Filling a List Widget with Data Programmatically on a Custom Dialog

By Jennifer DeLockery

1. Creating a Dialog and Installing on a Class

- Create a new dialog by clicking the easel icon on the main VisualWorks window (see Figure 1).

Figure 1: Creating a New Dialog

- In the GUI Painter Tool, click select Edit -> Install. You will see a dialog pop up that looks like Figure 2.

Figure 2: Install Dialog

- Click the “binoculars” icon to search for a class on which to install the dialog.

- Select the “Create” folder tab (see Figure 3).
  - Choose a namespace under “Location”.
  - Enter a name for the dialog.
  - Enter a superclass.
    - For a main GUI dialog, use ApplicationModel.
    - For a custom dialog, use SimpleDialog.
  - Select OK.

- Select OK back on the install dialog (see Figure 2).
2. Prepare the Main Dialog to Open a Custom Dialog

- If you do not have a main GUI dialog, create one now (see Part 1), and call it MainDlg.

- On MainDlg, add a button called “Open” (see Figure 4).
  - In the GUI Painter Tool, select “ActionButton1” (see Figure 6).
  - Under the “Basics” tab:
    - Type “Open” in the “String:” box.
    - Type “openDlg” in the “Action:” box.
    - Rename the ID to #ActionButtonOpen.

- Go to Edit -> Define in the GUI Painter Tool. Make sure “Add Initialization” is checked (see Figure 5). Select OK.
• Install your changes by going to Edit -> Install in the GUI Painter Tool. Just select “OK”.

• In the System Browser (select F5 in the main VisualWorks window), browse to wherever you placed MainDlg. NOTE: My classes never get put where I create them. They always get placed in the “(none)” package. If you can’t find your class where you put it, it may be there.

• Under the “actions” protocol, you will see the “openDlg” method that we defined (see Figure 7). This method gets called whenever the user presses the “Open” button on the main GUI. In this method, we will later create a new instance of our custom dialog and show it to the user.

• Add an instance variable for MainDlg called “listOfData” (see Figure 8).
• Add initialization code for listOfData as in Figure 9 (you will have to add the “initialize-release” protocol and create the “initialize” method within the protocol.

Figure 8: Adding listOfData Instance Variable

Figure 9: Initialization for MainDlg (make sure you have this)
3. Creating a Custom Dialog

- Create a new custom dialog called CustomDlg (see Part 1). 
  NOTE: Make sure you create the dialog from the "SimpleDialog" superclass when you install it onto a class.

- Add a list widget and an action button to CustomDlg (see Figure 10).

**Figure 10: A List Widget and Action Button Added to CustomDlg**

- In the GUI Painter Tool, set the following properties in the "Basics" tab:
  - ActionButton1 (see Figure 11):
    - String: OK
    - Action: #accept
    - ID: #ActionButtonAccept
  - List1 (see Figure 12):
    - Aspect: #data
    - ID: #ListData

**Figure 11: Action Button Properties**

**Figure 12: List Properties**

- Define the variables (see Part 2) and install the changes (see Part 1 if you haven't installed for the first time).

- In the CustomDlg class, look in the "aspects" protocol and select, "data". Make sure this method looks like Figure 13.
• Create an “accessing” protocol in CustomDlg. Add a modifier accessor as in Figure 14.

Figure 13: Data Method in CustomDlg

```
/* This method was generated by UIdentifier. Any edits made here */
/* may be lost whenever methods are automatically defined. The */
/* initialization provided below may have been prompted by an */
/* initialize method. */

?data isNil
  if True:
    [data := SelectionList new]
  if False:
    [data]
```

Figure 14: Adding an Accessor for Data

```
/* This method sets data to aList. */

data := SelectionList with: aList.
```

Method: #data (accessing) | Package: none
4. Putting It All Together (Hooking Up MainDlg to CustomDlg)

- Return to the “openDlg” method in MainDlg.
- Enter the code in Figure 15.

**Figure 15: The openDlg Method**

```cpp
openDlg
    "This method opens a dialog with a list."
    OpenDlg result |
    "Create a new CustomDlg object."
    dlg := CustomDlg new.
    "Set the 'data' instance variable in CustomDlg to the data we want to display, listOFData."
    dlg data listOFData.
    "Open the dialog and get the result when the user closes the dialog."
    result := dlg openFrom: (CustomDlg interfaceSpecFor: #windowSpec).
    "In this case, the result will always be true since the user only has an option of OK. The selection is printed to the Transcript."
    result ifTrue: [Transcript cr, show: 'The selection was:', (dlg data selection) printString, cr].
```

- Go to the “ResourceFinder” to find and select “MainDlg”. Choose “Start” (see Figure 16).

**Figure 16: Using the Resource Finder to Start MainDlg**
• The MainDlg should run (see Figures 17, 18, and 19). Try it out!

Figure 17: Step 1 – The MainDlg GUI

Figure 18: Step 2 – The CustomDlg Appears at a click of “Open”

Figure 19: Step 3 – The Selection Appears in the Transcript