SUPERSEDED
Superseded
The BBjToJavadoc utility has been superseded by the BBjDocsGenerator

BBj Documentation is as Easy as JavaDocs

Wikipedia defines Javadoc as "a documentation generator from Sun Microsystems [now Oracle] for generating API documentation in HTML format from Java source code. The HTML format is used to add the convenience of being able to hyperlink related documents together. The 'doc comments' format used by Javadoc is the de facto industry standard for documenting Java classes."

With the help of a new BASIS utility, BBjToJavadoc, and a little effort on your part, you now have a second motivation to put comments in your code, because BBjToJavadoc also gives you the ability to rapidly create API documentation from your BBj® application code using this new BASIS-supplied Javadoc documentation generator.

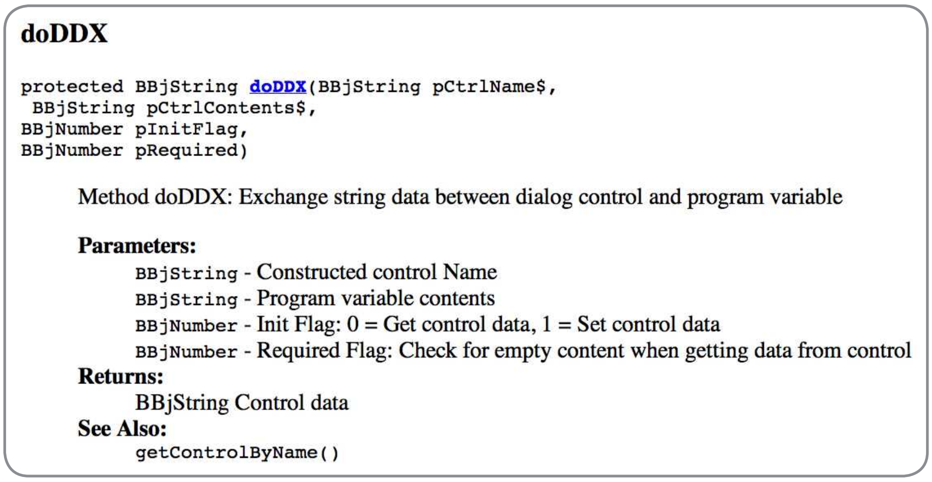
Figure 1 shows an example of doc comments in BBj code. In this example, the "block tags" are **`@param`**, **`@return`**, and **`@see`**.

```
rem /**
rem * Method doDDX:
rem * Exchange string data between dialog control and program variable
rem * @param BBjString Constructed control Name
rem * @param BBjString Program variable contents
rem * @param BBjNumber Init Flag: 0=Get control data, 1=Set control data
rem * @param BBjNumber Required Flag: Check for empty content when getting data from control
rem * @return BBjString Control data
rem * @see getControlByName()
rem */
method protected BBjString doDDX(BBjString pCtrlName$, BBjString pCtrlContents$, BBjNumber pInitFlag, BBjNumber pRequired)
```

Figure 1. BBj code incorporating doc comments

By Ralph Lance
Software Engineer

The resulting HTML generated would then look like **Figure 2.**



doDDX

```
protected BBjString doDDX(BBjString pCtrlName$,
 BBjString pCtrlContents$,
BBjNumber pInitFlag,
BBjNumber pRequired)
```

Method doDDX: Exchange string data between dialog control and program variable

Parameters:

`BBjString` - Constructed control Name
`BBjString` - Program variable contents
`BBjNumber` - Init Flag: 0 = Get control data, 1 = Set control data
`BBjNumber` - Required Flag: Check for empty content when getting data from control

Returns:

