FamilyTreeApp

Download the source for the reference application (described briefly in "FamilyTreeApp Reference Application" on page 10 of our book) at `https://java.net/projects/nbfamilytreeapp`.

1. Instructions for Accessing the FamilyTreeApp Code

You can access the project directly from the NetBeans IDE and Subversion source control tool using the following steps.

1. From the NetBeans IDE top-level menu, select **Team** | **Subversion** | **Checkout . . .**, as shown in Figure 1.

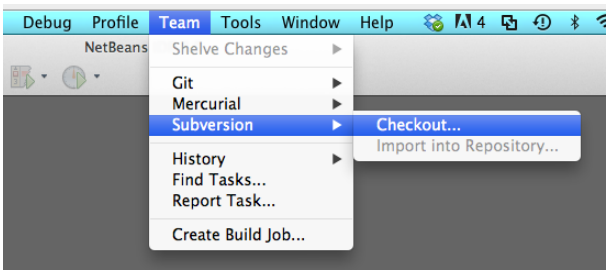

Figure 1. Team | Subversion | Checkout . . . menu item

2. For the Checkout process, Subversion displays the Subversion Repository dialog, as shown in Figure 2. For the Repository URI, specify

   `https://svn.java.net/svn/nbfamilytreeapp~src`

   Provide your user name and password (or leave blank for anonymous access). Adjust the Proxy configurations if necessary. Click **Next**.

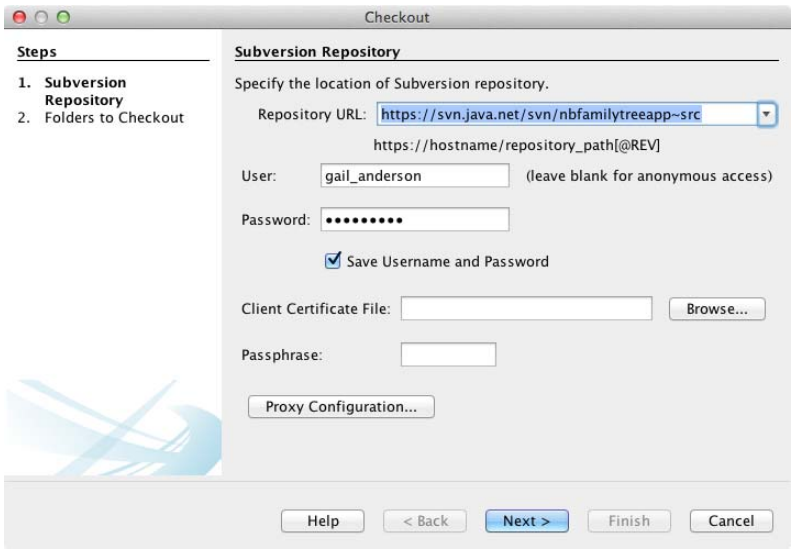

Figure 2. Configure the Subversion Repository settings

3. NetBeans displays the Folders to Checkout dialog. In the Repository Folders field, click the **Browse** button. In the Browse Repository Folders dialog, select the top-level folder, **FamilyTreeApp**, as shown in Figure 3 and click **OK**.

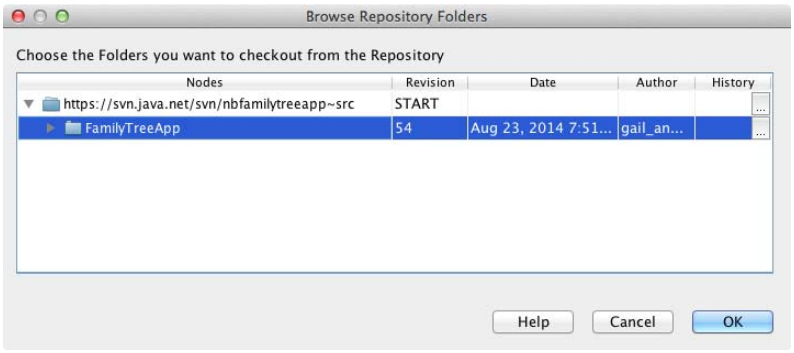

Figure 3. Choose the FamilyTreeApp folder

4. For the Local Folders field, click **Browse** and navigate to the destination folder on your local system. Figure 4 shows both the Repository Folders and Local Folder specified. Click **Finish**.

