

BRE Management in BDT

Version 24.02

To accompany the 24.02 release of BBj and BDT

Table of Contents

[Table of Contents](#)

[BRE Overview](#)

[Getting Started](#)

[Prerequisites](#)

[Viewing Properties of a BRE](#)

[Hover Text](#)

[Expanding the BRE Node](#)

[Classpath](#)

[Config file](#)

[PREFIX Resources](#)

[Configuring a BRE](#)

[Arguments](#)

[Look and Feel](#)

[Connections](#)

[Indexing](#)

[Managing the Collection of BREs](#)

[Selecting the BRE for a BBj Project](#)

[Creating a BRE](#)

[Creating a New BRE](#)

[Copying a BRE](#)

[Deleting a BRE](#)

[Seeing What BBj Projects are Using](#)

[The BBj Projects Using Every BRE](#)

[The BBj Projects Using Each Configuration File](#)

[The BBj Projects Using a Specific BRE](#)

BRE Overview

The purpose of this document is to provide reference details explaining the use and impact of the BBJ Runtime Environment (BRE) objects used in the Business BASIC Development Toolkit (BDT) in Eclipse.

BDT is a plug-in for the Eclipse IDE, which supports the development and maintenance of your BBJ programs. This maintenance includes creating, updating, running, and debugging those programs.

In order to know what version of BBJ is important, the BDT plug-in requires you to navigate to and select a specific on-disk BBJ version; this is known in BDT as the BBJ Installation. Although a BBJ Installation is critical for BDT to function, it is not enough information to run a BBJ program. Why? Because running a BBJ program requires BDT to communicate with a running instance of BBJServices. A BRE holds the details of how BDT finds and communicates with that running instance and what arguments it should pass along when it tells BBJServices to run a program. Each BBJ project in the user's Eclipse workspace must be configured to use a specific BRE, and that BRE is then used each time the user asks to run a BBJ program that is contained in that BBJ project.

For an explanation of Eclipse's hierarchical model for perspectives, projects, and views, see this article: [Perspectives and Projects and Views, Oh My! - A Survival Guide](#).

NOTE: *The BDT Explorer and other details covered in this document are current as of the 24.02 releases of BBJ and BDT. Please remember that you must update both BBJ and BDT to have your BDT provide all of the functionality documented here.*

Getting Started

To help you understand how to create and manage your BREs, let's look at what you can configure in a BRE. But in order to do that, we will first have to look at where you can find your BREs in the BDT Explorer. After we do those things, we can look at how you select and use one of those BREs in your BBJ Project.

Prerequisites

To get started, you must already have BBJ and Eclipse installed on your computer and have a version of the BDT Plug-ins installed that works with your version of BBJ. You must also have already searched for and selected your BBJ installation's on-disk location in the BDT (resulting in a valid BBJ Installation node appearing at the very top of the BDT Explorer, as shown in **Figure 1** below).

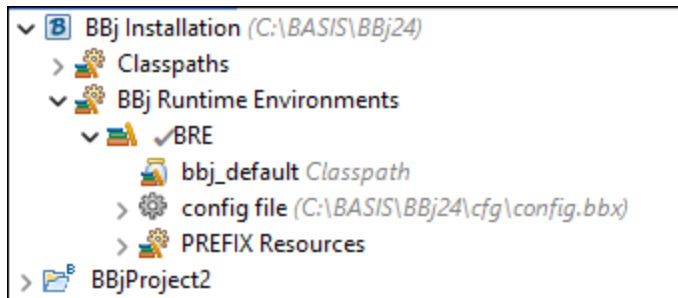


Figure 1. A Valid BBJ Installation as Seen in the BDT Explorer

In **Figure 1**, the BBJ Installation node in the BDT Explorer is expanded so that you can see where the BBJ Runtime Environments are located. With the “BBj Runtime Environments” node expanded, you see a list of all of the BREs defined for this BBJ installation. In **Figure 1**, there is exactly one BRE defined. When you first selected your BBJ installation, BDT created that one BRE for you:

- It is named “BRE”
- It is marked as the default, as indicated by the gray check mark preceding its name
- It is configured with all default properties (which we will examine next)

What is the significance of the “default” BRE? The BRE that is marked as the default will be assigned, by default, for any new BBJ project you create. You can select any valid BRE during the creation process, and you can always edit your BBJ project later and select any valid BRE in the installation (more on this later). But, the default BRE holds one other special position: when and if you ever set a different BRE to be the default, all of your BBJ projects that are set to use the “default BRE” (as opposed to using a specific BRE by name) will automatically begin using the new default. More on that later.

Viewing Properties of a BRE

Let’s take a look at the properties of a BRE that you can view without attempting to edit it. But one important note first: since many of the BRE’s properties are external (they refer to external files such as BBJ’s config.bbx or bbj.properties files), you may need to refresh the BRE occasionally to ensure that all of the information we view below is current. To do that, either:

- Right-click the individual BRE node and select **Refresh (F5)** (which will refresh that BRE).
- Right-click the BBJ Runtime Environments node and select **Refresh (F5)** (which will refresh all BREs).
- Right-click the BBJ Installation node and select **Refresh (F5)** (which will refresh all of the BREs and the entire BBJ installation).

Hover Text

By hovering the mouse over a BRE node in the BDT Explorer, hover text will show a summary of the critical properties as seen in **Figure 2** below:

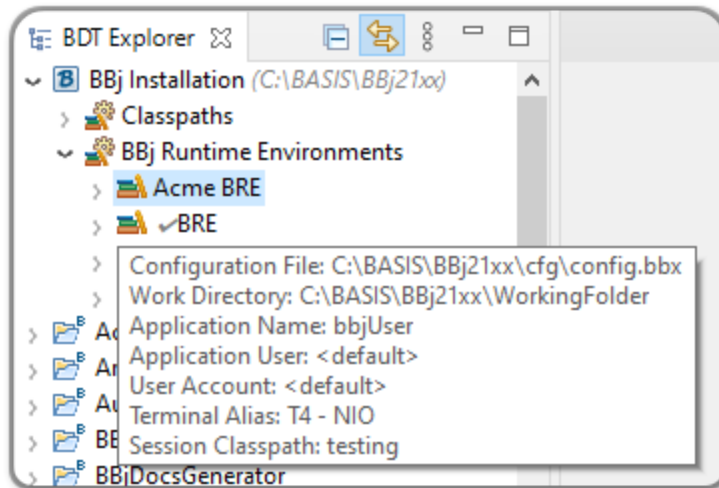


Figure 2. Hover Text Showing the Main Properties of “Acme BRE”

This hover information can save you time when you simply need to check the value for any BRE’s main properties since you do not need to open it for editing.

Expanding the BRE Node

By expanding the BRE node in the BDT Explorer by one level, you can see additional details about the classpath and configuration file it is using, and the resources that BDT found by traversing the PREFIX (**Figure 3**):

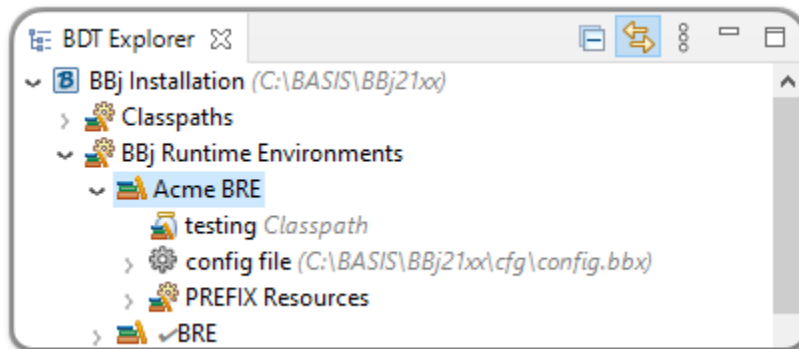


Figure 3. Expanding “Acme BRE” Shows the Top-Level Properties

You can drill down further to see additional details about these items.

Classpath

As shown in **Figure 3**, this node shows the name of the Session Specific ClassPath (SSCP) the BRE is using. In this case, it is named “testing.” Since a classpath is optional, this node only appears if the BRE is configured to use a specific SSCP (and not the default).