BBjString Control data

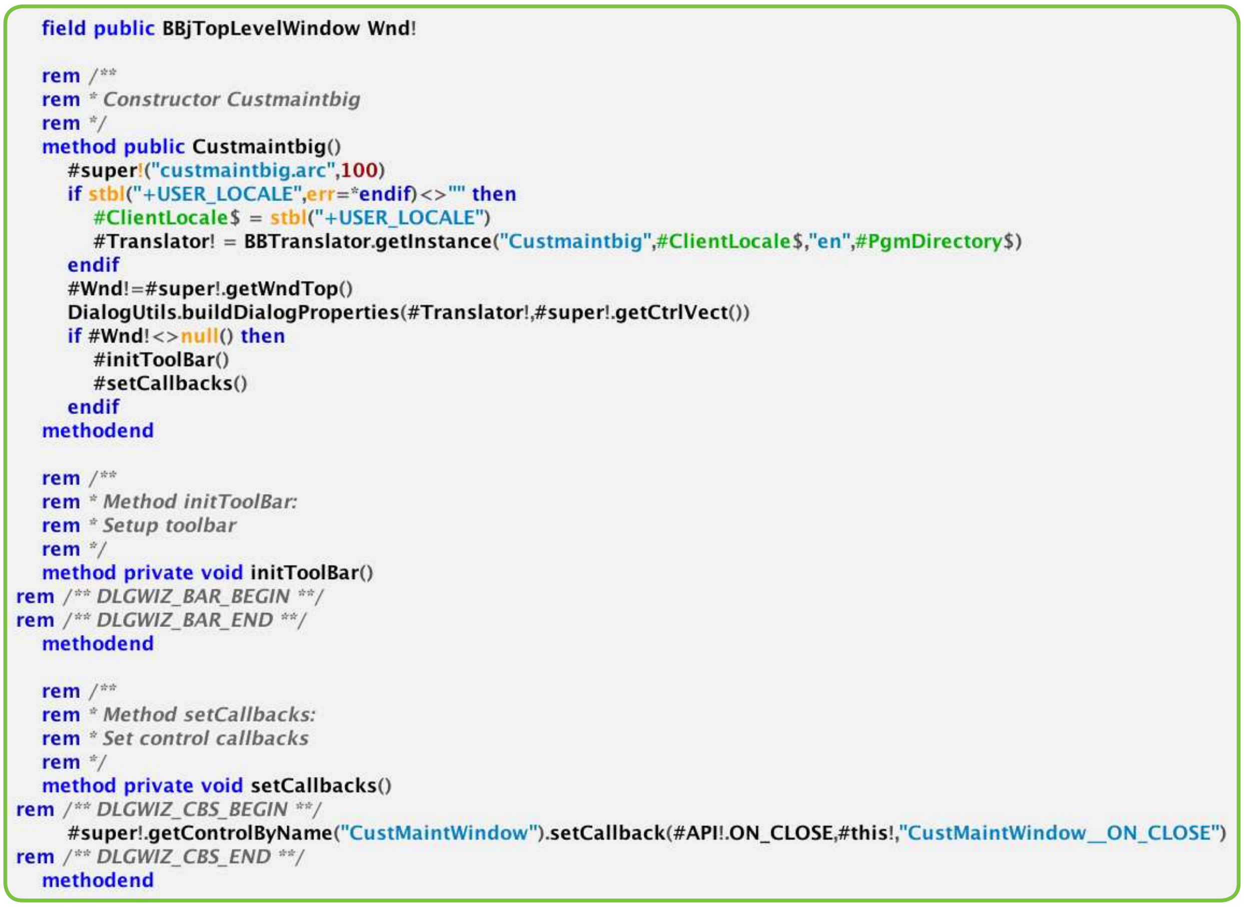
See Also:

`getControlByName()`

Figure 2. Resultant HTML API documentation example

With BBjToJavadoc, BASIS has made it possible for you to embed similar documentation comments in your BBj programs and generate your own documentation in HTML Javadoc format. In particular, your BBj custom classes can take advantage of some of the more advanced documentation capabilities, allowing you to navigate easily via hyperlinks among your classes. Because the Javadoc engine from the Java JDK toolset is used in the process, any valid document tag can be embedded in the doc comments.

