Chapter 4: Check Boxes

Check Boxes

The next region we'll do is the center region, with the toppings check boxes in it. It's similar to the west region and even a little simpler, so it should go quicker.

Once again, we'll add a new method to build it, this time named *makeCenterRegion()*. The start of the method is similar to our makeWestRegion() method: It creates a panel, gives it a vertical BoxLayout, and creates a border (this time with the title "Select Toppings"). We don't need a button group since check boxes don't have any effect on each other—depending on their preferences, users can select all, some, or none of them.

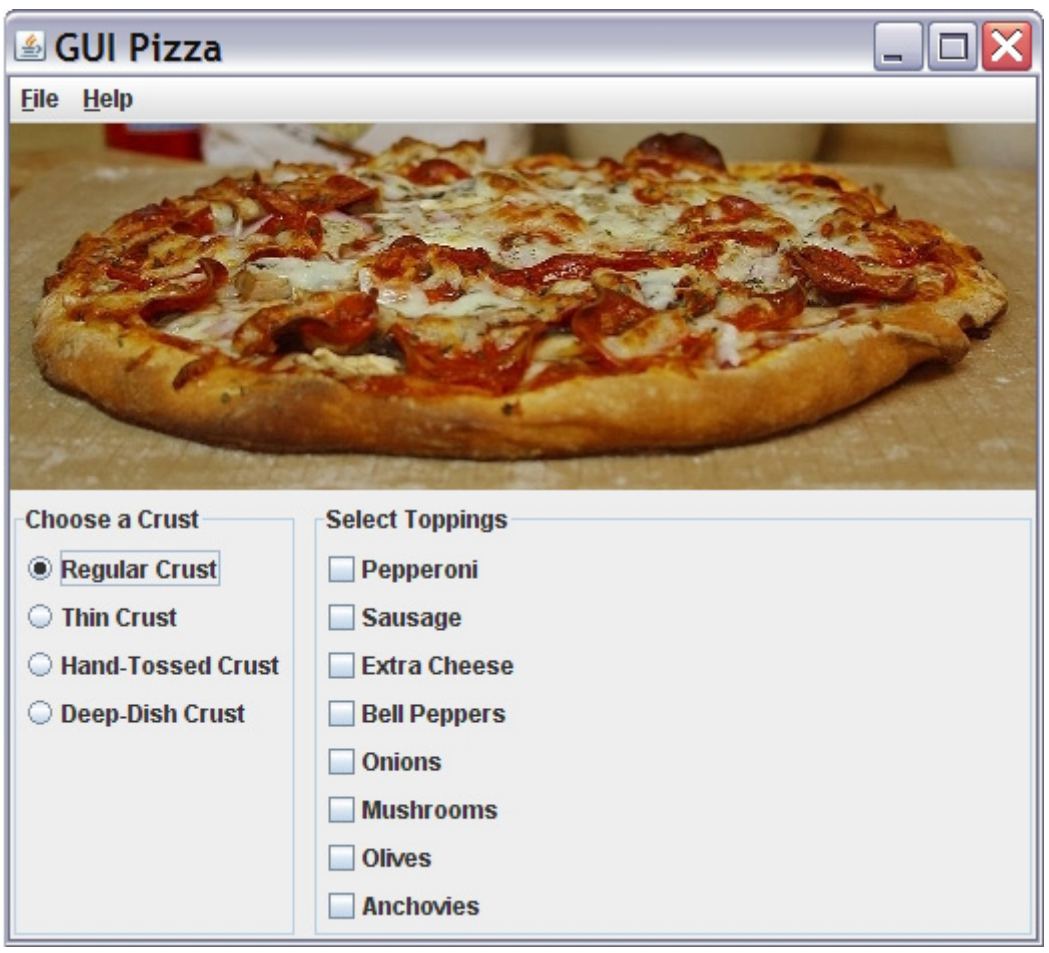
The following lines create one of our check boxes and add it to the panel:

```
pepperoniBox = new JCheckBox("Pepperoni", false);

panel.add(pepperoniBox);
```

There are seven more check boxes, and I'll let you add them yourself for practice. And don't forget to add the panel to the center region of the content pane before you end the method. If you want to check your code against mine, you can click here.