Right-clicking this node offers the option to “Show Session Classpath” (**Figure 4**):

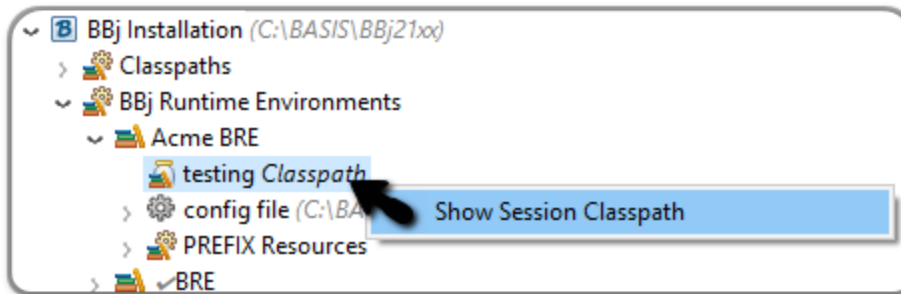


Figure 4. Asking for Details of the “testing” Classpath

Selecting this menu item causes the BDT Explorer to expand the BBj Installation’s “Classpaths” list and select the “testing” classpath for you. After expanding it, you can see the classpath details shown in **Figure 5**:

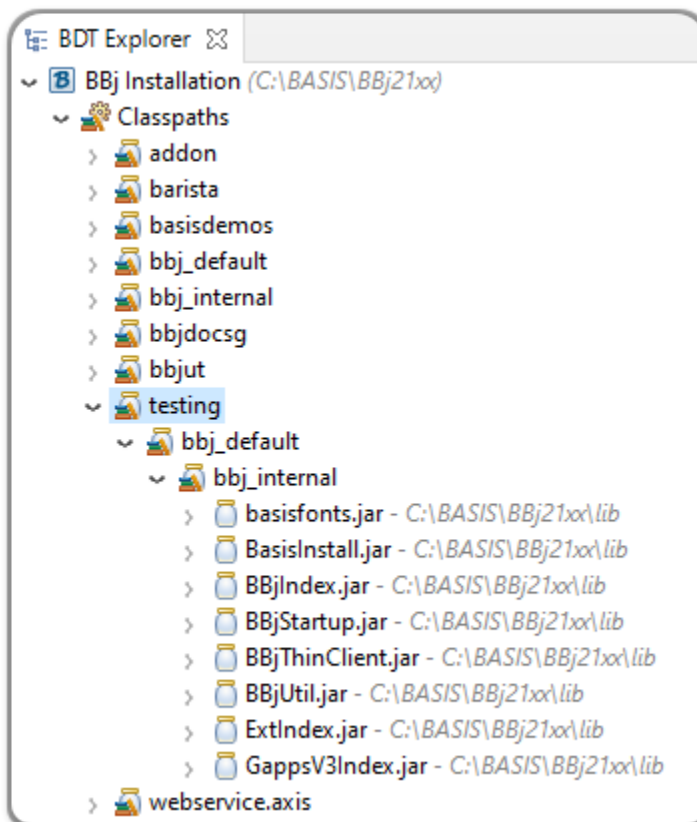


Figure 5. Details of the “testing” Classpath

Config file

As shown in **Figure 3**, the config file node displays the full path to the BBj configuration file the BRE is using in parentheses after the text, “config file.”

Expand this node to see an alphabetical list of the config file entries (**Figure 6**):

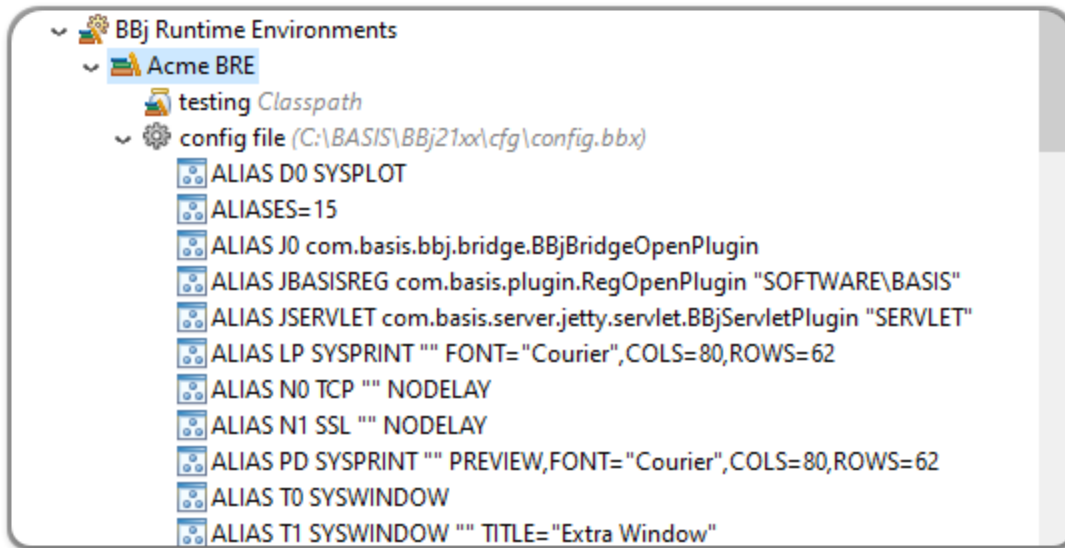


Figure 6. An Alphabetical Listing of the Entries in the “config file”

Viewing these in alphabetical order should make it easy to find a particular entry. However, to see the actual config file contents, right-click the config file node and select “View config.bbx” (**Figure 7**):

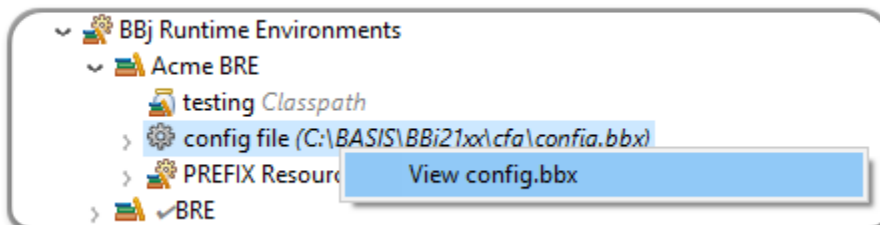


Figure 7. Asking for Details of the “config file”

Selecting this menu item causes the BDT Explorer to open a read-only text editor on the configuration file, as seen in **Figure 8**:



Figure 8. The Unsorted Contents of the Actual Configuration File

This feature permits you to answer more complicated questions such as, “In what order do the entries occur?”

PREFIX Resources

As shown in **Figure 3**, a BRE includes a “PREFIX Resources” node. Expand that node to view the hierarchical tree of files that BDT found when it traversed the PREFIX (based on how you configured the BRE to perform indexing), as seen in **Figure 9**:

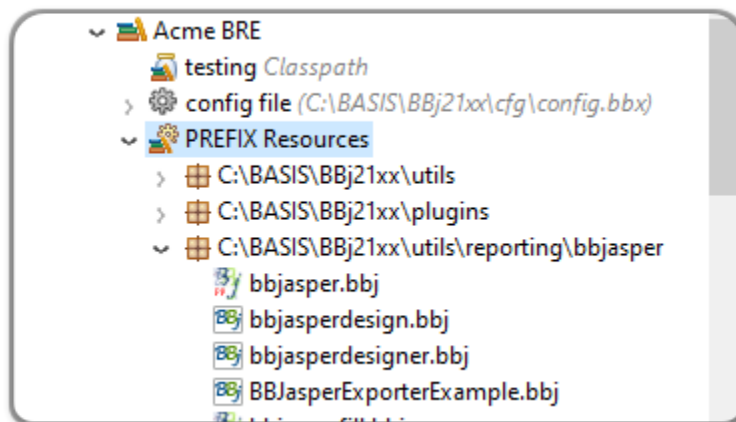


Figure 9. A Hierarchical Listing of the Files Found by Traversing the PREFIX

Files that were found are shown in the folder hierarchy where they were found; empty folders (ones where no relevant files were found) are not included, simply to make the list more useful.

Hover over a specific file to see hover text that should help explain the file icon used. For example, **Figure 10** shows the “Media Type” text from hovering over a SAVEPed file that would require a password to open (thus the unusual file icon):

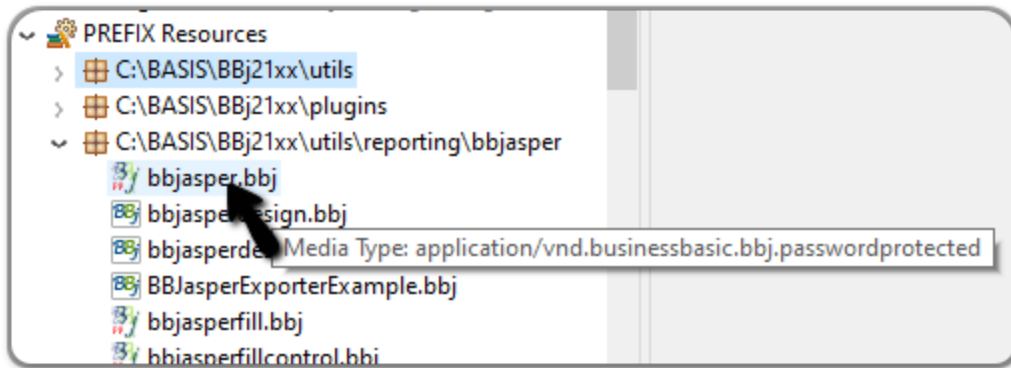


Figure 10. Viewing the Media Type Hover Text for a PREFIX Resource

Note: The PREFIX Resources list does NOT contain the same information as the PREFIX itself because the BRE can be configured to ignore irrelevant file types (more on this in the [Indexing](#) section below).

