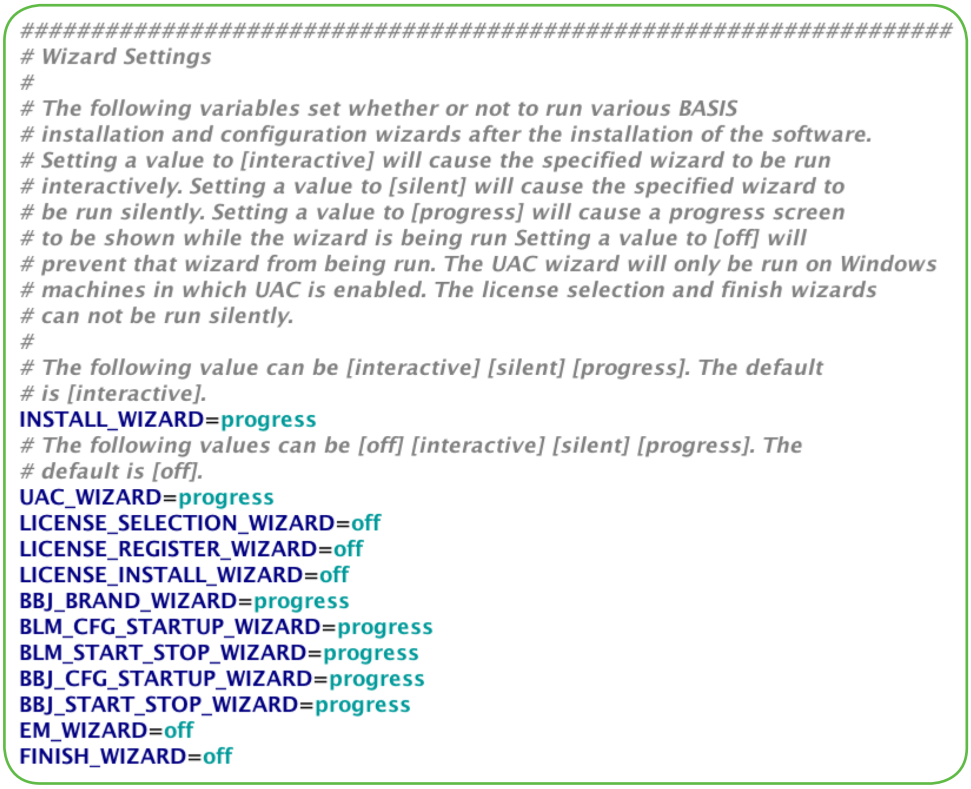
The BASIS Custom Installer (BCI) continues its evolution towards a fully-featured installation program, not only for BASIS products, but for customers' applications as well. It is very appropriate that the word 'custom' is included in the name of this installer. BASIS wrote the Installer with a great deal of flexibility for the developer and it is completely configurable in all aspects of installing any BBx®-based application. More explicitly, the developer can determine which commands to invoke and under what conditions, what files to install, and which BASIS installation wizards to run and in what manner. All of this functionality is specified in either an installation response file or an instructional XML file. These files include new options that BASIS incorporated into the latest version of BBj®.

Recently, the BASIS marketing team came to the engineers with an idea for a campaign to promote our ERP product written in BBj – AddonSoftware® by Barista®. They wanted the user to be able to download, install, configure, and run AddonSoftware without *any* interaction. The intended audience for the promotion would likely be unfamiliar with BASIS or our products, so it was imperative that this was a simple one-click process. They did not want any prompts to appear requiring users to make otherwise typical selections – the Java version, install directory, license registration, license manager, or how to configure and start the BLM or BBj Services. The team wanted it all to work and be configured automatically on all BBj-supported operating systems. Was the BCI up to the task?

Most definitely the BCI could handle the challenge, however, along the way we found opportunities for improvement that not only met the requirements of our marketing team, but also would offer great benefits to the BASIS community at large.

