Flex in a Week, Flex 4.5
Video 4.09: Using constraints to control component layout

In the last video, you learned how to layout and size child components using absolute x and y positions as well as relative percentage settings on both the parent and child components.

In this video, you will learn how to implement constraint-based layouts, where child components lay themselves out relative to pixel values set on anchor positions in the parent container.

To position and size components using constraint-based layout, you must place the components inside of a Spark container with the layout property set to an instance of the BasicLayout class.

You can also used constraint-based layout inside of MX containers, but, since the equivalent containers exist in Spark, you should use the Spark containers whenever possible.

Previously with the BasicLayout class, you have specified x and y properties for components in order to position them absolutely in the display.

For constraint-based layouts, you do not use x and y properties.

Instead, you use baseline, top, bottom, left, right, horizontalCenter and verticalCenter.

Each property takes a pixel value which determines the distance that the component will be positioned relative to the specified constraint for the parent container.

Before I get back to the Employee Portal example that I was working with in the last video, let me illustrate the implementation of constraint-based layouts using Design mode.

Here is an MXML application without any visual controls.

Note, however, that it does have a layout property set to BasicLayout.

I am removing the minWidth and minHeight properties from the Application container and then switching to Design mode and dragging a Panel container to the Design area.

When I save the application and run it, you can see that the Panel container looks the same as it did in Design mode.

Back in Design mode, I am scrolling down in the Properties view to the Size and Position section.

These checkboxes represent the constraint properties that I outlined earlier.

I am clicking on the left constraint and setting a value of 50 pixels.

I'm doing the same for the right, top and bottom constraints.

When I save the file and run the application, you can see that the panel is initially huge.

As I resize the browser, the Panel always stays 50 pixels from each of the four edges.

Back in Flash Builder, I am switching to Source mode to show you the left, right, top and bottom constraints set to 50 pixels.

I am removing those properties and replacing them with a horizontalCenter property set to 0 and a verticalCenter property set to 0.

When I save the file and run the application, the Panel returns to its default size, but it is positioned exactly zero pixels from the horizontal and vertical center of the application.

This is the Employee Portal application.

When I resize the browser, you can see that the application is fixed in size.

I am returning to Flash Builder and switching to Design mode.

In the States view, I am selecting the portalState and the Monthly Events Panel , which is the last panel in the display.

In the Properties view's Size and Position segment, I am checking the top left box and setting its value to 574. This will set the panel 574 pixels from the left edge of the browser.

Now I am setting the right constraint by checking the top right box and setting its value to 24. This sets the panel 24 pixels from the right edge of the browser.

I am saving the file and running the application.

When I enlarge the browser window, the Monthly Events panel expands to the right with the browser, but there is always a 24-pixel margin between the panel and the browser.

Notice that the text within the Monthly Events panel does not resize when I resize the browser.

I am returning to Flash Builder and selecting the Label control within the Monthly Events panel and, in the Properties view, I am changing the Width and Height properties to 100%.

I am save the file and running the application.

When I resize the browser, the Monthly Events panel and its contents resize as well.

Next I want to apply vertical constraints so that the Monthly Events Panel container will fill more of the vertical space and maintain that relative growth as the browser resizes.

In the Properties view, I am checking the box for the bottom constraint and setting its value to 15. This constrains the panel to 15 pixels from the bottom edge of the browser.

Now I am checking the top box on the left to set the top constraint 95 pixels from the top edge of the browser.

I am saving the file and running the application.

When I expand the browser, you can see that there is a 15-pixel margin between the bottom of the panel and the bottom of the browser and a 95-pixel margin between the top of the panel and the top of the browser.

Now I will set constraints for the other panels in the application.

In Design mode, I am selecting the Cafeteria Special panel and setting the top property to a value of 95 and the bottom property to a value of 15.

Next, I am locating the Employee Directory Panel container and adding the top property with a value of 410 and the bottom property with a value of 15.

I am saving the file and running the application.

When I resize the browser, notice that the panels expand and maintain the 15-pixel bottom border, but they also collapse to the point that their content slips outside the content area.

Next I will add the minHeight property to the Employee Directory panel.

Back in Flash Builder, I am selecting the Employee Directory Panel container and switching to Source mode to add the minHeight property with a value 160.

I am saving the file and running the application.

Now when I resize the browser, the Employee Directory panel maintains a 15-pixel border below it until it reaches the minHeight value of 160 pixels

The Cafeteria Special and Monthly Events panels continue to collapse.

Instead of adding the minHeight property to each of the Panel containers, I will add the property to the application.

I am returning to Flash Builder.

I am removing the minHeight property from the Employee Directory Panel.

To the opening Application tag, I am adding the minHeight property and setting its value to 575.

I am saving the file and running the application.

Now when I resize the browser, none of the panels collapse past the value set for the minHeight property on the Application.

Notice that all of the panels expand and collapse when the browser resizes, but the Employee Portal header bar and the Logout button do not.

I am returning to Flash Builder and switching to Design mode.

I am selecting the Employee Portal header bar.

In the Properties view, I am setting the right constraint to a value of 24 and the left constraint to a value of 24.

I am saving the file and running the application.

When I resize the browser, there is now a 24-pixel margin between the left and right of the header and the browser.

Notice that the Logout button does not expand with the title bar.

Back in Flash Builder, I am selecting the Logout Button control.

I am setting the right constraint to a value of 44, saving the file and running the application.

When I resize the browser, the Logout button now expands with the title bar.

Notice that when I reduce the width of the browser, the Employee Portal Label control collapses, while the other Panel containers do not.

I am returning the Flash Builder and switching to Source mode.

To the opening Application tag, I am adding the minWidth property and setting its value to 800.

By adding the minWidth property, I am making sure that when I reduce the browser size, all of the components in the application will stop collapsing when the browser reaches a width of 800-pixels.

I am saving the file and running the application.

Now when I reduce the width of the browser, all of the components stop resizing at a width of 800 pixels.

For your next step, work through the exercise titled “Creating a scalable user interface”.