Configuring a BRE

Now let's look at the properties you can configure for a BRE. To do that, let's tell BDT that we want to edit our one-and-only "BRE." As shown in **Figure 11**, right-click the "BRE" node and select **Edit**:

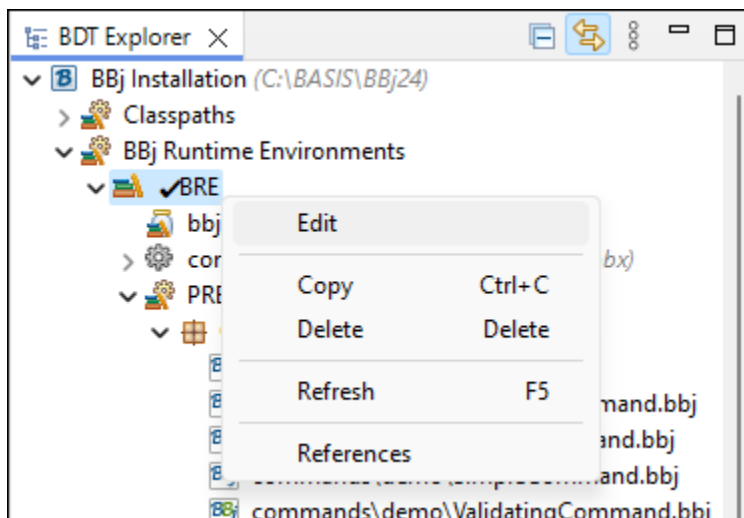


Figure 11. Editing the "BRE" Node in the BDT Explorer

This option opens the **Edit BBj Runtime Environment** window to the **Arguments** tab shown in **Figure 12**:

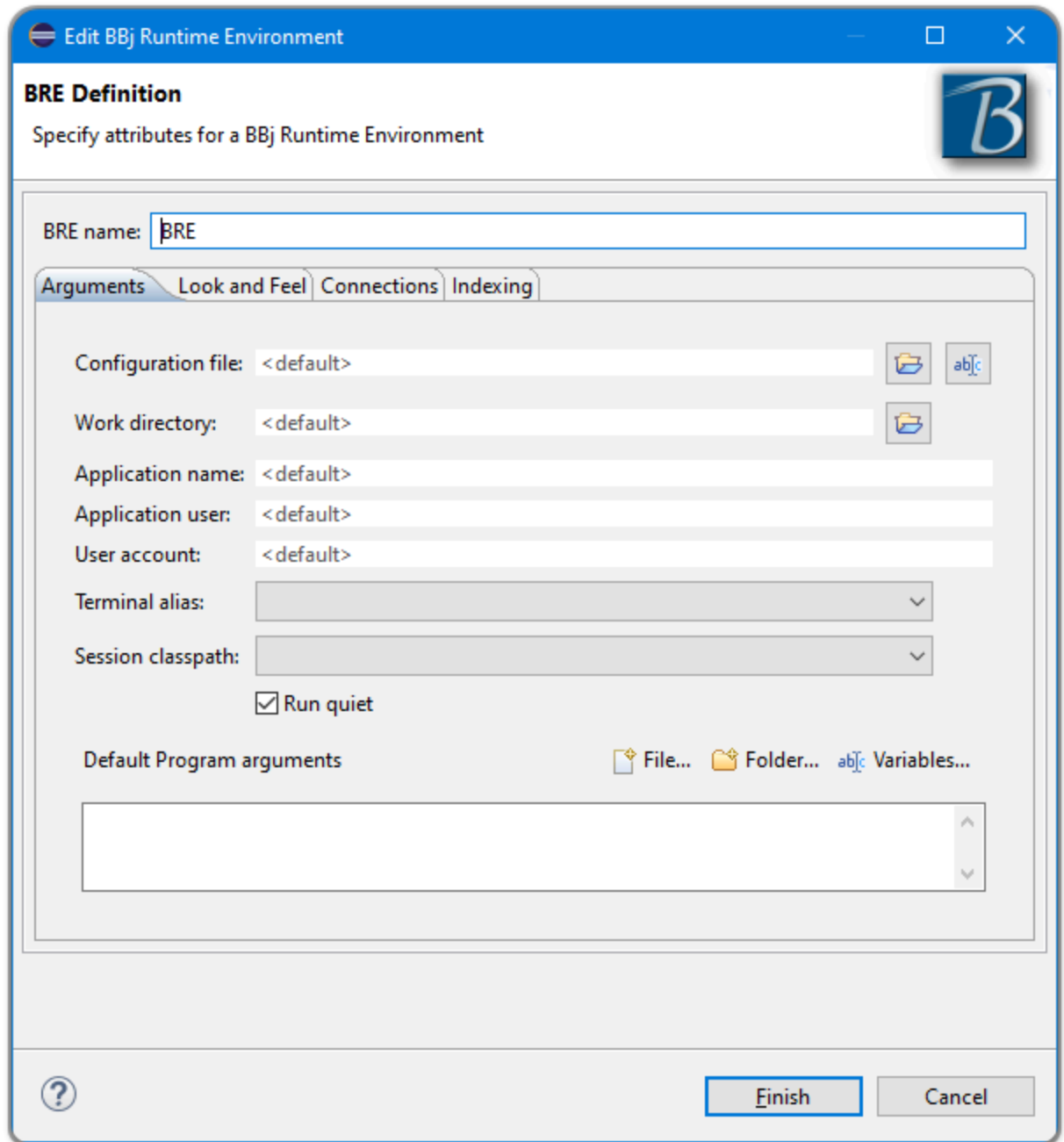


Figure 12. The Arguments Tab

Set the property fields as desired and click **[Finish]** to save your changes. To start, enter a name of your choice in the “BRE name:” property field.

- BRE names must be unique (no two BREs in the workspace can have the same name)
- The BRE name appears in several places in the BDT Explorer:
 - In the list of BREs for the BBJ Installation (as seen in **Figure 1**)
 - Under each BBJ Project that uses it; in **Figure 1**, you can see that the Acme and Arc BBJ Projects are both configured to use the default BRE, named “BRE”
 - When selecting a BRE for a specific BBJ Project (more on this later)

For the other property fields, details are provided below (organized by tab).

Arguments

The following properties are available on the **Arguments** tab to define the properties of a **BRE**.

Note: Unless otherwise specified, any of the following properties may be left empty. If you do so, BDT will show “<default>” there (as it does in **Figure 12**), and BBjServices will use whatever default value is appropriate at run time.

- **Configuration file** - the full path and name of the configuration file (often named “config.bbx”) that BDT should use when running BBj programs. You can edit the **Configuration file** property by manually entering a path and file name or by clicking the



navigation button to the right of the text entry field. This navigation button opens an OS-specific **Open** window that allows you to navigate and select the folder and file to use.

- **Work directory** - the full path to the folder which BDT should use as the working directory when building or running BBj programs. You can edit the **Work directory**



property by manually entering a path or clicking the navigation button to the right of the text entry field. The navigation button opens an OS-specific **Open** window that allows you to navigate and select the folder to use as the **Work directory**.

- **Application name** - this property is incorrectly offered here. Although it was intended to hold the name that BDT should pass to the web server for your program when registering it as a BUI application to run in a browser, that value is actually set now in the corresponding Eclipse Run Configuration (and it thus different for every BBj program you register and run). **You should leave this field empty**, with BDT showing “<default>” there (as it does in **Figure 12**).
- **Application user** - this is the user name that BDT should use when running your program. Enter the name of a user that is valid in BBjServices.
- **User account** - the user account to be used for any Data Server and AppServer access that may be needed to run your programs. Enter the name of a user that is valid for DS and AppServer access.
- **Terminal alias** - the terminal alias that BDT should use when running your program. Select the terminal alias from the offered list, but remember that the aliases offered come from the **Configuration file** selected earlier; if the **Configuration file** changes, any terminal alias specified here may no longer be valid.
- **Session classpath** - the name of the Session Specific ClassPath (SSCP) that BDT should use when running your program. Select the classpath from the offered list, but remember that the list of classpaths offered comes from the `bbj.properties` file in the BBj installation folder. If the `bbj.properties` file or the BBj installation changes, any specific classpath selected here may no longer be valid.

- **Run quiet** - check this box to have BDT run BBj programs quietly. When a BBj program is run quietly, it will not display an opening BBj splash screen as part of its startup, and the **SysConsole** window, if visible, will not display the BBj version information at the top.
- **Default Program Arguments** - Enter any BBj program arguments you would like BDT to set in any new Run Configuration that it creates for you in the future. This field would perhaps be easier to understand if it was instead named "Run Configuration Creation Program Arguments".
 - These arguments are creation defaults. The text you put in this field is only used when you create an Eclipse Run Configuration (a type of Launch Configuration used to run BBj GUI programs). Once a Run Configuration exists, changing the Default Program Arguments field here has no impact on what is sent to BBj at run time.
 - Leaving the Default Program Arguments field empty will cause BDT to create all future Run Configurations without any program arguments (there is no default set of program arguments that are set instead).
 - To have a BBj program receive unique program arguments at run time, edit the Program Arguments field in the corresponding Run Configuration. For a full explanation of how this BRE property interacts with Run Configurations at run time, see [BDT and Runtime Program Arguments](#).

These properties are all passed by BDT to BBj at run time. For more details on these properties, see [Running BBj from the Command Line](#) and the online help.