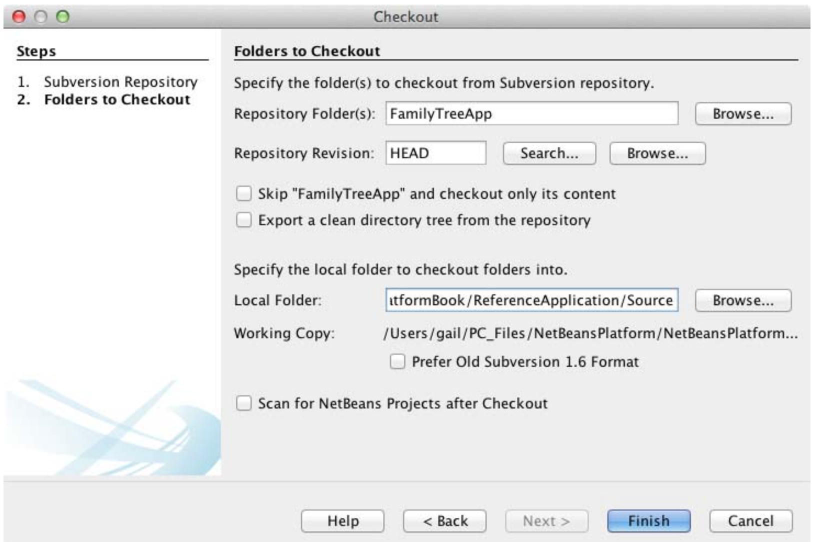

Figure 4. Specify the Repository Folders and the Local Folder

5. Subversion displays the results of the Repository Checkout in the Output window, as shown in Figure 5.

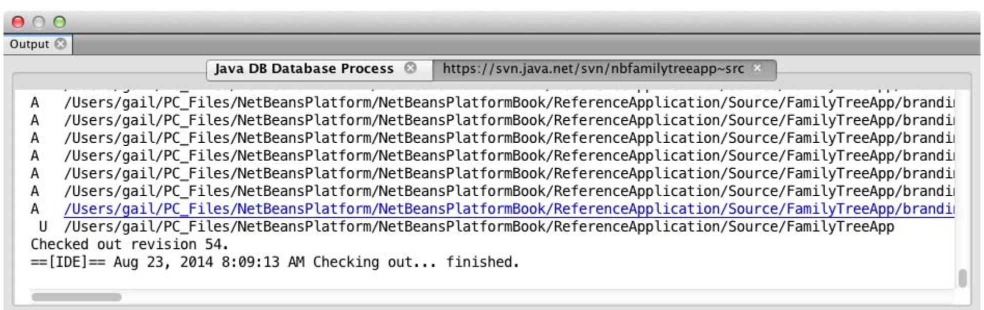

Figure 5. Subversion checkout results

6. Optionally, open the project's README file. In the Files window, expand the FamilyTreeApp node and double click the **README** file. The README file opens in the Editor window, as shown in Figure 6.

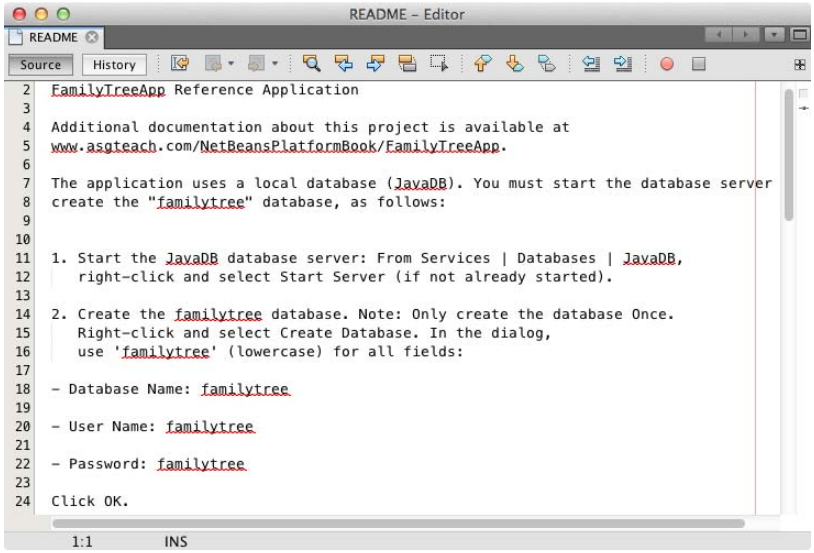

Figure 6. Access the README file from the Files window

2. Configuring the Database

The FamilyTreeApp reads and writes Person and Event data to the bundled database server, JavaDB. Here's how to start the database server and create the database.