Figure 3 is an example of a BBj custom dialog class with some doc comments that the Dialog Wizard embedded for us. Follow the required few simple steps below to generate the corresponding Javadoc documentation.



```
   field public BBjTopLevelWindow Wnd!

   rem /**
   rem * Constructor Custmaintbig
   rem */
   method public Custmaintbig()
      #super!("custmaintbig.arc",100)
      if stbl("+USER_LOCALE",err=*endif)<>"" then
         #ClientLocale$ = stbl("+USER_LOCALE")
         #Translator! = BBTranslator.getInstance("Custmaintbig",#ClientLocale$,"en",#PgmDirectory$)
      endif
      #Wnd!=#super!.getWndTop()
      DialogUtils.buildDialogProperties(#Translator!,#super!.getCtrlVect())
      if #Wnd!<>null() then
         #initToolBar()
         #setCallbacks()
      endif
   methodend

   rem /**
   rem * Method initToolBar:
   rem * Setup toolbar
   rem */
   method private void initToolBar()
rem /** DLGWIZ_BAR_BEGIN **/
rem /** DLGWIZ_BAR_END **/
   methodend

   rem /**
   rem * Method setCallbacks:
   rem * Set control callbacks
   rem */
   method private void setCallbacks()
rem /** DLGWIZ_CBS_BEGIN **/
      #super!.getControlByName("CustMaintWindow").setCallback(#API!.ON_CLOSE,#this!,"CustMaintWindow__ON_CLOSE")
rem /** DLGWIZ_CBS_END **/
   methodend
```

Figure 3. BBj custom dialog class with embedded doc comments

Step 1. Start the Wizard

To start the wizard, run the program called **BBjToJavadocWizardRun.bbj** located in the **<BBjHome>/utils/BBjToJavadoc** folder.

Once the wizard launches (**Figure 4**), stipulate where to store the generated documentation and choose the level of scope, or visibility, the documentation includes - public, protected, or private. Choosing a 'private' level of scope documents everything.

Lastly, choose one or more non-tokenized BBj source files to document.

Step 2. Set Options

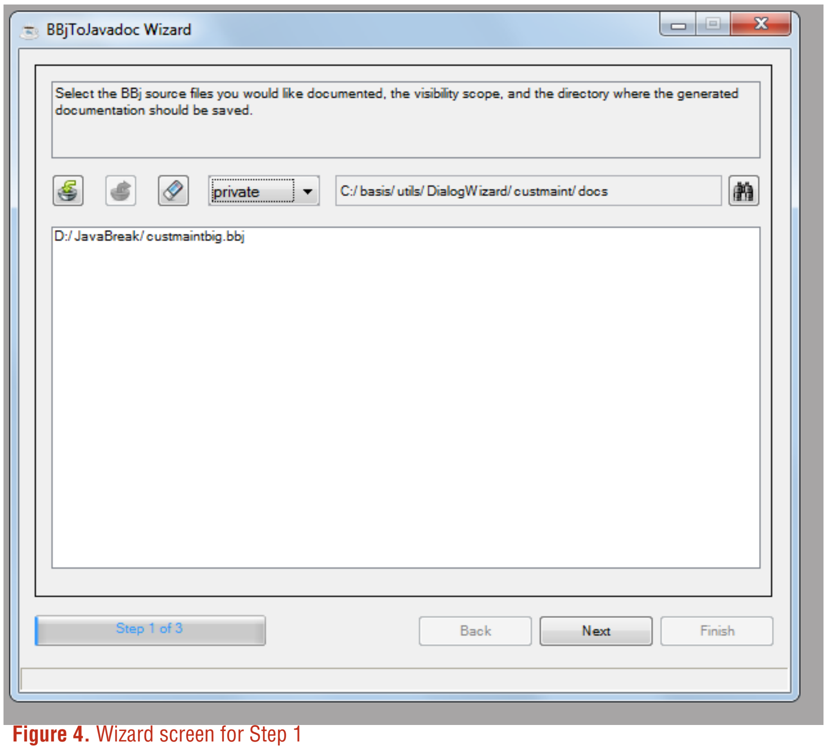
After clicking the [Next] button, proceed to the second screen (**Figure 5**) to set options for the Javadoc program. Read more about these options in the online Javadoc documentation.