Look and Feel

The following properties are available on the **Look and Feel** tab to define the properties of a **BRE**, as shown in **Figure 13**:

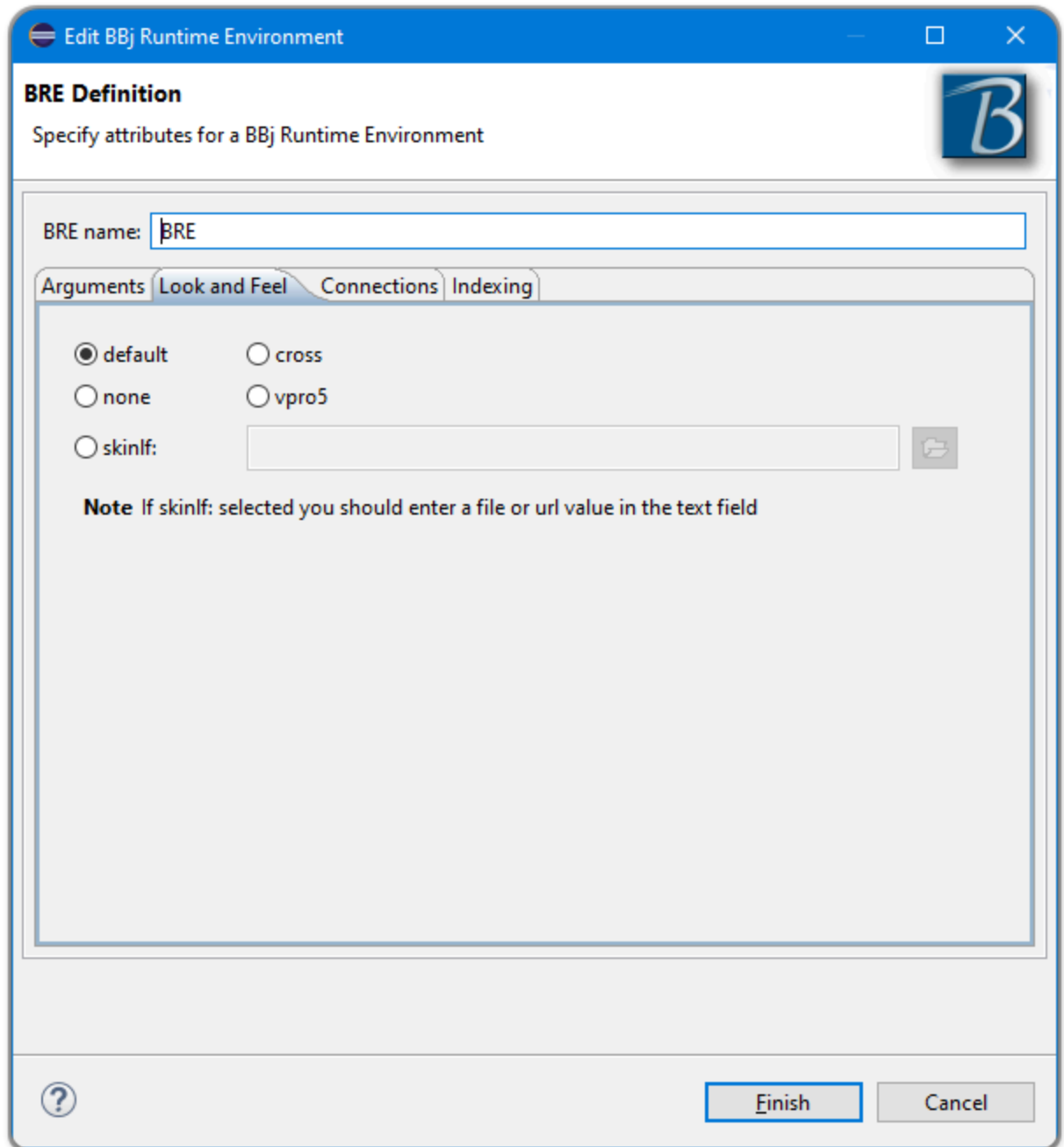


Figure 13. The Look and Feel Tab

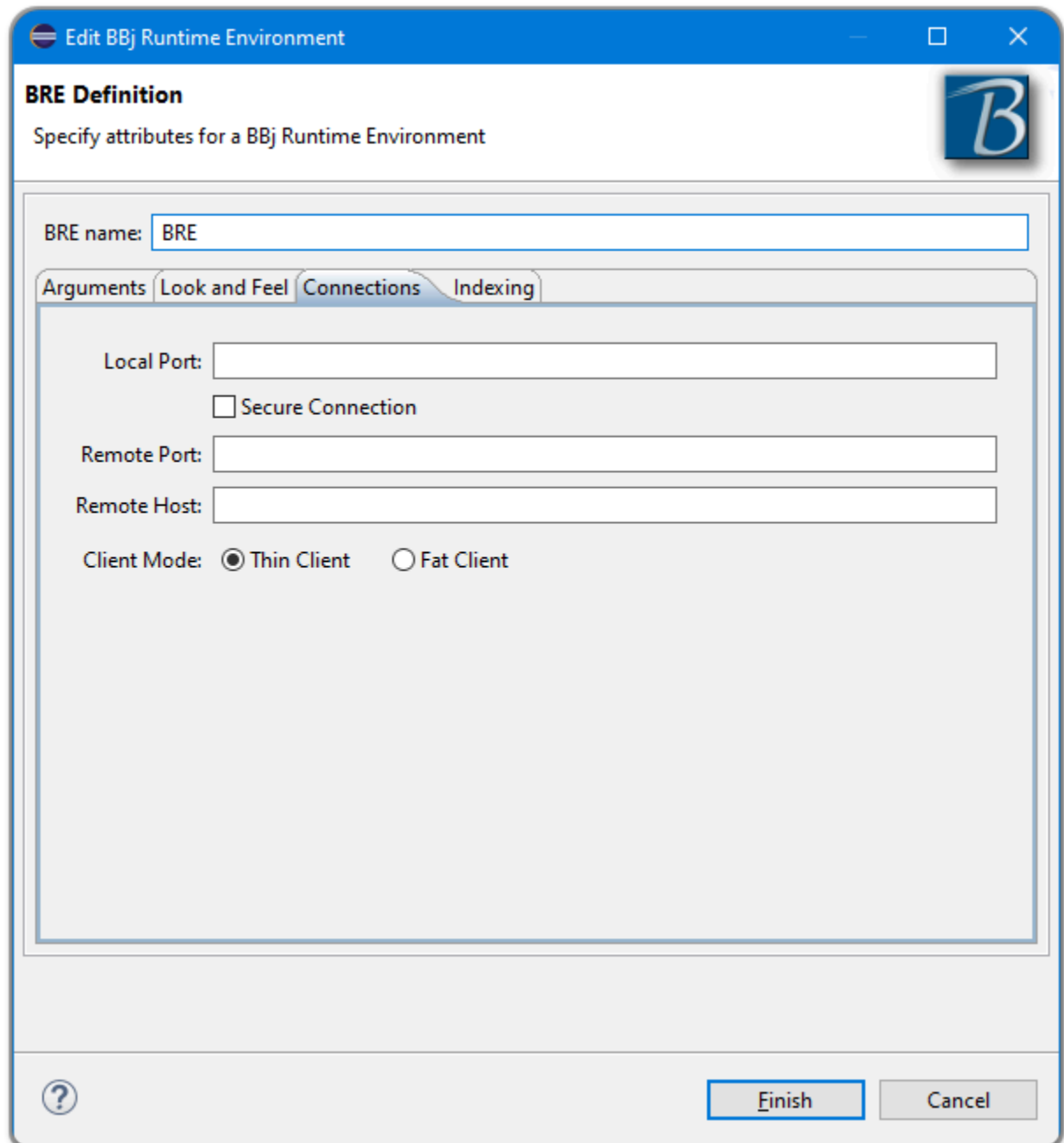
- **Look-and-Feel Theme** - select one of the **default**, **cross**, **none**, **vpro5**, and **skinlf** radio buttons to select the theme you would like to use. The **default** button is the default. If you select **skinlf**, you must specify a theme file (as a file location or a URL) to customize your look and feel. Enter the file information manually, or use the navigation button



to the right of the text entry field to navigate to and select the theme.

Connections

The following properties are available on the **Connections** tab to define the properties of a **BRE**, as shown in **Figure 14**:



Edit BBj Runtime Environment

BRE Definition

Specify attributes for a BBj Runtime Environment

BRE name: BRE

Arguments Look and Feel **Connections** Indexing

Local Port:

☐ Secure Connection

Remote Port:

Remote Host:

Client Mode: ☒ Thin Client ☐ Fat Client

? Finish Cancel

Figure 14. The Connections Tab

The following settings are available on the **Connections** tab for the properties of a **BRE**:

- **Local Port** - Specifies the Port Request Server's port (by default, it will be port 2008). Your program connects on this port to get the appropriate port to use to connect to the ThinClientProxyServer.

- **Secure Connection** - check this box to have your program use a secure connection to the Thin Client Server. The default secure port is 2103.
- **Remote Port** - Specifies the ThinClientServer port on the remote machine (by default, it will be port 2003). Your program connects to the ThinClientProxyServer on the local machine, and the TCPS makes the connection to the remote machine using the **Remote Host** and **Remote Port**.
- **Remote Host** - Specifies the machine name or IP address of the server on which the ThinClientServer or SecureThinClientServer resides. See the comments on the **Remote Port** for details on how your program uses it.
- **Client Mode: Thin Client/Fat Client** - choose whether your programs should run in thin or fat client mode. Fat client programs use more memory and resources but are able to communicate with other programs in the same JVM.