1. In the NetBeans IDE Services window, expand Databases and select **JavaDB | Start Server**, as shown in Figure 7. The JavaDB database process will confirm its successful startup in the Output window.

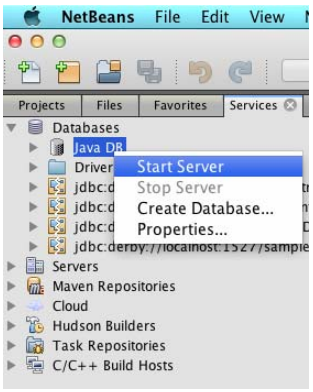

Figure 7. JavaDB | Start Server menu

2. After you start up the JavaDB server, you must create the database (this is a one-time step). In the Services window, select **JavaDB | Create Database . . .** , as shown in Figure 8.

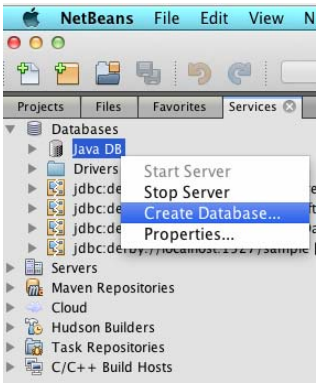

Figure 8. JavaDB | Create Database . . . menu

3. In the Create Java DB Database dialog, specify **familytree** for all fields (including the Password and Confirm Password fields), as shown in Figure 9. Click **OK**.

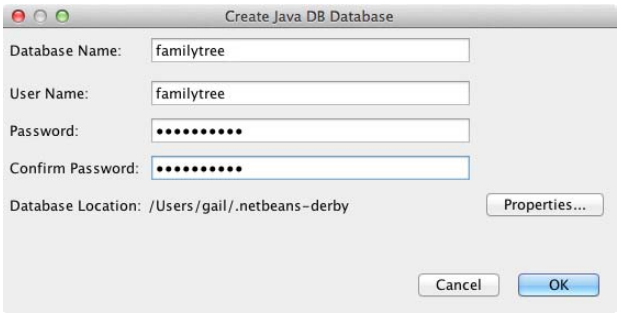

Figure 9. Create Java DB Database dialog

3. Run the FamilyTreeApp Application

With the JavaDB Database server started and the FamilyTree database created, you're now ready to run the application. Note that this application uses source level 1.8 and the JDK8 platform.

In the Projects window, select the FamilyTreeApp application, right click, and select Run. The startup process includes creating the database tables and the data. The application displays the Welcome Window, the Events and Person viewer windows, and the Person Timeline window, as shown in Figure 10.

All database access is performed in background threads to keep the UI responsive.

