**Setting up the CelsiusConverter Project**

If you have worked with the NetBeans IDE in the past, much of this section will look familiar, since the initial steps are similar for most projects. Still, the following steps describe settings that are specific to this application, so take care to follow them closely.

**Step 1: Create a New Project**

To create a new project, launch the NetBeans IDE and choose New Project from the File menu:



Creating a New Project

Keyboard shortcuts for each command appear on the far right of each menu item. The look and feel of the NetBeans IDE may vary across platforms, but the functionality will remain the same.

**Step 2: Choose General -> Java Application**

Next, select General from the Categories column, and Java Application from the Projects column:



You may notice mention of "J2SE" in the description pane; that is the old name for what is now known as the "Java SE" platform. Press the button labeled "Next" to proceed.

**Step 3: Set a Project Name**

Now enter "CelsiusConverterProject" as the project name. You can leave the Project Location and Project Folder fields set to their default values, or click the Browse button to choose an alternate location on your system.



Make sure to deselect the "Create Main Class" checkbox; leaving this option selected generates a new class as the main entry point for the application, but our main GUI window (created in the next step) will serve that purpose, so checking this box is not necessary. Click the "Finish" button when you are done.



When the IDE finishes loading, you will see a screen similar to the above. All panes will be empty except for the Projects pane in the upper left hand corner, which shows the newly created project.

**Step 4: Add a JFrame Form**



Now right-click the CelsiusConverterProject name and choose New -> JFrame Form (JFrame is the Swing class responsible for the main frame for your application.) You will learn how to designate this class as the application's entry point later in this lesson.

**Step 5: Name the GUI Class**

Next, type CelsiusConverterGUI as the class name, and learn as the package name. You can actually name this package anything you want, but here we are following the tutorial convention of naming the package after the lesson in which is resides.



The remainder of the fields should automatically be filled in, as shown above. Click the Finish button when you are done.



When the IDE finishes loading, the right pane will display a design-time, graphical view of the CelsiusConverterGUI. It is on this screen that you will visually drag, drop, and manipulate the various Swing components.

**NetBeans IDE Basics**

It is not necessary to learn every feature of the NetBeans IDE before exploring its GUI creation capabilities. In fact, the only features that you really need to understand are the *Palette*, the *Design Area*, the *Property Editor*, and the *Inspector*. We will discuss these features below.

**The Palette**

The Palette contains all of the components offered by the Swing API. You can probably already guess what many of these components are for, even if this is your first time using them (JLabel is a text label, JList is a drop-down list, etc.)



From this list, our application will use only JLabel (a basic text label), JTextField (for the user to enter the temperature), and JButton (to convert the temperature from Celsius to Fahrenheit.)

**The Design Area**

The Design Area is where you will visually construct your GUI. It has two views: *source view*, and *design view*. Design view is the default, as shown below. You can toggle between views at any time by clicking their respective tabs.



The figure above shows a single JFrame object, as represented by the large shaded rectangle with blue border. Commonly expected behavior (such as quitting when the user clicks the "close" button) is auto-generated by the IDE and appears in the source view between uneditable blue sections of code known as *guarded blocks*.



A quick look at the source view reveals that the IDE has created a private method named initComponents, which initializes the various components of the GUI. It also tells the application to "exit on close", performs some layout-specific tasks, then packs the (soon to be added) components together on screen.

**The Property Editor**

The Property Editor does what its name implies: it allows you to edit the properties of each component. The Property Editor is intuitive to use; in it you will see a series of rows — one row per property — that you can click and edit without entering the source code directly. The following figure shows the Property Editor for the newly added JFrame object:



The screenshot above shows the various properties of this object, such as background color, foreground color, font, and cursor.

**The Inspector**

The last component of the NetBeans IDE that we will use in this lesson is the Inspector:



The Inspector

The Inspector provides a graphical representation of your application's components. We will use the Inspector only once, to change a few variable names to something other than their defaults.

**Creating the CelsiusConverter GUI**

This section explains how to use the NetBeans IDE to create the application's GUI. As you drag each component from the Palette to the Design Area, the IDE auto-generates the appropriate source code.

**Step 1: Set the Title**

First, set the title of the application's JFrame to "Celsius Converter", by single-clicking the JFrame in the Inspector:



Selecting the JFrame

Then, set its title with the Property Editor:



Setting the Title

You can set the title by either double-clicking the title property and entering the new text directly, or by clicking the button and entering the title in the provided field. Or, as a shortcut, you could single-click the JFrame of the inspector and enter its new text directly without using the property editor.

**Step 2: Add a JTextField**

Next, drag a JTextField from the Palette to the upper left corner of the Design Area. As you approach the upper left corner, the GUI builder provides visual cues (dashed lines) that suggest the appropriate spacing. Using these cues as a guide, drop a JTextField into the upper left hand corner of the window as shown below:



Adding a JTextField

You may be tempted to erase the default text "JTextField1", but just leave it in place for now. We will replace it later in this lesson as we make the final adjustments to each component.

**Step 3: Add a JLabel**

Next, drag a JLabel onto the Design Area. Place it to the right of the JTextField, again watching for visual cues that suggest an appropriate amount of spacing. Make sure that text base for this component is aligned with that of the JTextField. The visual cues provided by the IDE should make this easy to determine.



Adding a JLabel

**Step 4: Add a JButton**

Next, drag a JButton from the Palette and position it to the left and underneath the JTextField. Again, the visual cues help guide it into place.



Adding a JButton

You may be tempted to manually adjust the width of the JButton and JTextField, but just leave them as they are for now. You will learn how to correctly adjust these components later in this lesson.

**Step 5: Add a Second JLabel**



Adding a Second JLabel

Finally, add a second JLabel, repeating the process in step 2. Place this second label to the right of the JButton, as shown above.

**Adjusting the CelsiusConverter GUI**

With the GUI components now in place, it is time to make the final adjustments. There are a few different ways to do this; the order suggested here is just one possible approach.

**Step 1: Set the Component Text**

First, double-click the JTextField and JButton to change the default text that was inserted by the IDE. When you erase the text from the JTextField, it will shrink in size as shown below. Change the text of the JButton from "JButton1" to "Convert." Also change the top JLabel text to "Celsius" and the bottom to "Fahrenheit."



Setting the Component Text

**Step 2: Set the Component Size**

Next, shift-click the JTextField and JButton components. This will highlight each showing that they are selected. Right-click (control-click for mac users) Same Size -> Same Width. The components will now be the same width, as shown below. When you perform this step, make sure that JFrame itself is not also selected. If it is, the Same Size menu will not be active.



Setting the JTextField and JButton Sizes

**Step 3: Remove Extra Space**

Finally, grab the lower right-hand corner of the JFrame and adjust its size to eliminate any extra whitespace. Note that if you eliminate all of the extra space (as shown below) the title (which only appears at runtime) may not show completely. The end-user is free to resize the application as desired, but you may want to leave some extra space on the right side to make sure that everything fits correctly. Experiment, and use the screenshot of the finished GUI as a guide.



The Completed GUI

The GUI portion of this application is now complete! If the NetBeans IDE has done its job, you should feel that creating this GUI was a simple, if not trivial, task. But take a minute to click on the source tab; you might be surprised at the amount of code that has been generated.



To see the code in its entirety, scroll up and down within the IDE as necessary. You can expand or collapse certain blocks of code (such as method bodies) by clicking the + or - symbol on the left-hand side of the source editor.