Indexing

The following properties are available on the **Connections** tab to define the properties of a **BRE**, as shown in **Figure 15**:

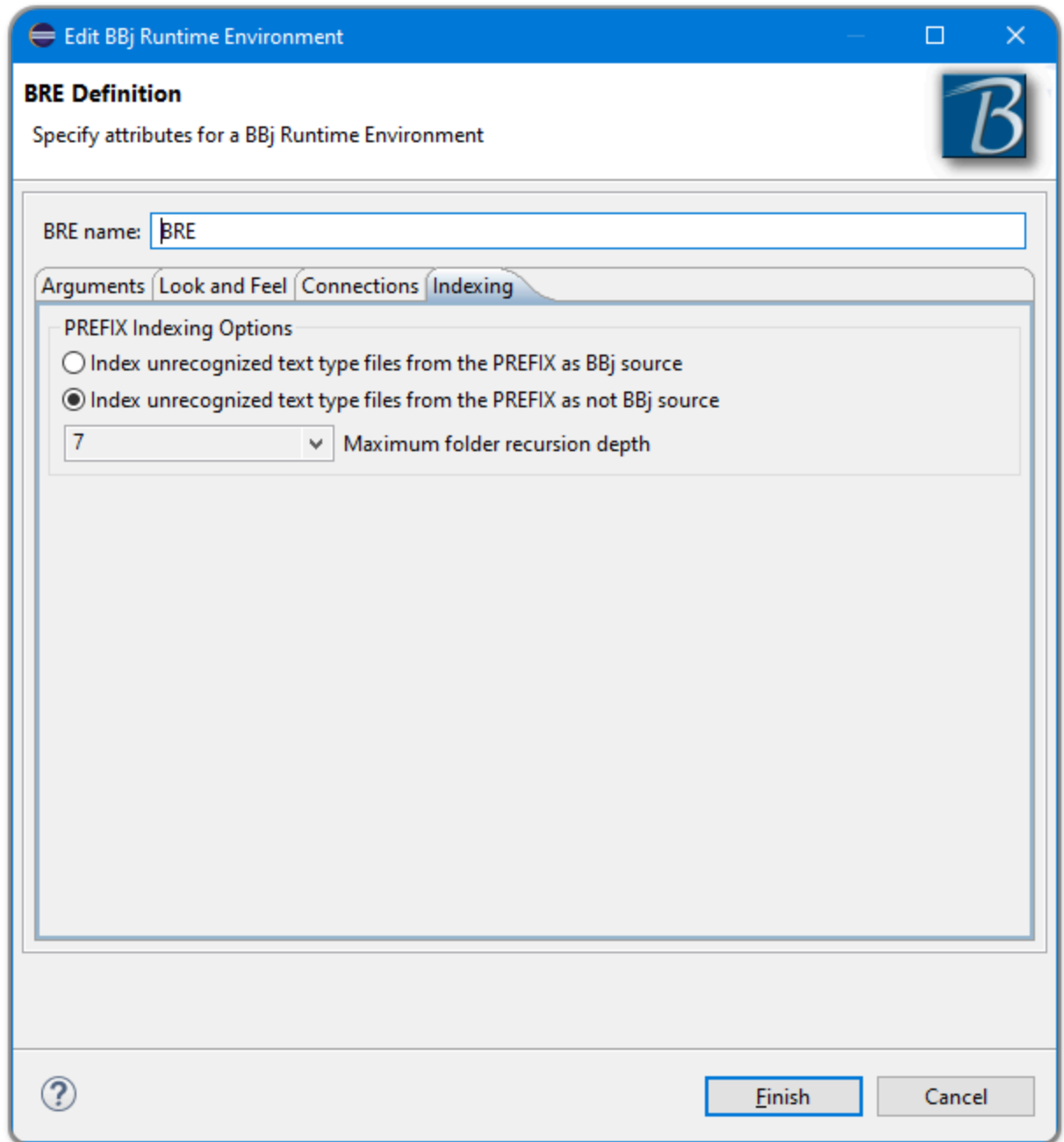


Figure 15. The Indexing Tab

For BDT to offer you code completion information about BBJ program files other than the one(s) you are editing, it must index them. If it finds a file in any folder referenced in the PREFIX (configured in BBJServices) that it cannot immediately be recognized as “containing BBJ source code” (with a recognized file extension, such as .src or .bbj), then it follows the rule configured by the properties on this tab.

To see the list of files that BDT indexes based on the PREFIX, expand the nodes under BBJ Installation/BBJ Runtime Environments/<BRE Name>/PREFIX Resources as shown in **Figure 16**:

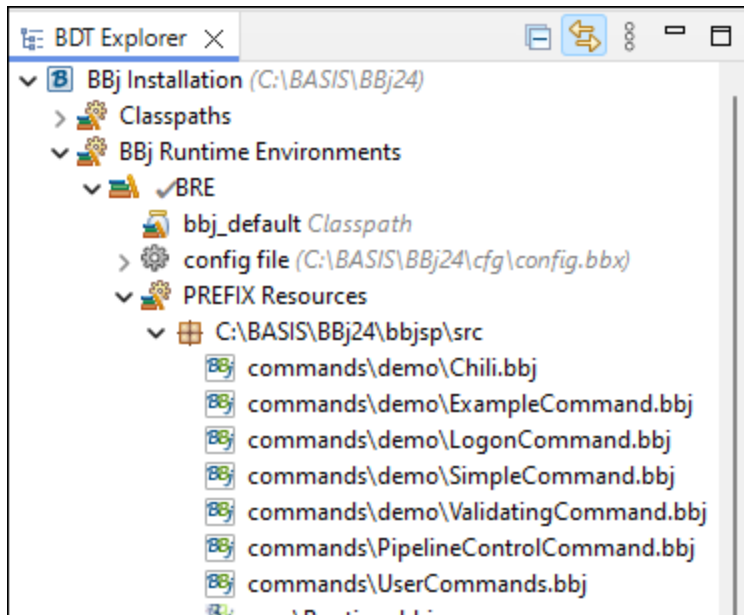


Figure 16. PREFIX Resources for a Single BRE

BDT lists the PREFIX files that it has indexed for that BRE, and that list is limited by the properties on this tab:

- **Index unrecognized text type files from the PREFIX as BBj source** - Specifies that BDT should treat any unrecognized PREFIX file as BBj source and attempt to parse and index it.
- **Index unrecognized text type files from the PREFIX as not BBj source** - Specifies that BDT should NOT treat any unrecognized PREFIX file as BBj source and should NOT attempt to parse and index it (it will skip it). This indexing setting is the default.
- **Maximum folder recursion depth** - When BDT is indexing files in the PREFIX folders, it needs to index more than just the top-level folder (since USE statements in BBj programs can drill down any number of folder levels to locate a particular file). Set this field to the number of sublevels under each PREFIX folder where you would like BDT to index files. With a large number, BDT may be slow in indexing PREFIX files. However, if you make the number too small, you may inhibit BDT from indexing files for which you would like to get code completion assistance. The default is seven levels of recursion.

NOTE: BDT only places folders in the PREFIX Resources list containing at least one BBj program file; empty folders do not appear here. Because of that, this list is NOT the list of PREFIX folders (to see that, examine your PREFIX in your config file).

Managing the Collection of BREs

Now that you know what a BRE is and how to edit one, let's look at how you select and use one of those BREs in your BBj Project. Then we will see how to create and delete a BRE.

Selecting the BRE for a BBj Project

Since the BREs for your workspace are part of the BBj Installation, they are available to every BBj Project in that workspace. Each project must have a BRE selected/set to be valid, but one BRE can be selected by any number of BBj Projects (including “none”). Let’s look at how we configure a BBj Project to use a specific BRE. In this example, two BREs are defined in our workspace/BBj installation, as shown in **Figure 17**.

- “BRE”: the BRE that BDT created for us by default when we selected the BBj installation
- “My Other BRE”: a BRE that we will show you how to create shortly

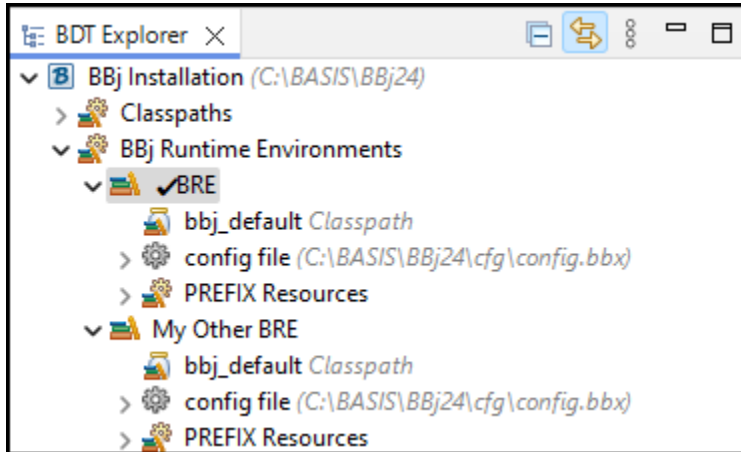


Figure 17. Setting the BRE for a BBj Project in the BDT Explorer

Notice that the “BRE” node has a check before its name, indicating that it is set as the default BRE.