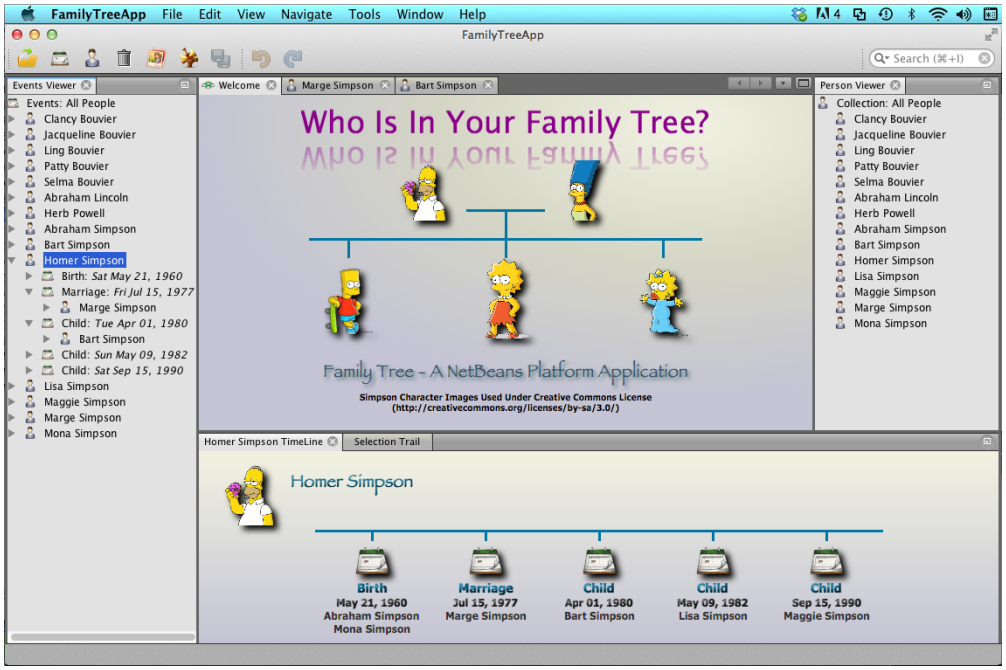

Figure 10. FamilyTreeApp running

4. Major Features

This section provides a top-level description of the FamilyTreeApp application, with an emphasis on the JavaFX integration. All of the techniques used in this application have been described in detail in the text book.

Welcome Window

The Welcome Window is a JavaFX-enabled TopComponent with two animations: a scale-up transition, as shown in Figure 11 and a fade in transition, as shown in Figure 12.

Figure 11. Welcome Window mouse entered scale up

Figure 12. Welcome Window mouse click fade in

JavaFX features used:

- TopComponent / JavaFX communication strategies as discussed in Chapter 6.
- Mouse click-initiated animations: parallel transition for fade out of all images, pause transition, followed by a sequential fade in of all images.
- Mouse entered / mouse exited scale transition: scale up image on mouse entered and subsequently scale down on mouse exited.
- Effects: radial gradient, drop shadows for images, reflection for text.
- Layout: BorderPane with embedded GridPane.

Events Viewer / Person Viewer

The Events Viewer window does not use JavaFX. This window uses the Nodes and Explorer Views APIs. Each Person can be expanded to show his or her list of life events. Each event, in turn, is expanded to show the life event relationships (child, parent, spouse, and so forth).

Nodes typically show a context menu of actions. Here, you see a Person has several actions, including Open (opens the Person in the Person Editor), Add Picture, New Event, and Delete.

The Person Viewer window is similar. Here, however, Person nodes are leaf nodes, so node expansion is not possible. Figure 13 shows both the Events Viewer and Person Viewer windows.

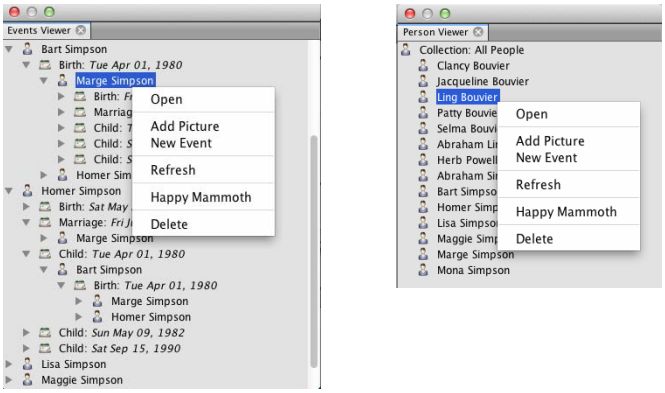

Figure 13. Events Viewer window and Person Viewer window

Person Timeline

The Person Timeline window is JavaFX content and displays the Person that is selected (either in the Events Viewer or Person Viewer window), as shown in Figure 14. When the user selects a different Person, the selected Person's life events are displayed in chronological order in a sequential fade in.

The window automatically updates if the selected Person's data changes.

Using JavaFX animation, the person's name, image, and life events all scale up with mouse over events. Figure 14 shows the first Child Event scaled up.

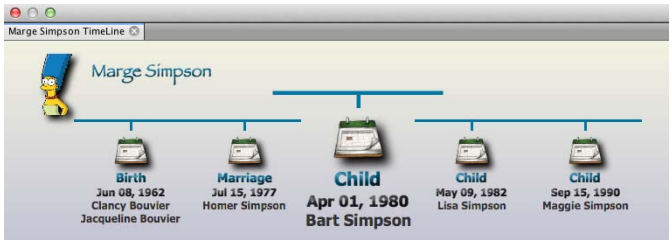

Figure 14. Person Timeline window

PersonFXEditor

When the user either double clicks on a Person or selects Open from the Person's context menu, the selected Person opens in the Person Editor, as shown in Figure 15. The Person Editor is a JavaFX-enabled form editor.