Step 3. Generate the Docs

Now, you are ready to generate the actual documentation, so click [Next] and then [Finish] to display the completed screen shown in (**Figure 6**). You have the option to show the results of the generation immediately upon completion.

If desired, the Wizard displays the nicely structured HTML documentation in your default browser (**Figure 7**). You can easily navigate between the different regions of the document - the field summary, constructors, and the methods themselves. Clicking the hyperlinks for methods or fields in the left navigation pane takes you to a copy of the code that defines the entity.

So just that easily, the BBjToJavadoc utility has done all the work for you, simply creating a robust hyperlinked documentation system derived from your own code.



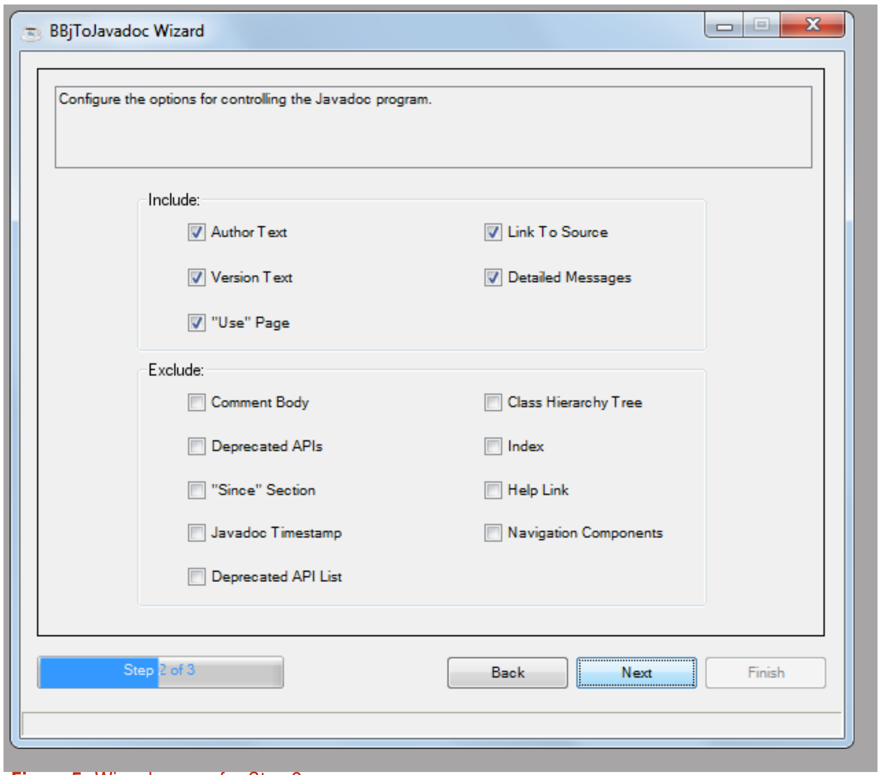
Select the BBj source files you would like documented, the visibility scope, and the directory where the generated documentation should be saved.

private

C:/ basis/ utils/ DialogWizard/ custmaint/ docs

D:/ JavaBreak/ custmaintbig.bbj

Figure 4. Wizard screen for Step 1



Configure the options for controlling the Javadoc program.