Solution: Try not to peek!

Here's what our window looks like after adding the center region:



Window with check boxes

You can see that the west region's height expanded to match the center's, and that the center region's width adjusted to match the north region's width. Next, we'll add the east region, which has a couple of text entry fields.

Text Fields

Looking back at our finished window, we see that the east region has two text fields. Text fields are GUI components that allow a user to enter text. The two in this region let users enter the number of side orders they want. Each text field has a label next to it so we know what a number in that field means. Unlike radio buttons and check boxes, text fields don't have automatic text labels associated with them, so we'll have to generate the text fields and labels separately and then put them together on a line.

Let's go ahead and add our makeEastRegion() method and its call in makeContent() so we have a starting point. We'll also set up the panel, its layout (BoxLayout with a vertical axis again), and its border. Our method looks like this:

```
private void makeEastRegion()

{

    // set up side orders with quantities in EAST

    JPanel panel = new JPanel();

    panel.setLayout(new BoxLayout(panel,BoxLayout.Y_AXIS));

    panel.setBorder(BorderFactory.createTitledBorder("Sides (Enter Quantity)"));

}
```

Here's where things get a little more interesting. We want each line of our panel in this method to hold a text field and a label side by side. To do that, we have to introduce one more layer—another panel with its own layout. Let's create one more panel named *smallPanel*, and we'll give it a BoxLayout with a horizontal axis. Here's how to do that:

```
JPanel smallPanel = new JPanel();

smallPanel.setLayout(new BoxLayout(smallPanel,BoxLayout.X_AXIS));
```

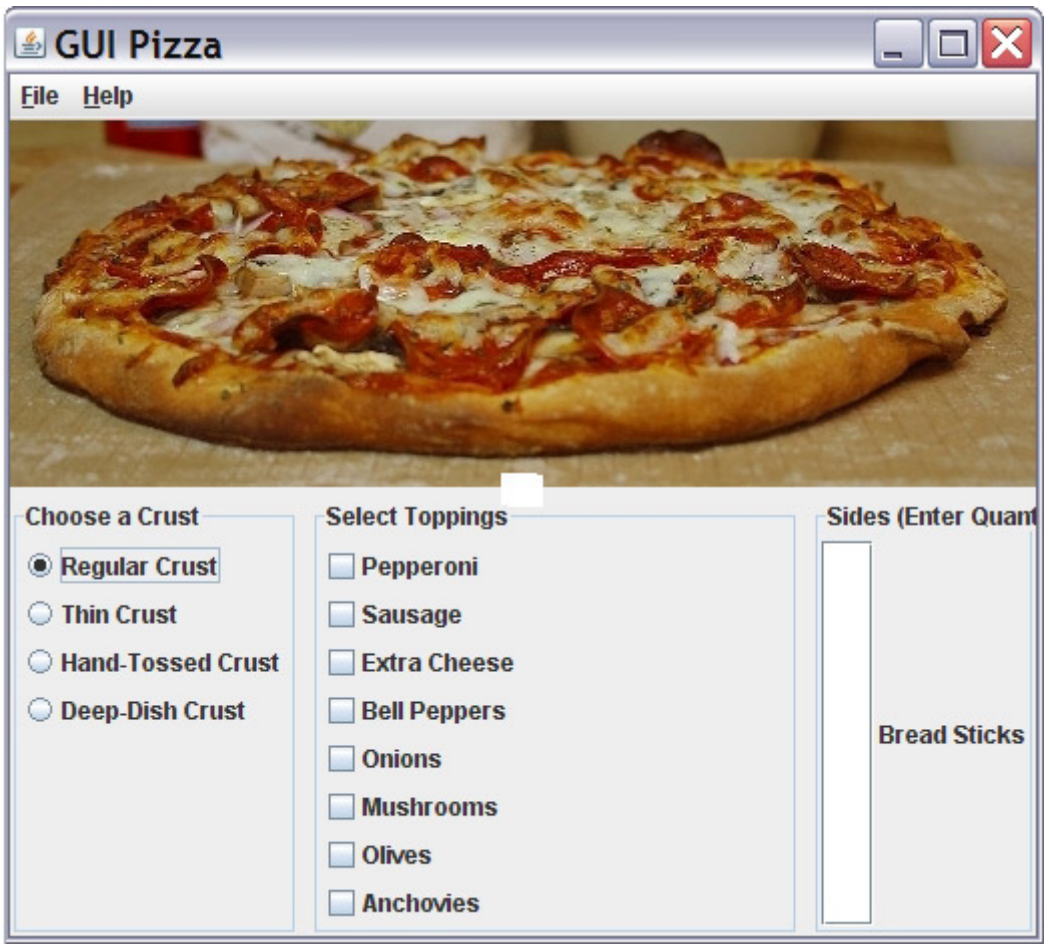
Now we're ready to create our text field and label. After that, we'll do three things: add the text field and label to the small panel, add the small panel to the larger panel, and add the larger panel to the content pane.