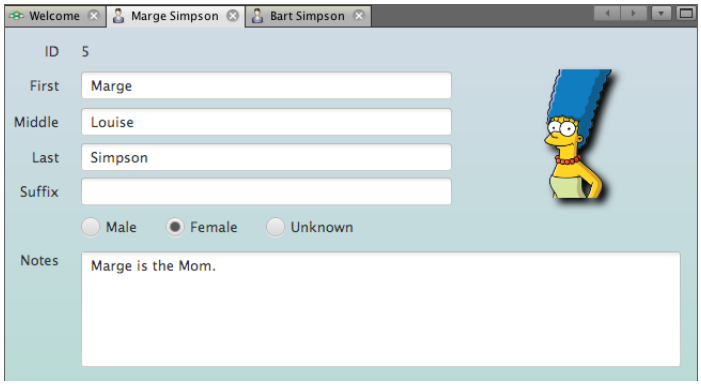

Figure 15. Person Editor

Module PersonFXEditor uses the communication strategies described in Chapter 6 to keep the JavaFX code as isolated as possible from TopComponent code. In addition, the module uses the NetBeans Platform Savable capability to interface with the Save and Save All menu items and toolbar icons. Updated Person data is propagated to the other application windows. PersonFXEditor is an example of a non-singleton TopComponent.

The PersonFXEditor was built using Scene Builder.

EventFXEditor

You can also edit an Event, as shown in Figure 16. Either double click an event or select Open from an event's context menu. The Event Editor is also a JavaFX-enabled form editor.

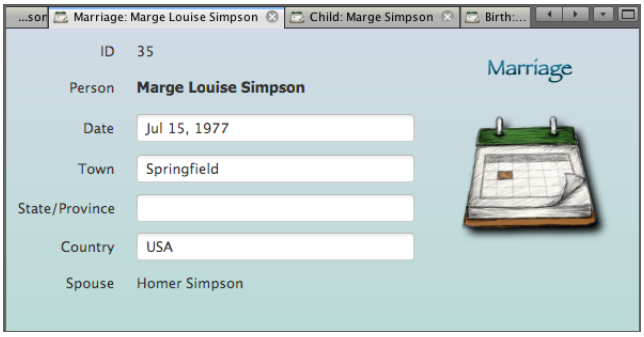

Figure 16. Event Editor

The Event Editor is structured exactly like the Person Editor, as follows.

- Uses the same communication strategies to keep modules cohesive and loosely coupled.
- Uses the NetBeans Platform Savable capability.
- Provides a non-singleton TopComponent.
- Changes to the event are propagated to other windows.

The EventFXEditor was built using Scene Builder.

New Event Wizard

Users can create a new event for a selected Person with the context menu item New Event. The New Event action is implemented with a NetBeans Platform wizard using JavaFX-enabled visual wizard panels. The New Event wizard has three steps, and the first step has asynchronous validation. Figure 17 shows the third and final step of the New Event wizard.

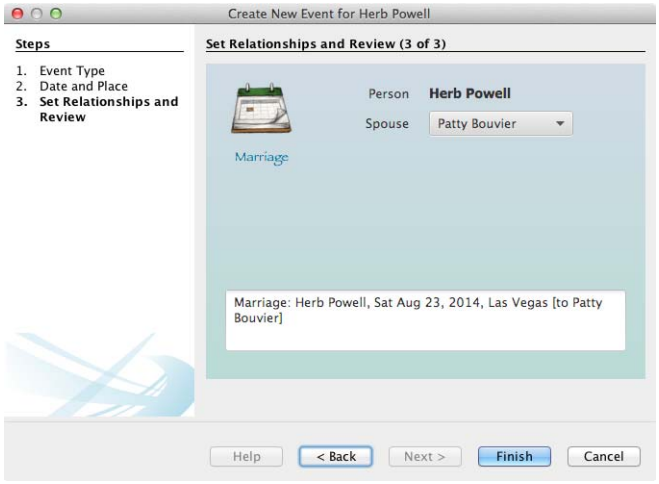

Figure 17. New Event Wizard