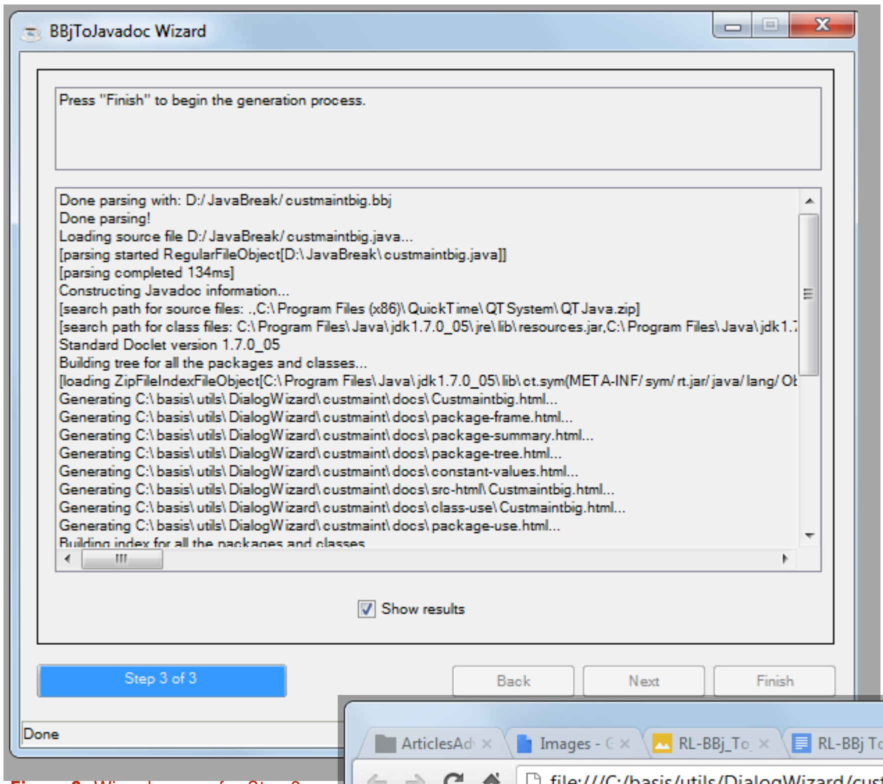
Include:

- ✓ Author Text
- ✓ Link To Source
- ✓ Version Text
- ✓ Detailed Messages
- ✓ "Use" Page

Exclude:

- ☐ Comment Body
- ☐ Class Hierarchy Tree
- ☐ Deprecated APIs
- ☐ Index
- ☐ "Since" Section
- ☐ Help Link
- ☐ Javadoc Timestamp
- ☐ Navigation Components
- ☐ Deprecated API List