Here's the code that will do that for the first line:

```
breadSticksText = new JTextField("",2);

smallPanel.add(breadSticksText);

smallPanel.add(new JLabel(" Bread Sticks"));

panel.add(smallPanel);

contentPane.add(panel, BorderLayout.EAST);
```

The first line above creates a new text field, a JTextField object, with no text in it and a minimum width of two columns. This is where customers will enter how many orders of bread sticks they want. The second line adds the text field to our smallPanel. The third line creates a label to go alongside the text field, and it adds the label to smallPanel. The fourth line adds smallPanel to panel, and the last line adds panel to contentPane.

If you add this code to our program so far and run it, the east region of our window will look rather strange:



Window with a tall text box

There are a couple of problems here that we're going to fix. The first is obvious: The text box is too tall. By default, text boxes will fill as much space as you give them. We set the text box width to two columns, but we didn't set its height, so it automatically became as tall as our other two regions.

The second problem is not quite so obvious: The panel isn't wide enough for our border title. Remember, the east and west regions have their widths set by the widths of the items in them. And so far, our items are not wide enough for our title.

Let's fix the width of our panel first. We can set a panel (and many other components) to a preferred size that some layout managers will honor. Here's how we'll set ours:

```
panel.setPreferredSize(new Dimension(150,0));
```

This statement can go just about anywhere in our method. Let's put it on the line after we set the border—the fourth line of the method. That keeps the panel's setup statements together. The method we called, setPreferredSize(), uses one argument, and it has to be a *Dimension* object. Dimension is a Java class designed to hold two values: another object's width and height. We set our panel's width to 150 pixels, which is wide enough to display our title. We set its height to 0, since as long as it is less than the height of our center region, it won't make a difference. Remember, the tallest region of a BorderLayout's east, center, and west regions controls the height of all three.

Now let's fix the height of our text field. Just like we set the panel's preferred size, we can set a component's maximum size:

```
breadSticksText.setMaximumSize(new Dimension(20,24));
```

We set the size of the text field to a maximum width of 20 pixels and a maximum height of 24 pixels, which is about the size we want. If you add that line and run the program, you'll see one more little problem. Our small panel is centered horizontally in our larger panel, but we want it to be at the left side of the panel. We can fix that with one more adjustment:

```
smallPanel.setAlignmentX(Component.LEFT_ALIGNMENT);
```

That finally takes care of our first line in the east region. I told you it would be interesting!

Go ahead and add the second item to the panel by using the first as a model. If you have trouble with it, you can look at my version of the method here.

Solution: Try not to peek!

If you put everything together and run the program now, you should see this window:

Window with text fields