To set “MyProject” to use “My Other BRE” as its BRE, follow these steps:

1. Expand the “MyProject” node in the BDT Explorer
2. Right-click the “BRE” node immediately below the project node, as shown in **Figure 18**, and select “Select BRE”:

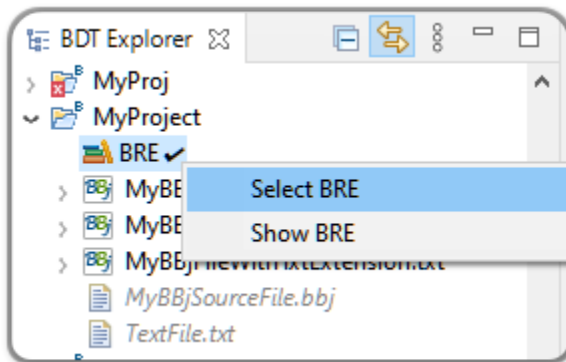


Figure 18. Setting the BRE for MyProject in the BDT Explorer

3. In the **Select BBj Runtime Environment** window, you can see which BRE is configured for MyProject in the Select Environment drop-down button, as shown in **Figure 19**:

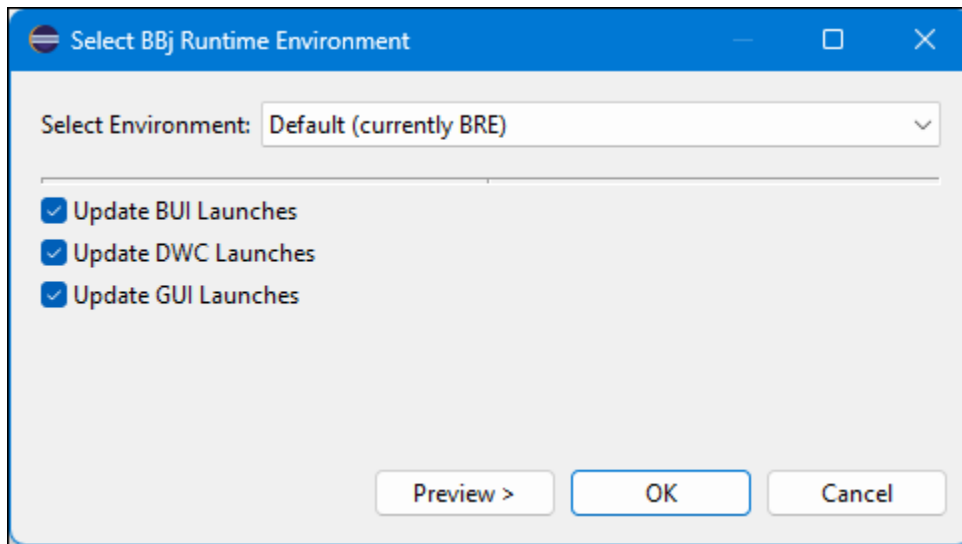


Figure 19. The BRE Configured for MyProject in the BDT Explorer

When MyProject was created, it was automatically set to use whatever BRE has the checkmark in the BBj Installation — in this case, the BRE named “BRE.” Thus the text shows “Default (currently BRE)” in **Figure 19**. If instead of coming here to select “My Other BRE,” we had changed “My Other BRE” to be the default BRE, then every BBj Project in this workspace, like MyProject, that is configured to use the default BRE, would have changed from using “BRE” (the old default) to using “My Other BRE” (the new default). By coming to the project node and selecting a BRE there, we only impact MyProject.

The options for selecting a BRE for MyProject are shown in **Figure 20**:

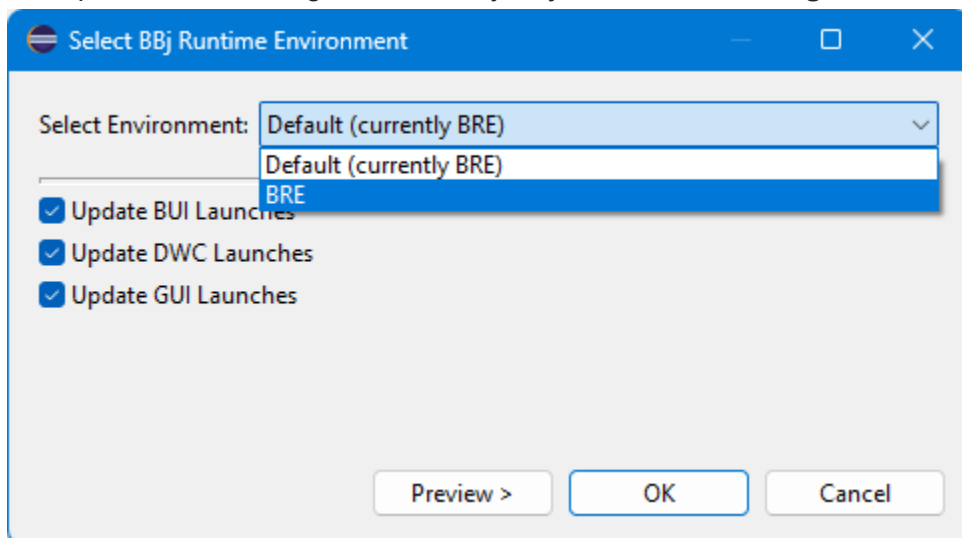


Figure 20. Setting the BRE for MyProject in the BDT Explorer

We can configure MyProject by selecting one of the three entries shown in the drop-down list above:

- Select “Default (currently BRE)” to make no change. In this case, MyProject will continue to use whatever BRE is set as the default, and MyProject’s BRE will change in the future each time you set a different default BRE for the BBJ Installation.
- Select “BRE,” and MyProject will use the BRE named “BRE” as long as that BRE exists in the BBJ Installation, regardless of any future change to which BRE is set as the default.
- Select “My Other BRE,” and MyProject will use the new BRE named “My Other BRE” as long as that BRE exists in the BBJ Installation, regardless of any future change to which BRE is set as the default.

Note: Depending on the number of BREs that are defined in your BBJ Installation, you may have more or fewer entries below the “Default (currently BRE)” entry.

4. Check all of the checkboxes, “Update BUI Launches”, “Update DWC Launches”, and “Update GUI Launches,” unless BASIS technical support instructs you to do otherwise. This ensures that any Eclipse launch configurations defined for the BBJ programs in MyProject are also updated to use the newly chosen BRE.
5. Optionally, click the **[Preview >]** button to display information about the actual changes BDT will make as it selects the new BRE (as shown in **Figure 21**):

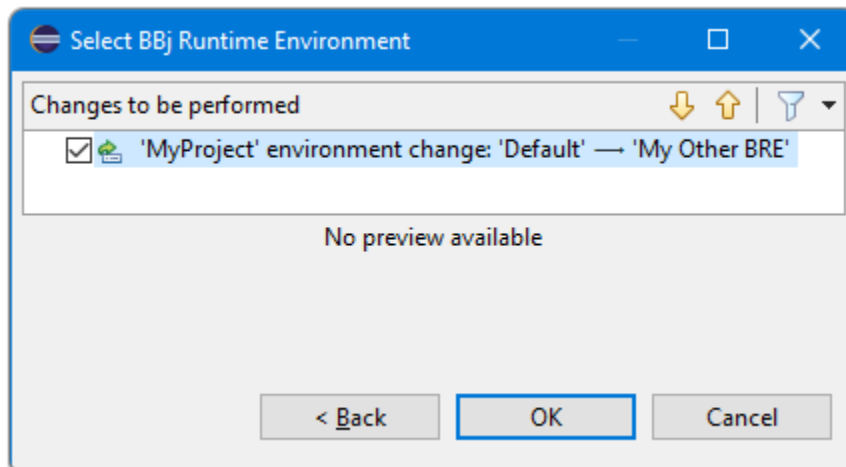


Figure 21. Previewing the Changes to Select “My Other BRE”

6. When you are satisfied, click the **[OK]** button to save your change. Once this process is completed, MyProject will display “My Other BRE” as its selected BRE (see **Figure 22**).

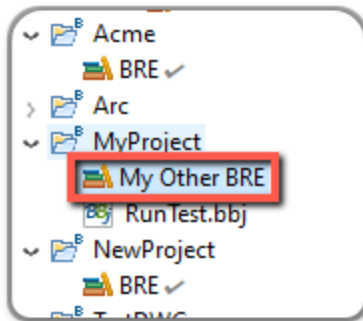


Figure 22. MyProject Set to Use “My Other BRE”

Notice that there is no checkmark to the right of the “My Other BRE” node; MyProject is no longer using the default BRE.

Creating a BRE

Let’s look at each of the ways that are available for you to create a BRE.

Creating a New BRE

To create a new BRE with all of its properties set to their default values, follow these steps:

1. Expand the **BBj Installation** node so that the **BBj Runtime Environments** node is visible in the BDT Explorer.
2. Right-click the **BBj Runtime Environments** node as shown in **Figure 23**, and select **Create New BRE**:

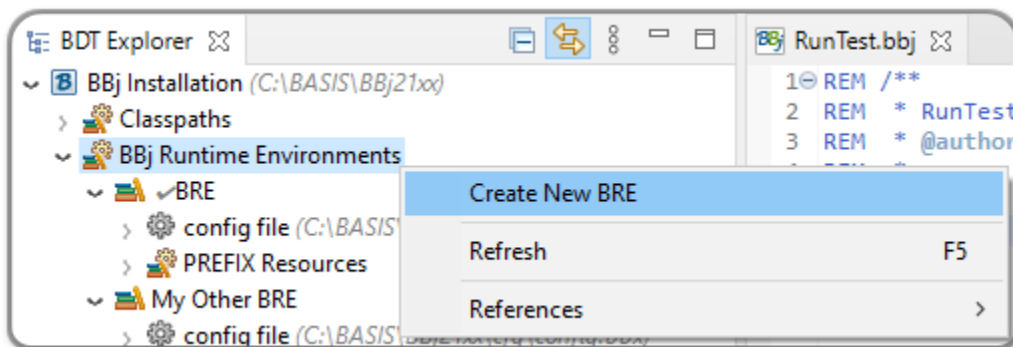


Figure 23. Creating a New BRE

This action opens the **Add BBj Runtime Environment** window shown in **Figure 24**.