Enhancement #1 - Progress Mode

The BCI gives the choice to configure the installation wizards to run either interactively or silently via the response file. Since we definitely did not want the installation to be interactive, we chose the silent option. What we found out with this option was that depending on the system, it could take several minutes for BBj and AddonSoftware to be installed, without giving users any sense of progression. Here was the first opportunity for enhancement. We added a new progress mode to the response file to display a progress screen in both GUI and CUI. This new option provides the end user an indication of the progress of the installation and improves the user experience. An example of the new progress mode is shown in an excerpt of a response file in **Figure 1**.



```
######################################################################
# Wizard Settings
#
# The following variables set whether or not to run various BASIS
# installation and configuration wizards after the installation of the software.
# Setting a value to [interactive] will cause the specified wizard to be run
# interactively. Setting a value to [silent] will cause the specified wizard to
# be run silently. Setting a value to [progress] will cause a progress screen
# to be shown while the wizard is being run Setting a value to [off] will
# prevent that wizard from being run. The UAC wizard will only be run on Windows
# machines in which UAC is enabled. The license selection and finish wizards
# can not be run silently.
#
# The following value can be [interactive] [silent] [progress]. The default
# is [interactive].
INSTALL_WIZARD=progress
# The following values can be [off] [interactive] [silent] [progress]. The
# default is [off].
UAC_WIZARD=progress
LICENSE_SELECTION_WIZARD=off
LICENSE_REGISTER_WIZARD=off
LICENSE_INSTALL_WIZARD=off
BBJ_BRAND_WIZARD=progress
BLM_CFG_STARTUP_WIZARD=progress
BLM_START_STOP_WIZARD=progress
BBJ_CFG_STARTUP_WIZARD=progress
BBJ_START_STOP_WIZARD=progress
EM_WIZARD=off
FINISH_WIZARD=off
```

Figure 1. Example of the response file that sets the progress mode for the wizards

Enhancement #2 - License Installation

Another ability we needed was to have a temporary license installed by default. To accomplish this, the installer now can utilize a BASIS license file in the installable jar file when it's named **`basisdemo.lic`**. When found, this license installs automatically and BBj and the BLM configure appropriately to utilize it. You can easily add a license file to an existing installation jar via the syntax and example shown in **Figure 2**.

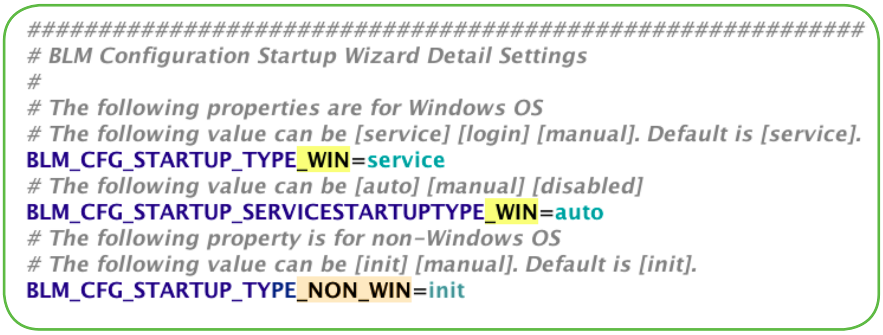
```
<JAVA_HOME>/bin/jar -uf <BASIS Installable.jar> basisdemo.lic

/usr/local/java/bin/jar -uf
BBjBaristaAddonIDE1201_08-13-2012_1245.jar basisdemo.lic
```

Figure 2. Generic syntax (top) and example (bottom) command for adding a demo license to an installation jar

Enhancement #3 - Platform Specificity

We needed more granular specifications for the Java directory, installation directory, and configuration options. Response file values were not OS specific and only gave the developer one chance to set them, regardless of the platform. Therefore, we added _WIN and _NON_WIN suffixes to the desired property keys for Windows- and non-Windows-specific operating systems. Now, we can include the options for different platforms conveniently in the same response file. **Figure 3** shows some of the OS-specific property keys.