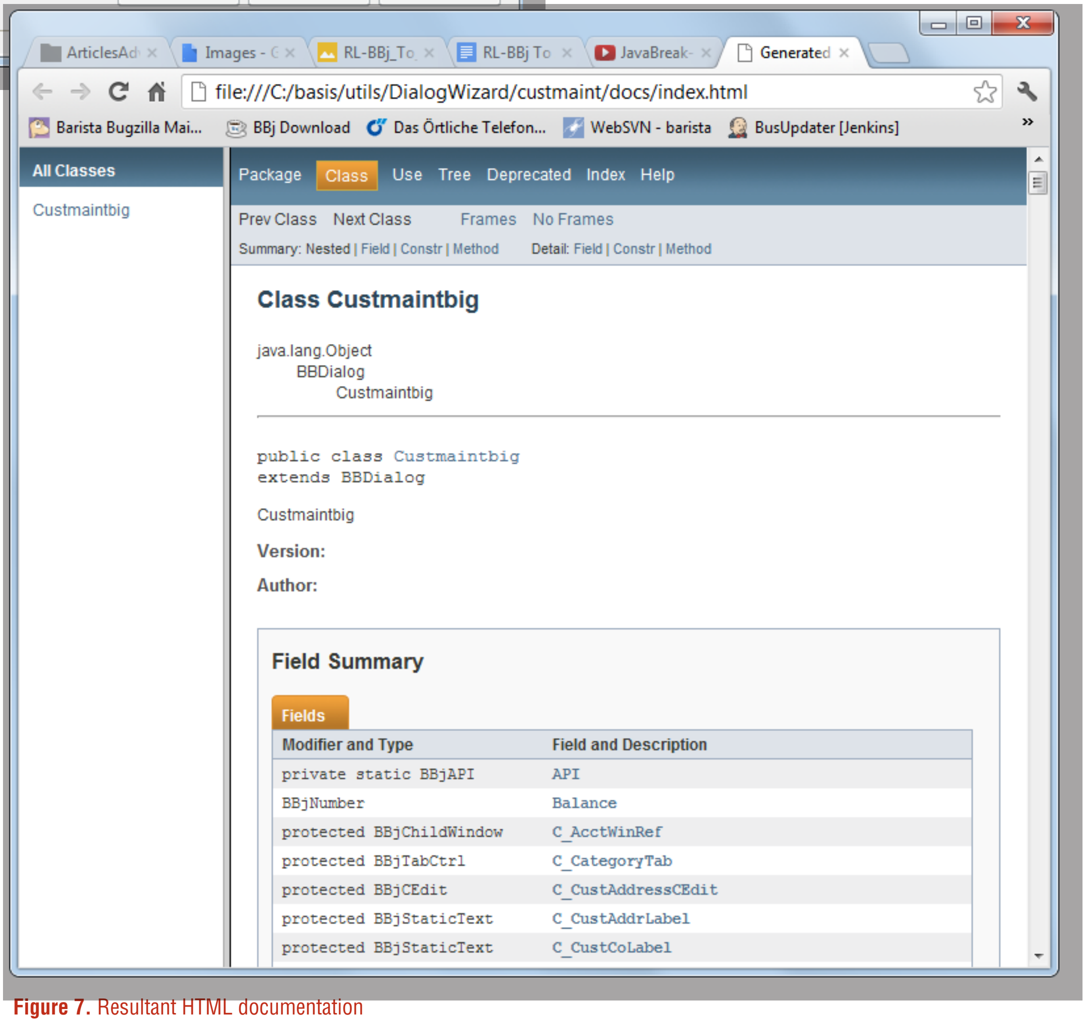
Figure 5. Wizard screen for Step 2



Press "Finish" to begin the generation process.

```
Done parsing with: D:/ JavaBreak/ custmaintbig.bbj
Done parsing!
Loading source file D:/ JavaBreak/ custmaintbig.java...
[parsing started RegularFileObject[D:\ JavaBreak\ custmaintbig.java]]
[parsing completed 134ms]
Constructing Javadoc information...
[search path for source files: .,C:\ Program Files (x86)\ QuickTime\ QT System\ QT Java.zip]
[search path for class files: C:\ Program Files\ Java\ jdk1.7.0_05\ jre\ lib\ resources.jar,C:\ Program Files\ Java\ jdk1.7
Standard Doclet version 1.7.0_05
Building tree for all the packages and classes...
[loading ZipFileIndexFileObject[C:\ Program Files\ Java\ jdk1.7.0_05\ lib\ ct.sym(META-INF/ sym/ rt.jar/ java/ lang/ Ob
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ Custmaintbig.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ package-frame.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ package-summary.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ package-tree.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ constant-values.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ src-html\ Custmaintbig.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ class-use\ Custmaintbig.html...
Generating C:\ basis\ utils\ DialogWizard\ custmaint\ docs\ package-use.html...
Building index for all the packages and classes
```

Figure 6. Wizard screen for Step 3



Class Custmaintbig

```
java.lang.Object
    BBDialog
        Custmaintbig
```

```
public class Custmaintbig
extends BBDialog
```

Custmaintbig

Version:

Author:

Field Summary

Fields

Modifier and Type	Field and Description
`private static BBjAPI`	`API`
`BBjNumber`	`Balance`
`protected BBjChildWindow`	`C_AcctWinRef`
`protected BBjTabCtrl`	`C_CategoryTab`
`protected BBjCEdit`	`C_CustAddressCEdit`
`protected BBjStaticText`	`C_CustAddrLabel`
`protected BBjStaticText`	`C_CustCoLabel`

Figure 7. Resultant HTML documentation

Summary

BBjToJavadoc is a documentation generator from BASIS for generating API documentation in HTML format from BBj source code. The HTML format is used to add the convenience of being able to hyperlink related documents together. The 'doc comments' format used by BBjToJavadoc is the BASIS standard for documenting BBj custom classes. While we can't actually remove the often neglected task of documenting your code from your 'to-do' list, this BASIS building block utility makes it as easy as possible for you to write self-documenting object-oriented code. ■

For more information, visit
- BBDocsGenerator User Guide
- Oracle Java API Documentation Generator
- Oracle Javadoc Tool