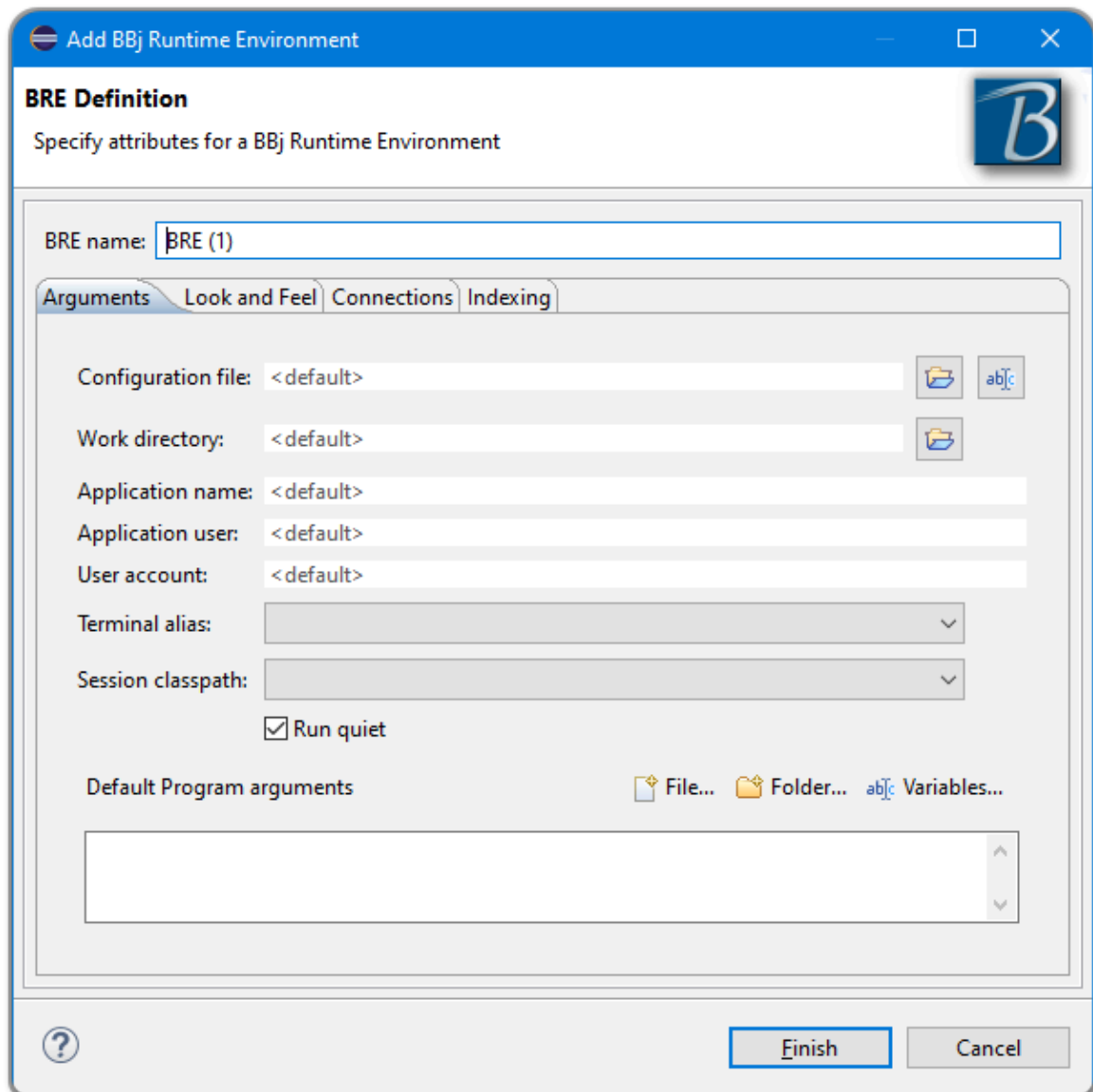


Figure 24. The Add BBj Runtime Environment Window

By default, BDT suggests the name “BRE (1)” for this new BRE since there is already one named “BRE,” and BRE names must be unique within the workspace. You can, of course, accept this name or change it to a more meaningful one.

Since we discussed the properties in the section on [Configuring a BRE](#), we won’t repeat that information here.

3. Set the properties of the new BRE to support your runtime needs and click the **[Finish]** button.

Your new BRE now appears in the list under the **BBj Runtime Environments** node and is available to select in specific BBj projects.

Copying a BRE

To make a copy of an existing BRE (with all of the same properties except for the name), follow these steps:

1. Expand the **BBj Installation > BBj Runtime Environments** node so that the BRE you want to copy is visible in the BDT Explorer.
2. Right-click the node of the BRE as shown in **Figure 25**, and select **Copy**:

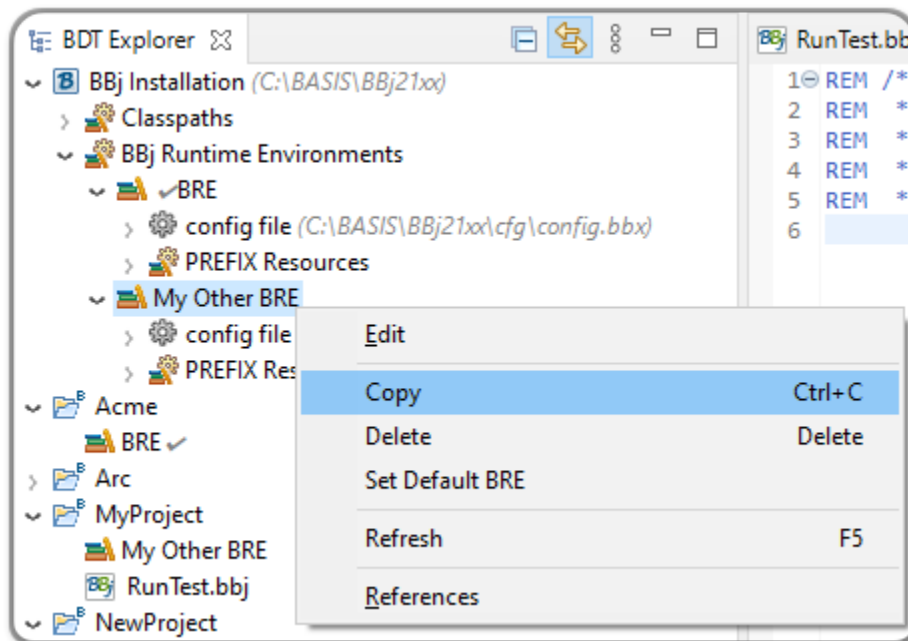


Figure 25. Copying the Existing “My Other BRE” BRE

This option opens the **Copy BBj Runtime Environment** window, which is similar to the **Add BBj Runtime Environment** window shown in **Figure 24** (except that in this case, it is suggesting the name of the BRE to be copied with “ (1)” appended).

Since we discussed the properties in the section on [Configuring a BRE](#), we won’t repeat that information here.

3. Modify the properties of the duplicate BRE to support your runtime needs and click the **[Finish]** button.

Your new BRE now appears in the list under the **BBj Runtime Environments** node and is available to select in specific BBj projects.

Deleting a BRE

To delete an existing BRE, follow these steps:

1. Expand the **BBj Installation > BBj Runtime Environments** node so that the BRE you want to delete is visible in the BDT Explorer.
2. Right-click the BRE node and select the **Delete** menu item shown in **Figure 11**.

This action opens the **Delete BBJ Runtime Environment** window shown in **Figure 26**.

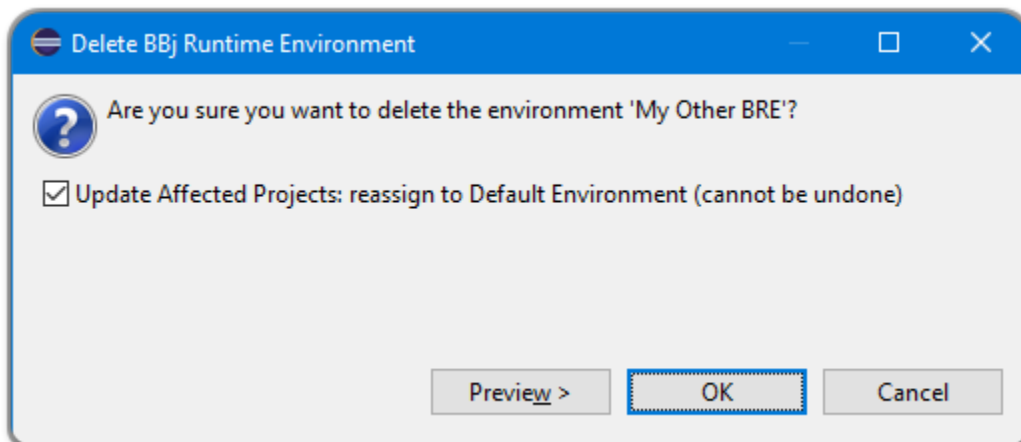


Figure 26. Deleting “My Other BRE”

Notice that the default is to reset all of the BBJ Projects that currently use “My Other BRE” to use the default BRE instead, and these are not “undoable actions.”