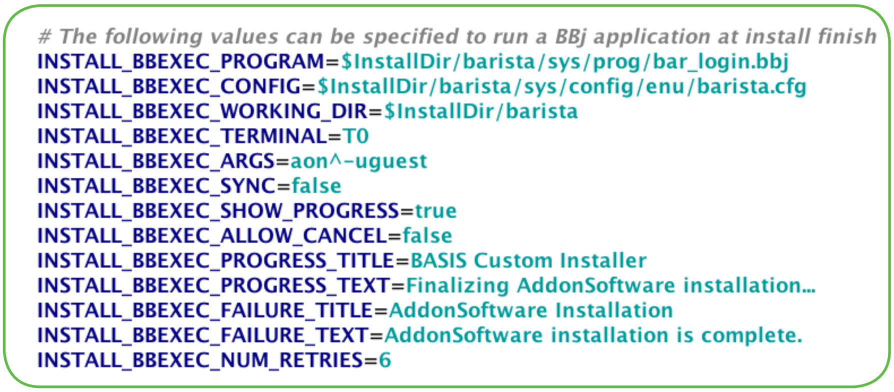


```
###########################################################################
# BLM Configuration Startup Wizard Detail Settings
#
# The following properties are for Windows OS
# The following value can be [service] [login] [manual]. Default is [service].
BLM_CFG_STARTUP_TYPE_WIN=service
# The following value can be [auto] [manual] [disabled]
BLM_CFG_STARTUP_SERVICESTARTUPTYPE_WIN=auto
# The following property is for non-Windows OS
# The following value can be [init] [manual]. Default is [init].
BLM_CFG_STARTUP_TYPE_NON_WIN=init
```

Figure 3. OS-specific property keys for setting specific values

Enhancement #4 - Auto Execute

Lastly, we needed to run the AddonSoftware application after all the components had been installed. This was previously possible by specifying a BBExec node in the instructional XML file: **`custominstall.xml`**. However, we thought this might be an option that our customers would frequently want to take advantage of so we added this ability to the more accessible response file as shown in **Figure 4**.



```
# The following values can be specified to run a BBj application at install finish
INSTALL_BBEXEC_PROGRAM=$InstallDir/barista/sys/prog/bar_login.bbj
INSTALL_BBEXEC_CONFIG=$InstallDir/barista/sys/config/enu/barista.cfg
INSTALL_BBEXEC_WORKING_DIR=$InstallDir/barista
INSTALL_BBEXEC_TERMINAL=T0
INSTALL_BBEXEC_ARGS=aon^-uguest
INSTALL_BBEXEC_SYNC=false
INSTALL_BBEXEC_SHOW_PROGRESS=true
INSTALL_BBEXEC_ALLOW_CANCEL=false
INSTALL_BBEXEC_PROGRESS_TITLE=BASIS Custom Installer
INSTALL_BBEXEC_PROGRESS_TEXT=Finalizing AddonSoftware installation…
INSTALL_BBEXEC_FAILURE_TITLE=AddonSoftware Installation
INSTALL_BBEXEC_FAILURE_TEXT=AddonSoftware installation is complete.
INSTALL_BBEXEC_NUM_RETRIES=6
```

Figure 4. Defining a BBj program to run after installation completes

This ended up being a great win-win experience as "eating our own dog food" gave us the chance to "taste" some of the same BCI hurdles as our customers and implement solutions that benefited us in our immediate need, and BASIS customers down the road.

But Wait, There's More...

While our own marketing campaign was the catalyst for this string of improvements, shortly afterwards a customer came to us with some additional BCI requirements. They needed to 1) install their own product outside of the BASIS directory and 2) configure the uninstall wizard not to uninstall their product.

Seeing these customer requirements as viable enhancements, we added the ability to identify the installation drive via a new $InstallDrive variable so that on Windows, for example, the developer could provide a full path in order to create shortcuts to their product offerings. In order to be exempt from the uninstall process, we added the **`uninstall="never"`** attribute on both the suite and feature nodes in the custominstall.xml file.

Summary

"So what," you ask? Two things are worth noting. Firstly, the BCI is a very powerful and useful tool and it is included with BBj obviating the need to purchase a third party tool. If you are not already taking advantage of it, it might be time to exploit it. Secondly, and perhaps more importantly, if you are using the BCI and it currently doesn't have the features you want, don't be shy. Let us know what you need and if it makes sense, we will add it to this powerful building block! 🟩

Read *Custom Apps Install – Easy as 1- 2- 3!* at
links.basis.com/11custominstall