3. Clicking the **[Preview >]** button causes the window to show expanded details on the actions BDT will take to delete that BRE, as seen in **Figure 27**:

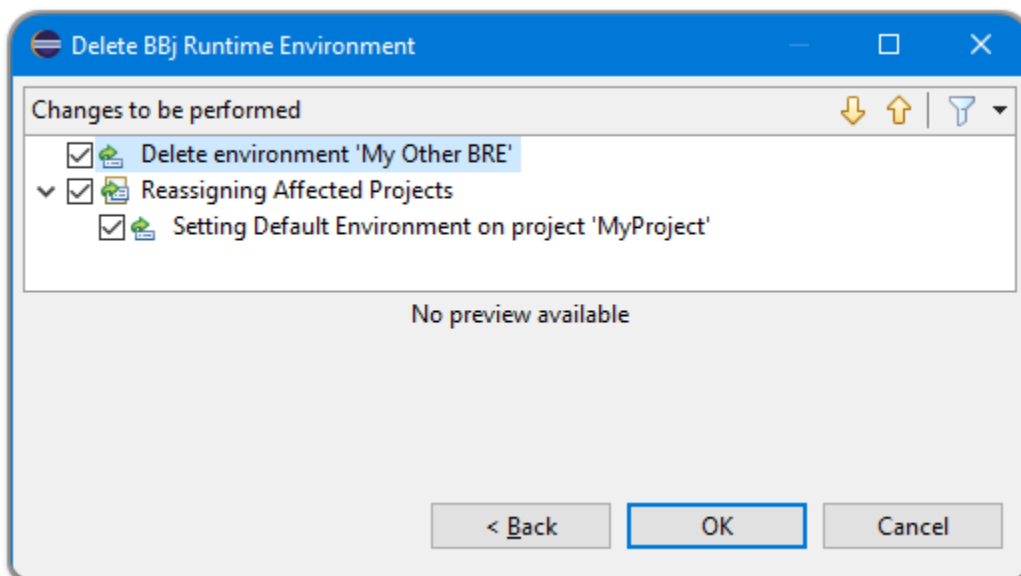


Figure 27. Previewing the Deletion of “My Other BRE”

This preview shows that only one BBJ Project will be affected: “My Project.”

4. Click the **[OK]** button to complete the deletion. “My Other BRE” is no longer listed as a BRE for this BBJ Installation, and “My Project” is now using the default “BRE.”

Note: You cannot delete the BRE currently set as the default. If you attempt to do so, you will be warned, as seen in **Figure 28**:

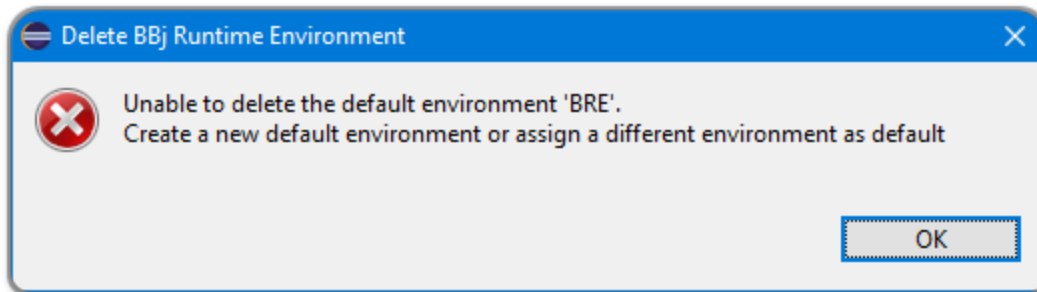


Figure 28. You Cannot Delete the Default BRE

To delete the default BRE, you must first set another BRE as the default BRE.

Seeing What BBJ Projects Are Using

At times, you may want to know which BBJ Projects are configured to use a particular BRE or a particular config file. The slow way to answer that question is to simply expand each BBJ Project in your workspace and hover over its BRE node. If you only have a few projects, this may work for you. However, if you have many projects in this workspace, you need something faster. BDT lets you see hierarchical trees with this information in the following three cases.

The BBJ Projects Using Every BRE

You can ask BDT to show you a tree with a node for each BRE, where each BRE can expand to show all of the BBJ Projects that are configured to use it. To see this, simply right-click the BBJ Runtime Environments node as shown in **Figure 15**, and select **References > Projects**. BDT will open the Search View with the tree shown in **Figure 29**:

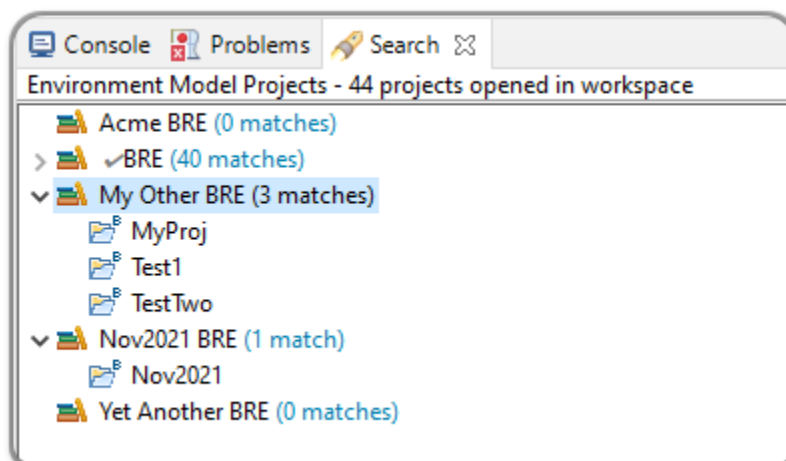


Figure 29. All of the BBJ Projects that Use Each BRE

In this example, you can see that there are BREs in this workspace, and there are between zero and forty BBJ Projects configured to any specific BRE. By expanding each BRE node, you can see the names of the BBJ Projects that use that BRE.

The BBJ Projects Using Each Configuration File

You can ask BDT to show you a tree with a node for each config file set in either:

- the BBJ Installation itself (the default config file)
- any BRE in the BBJ Installation

Each config file node can expand to show all of the BBJ Projects that are configured to use it. To see this, simply right-click the BBJ Runtime Environments node as shown in **Figure 15**, and select **References > Configuration Files**. BDT will open the Search View with the tree shown in **Figure 30**:

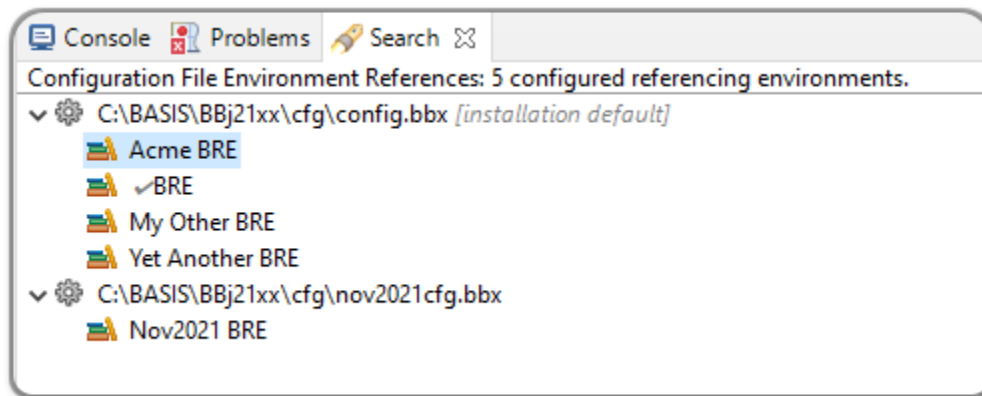


Figure 30. All of the BBJ Projects Using Each BBJ Configuration File

In this example, you can see that there are two config files used by any BRE in the workspace, and the full path to each. By expanding each config file node, you can see the names of the BRES that use that file.

The BBJ Projects Using a Specific BRE

You can ask BDT to show you all of the BBJ Projects that are configured to use a specific BRE. To see this, simply right-click that BRE in the BDT Explorer, as shown in **Figure 11**, and select **References**. BDT will open the Search View with the tree shown in **Figure 31**:

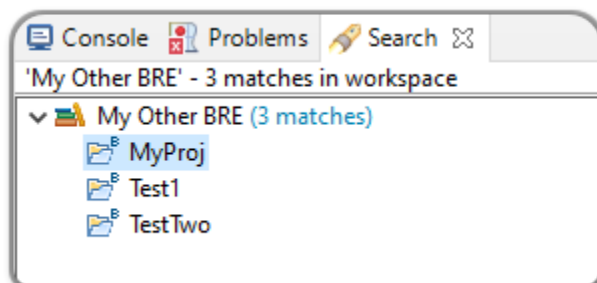


Figure 31. All of the BBJ Projects Using “My Other BRE”

In this example, you can see three BBJ Projects are configured to use “My Other BRE,” and by expanding the BRE node, you can see their names.