Intro to Auto Layout

Hands On Challenges

Introduction to Auto Layout Hands-On Challenges

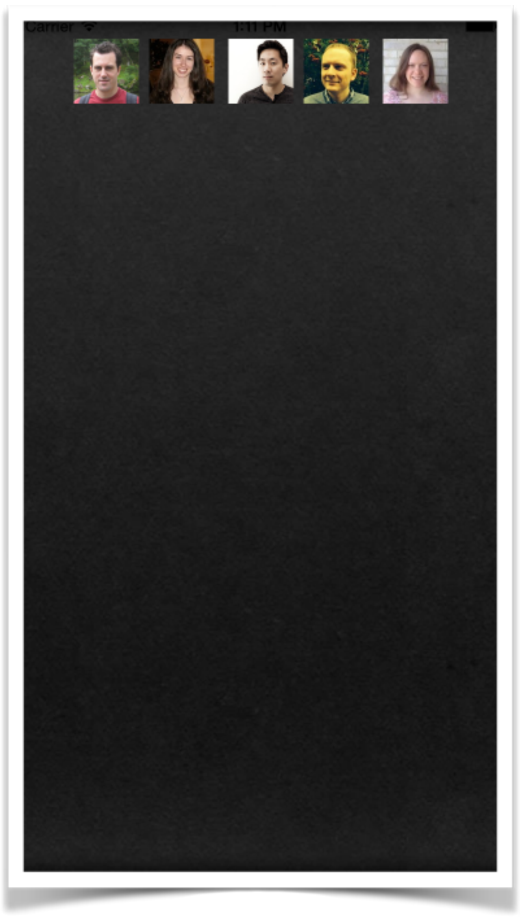
Challenge C: Working with Interface Builder

Often times when building your constraints, you will need to alter them in some way. For example, you may want to increase the amount of distance from the leading space of your control to the left hand side of the margin.

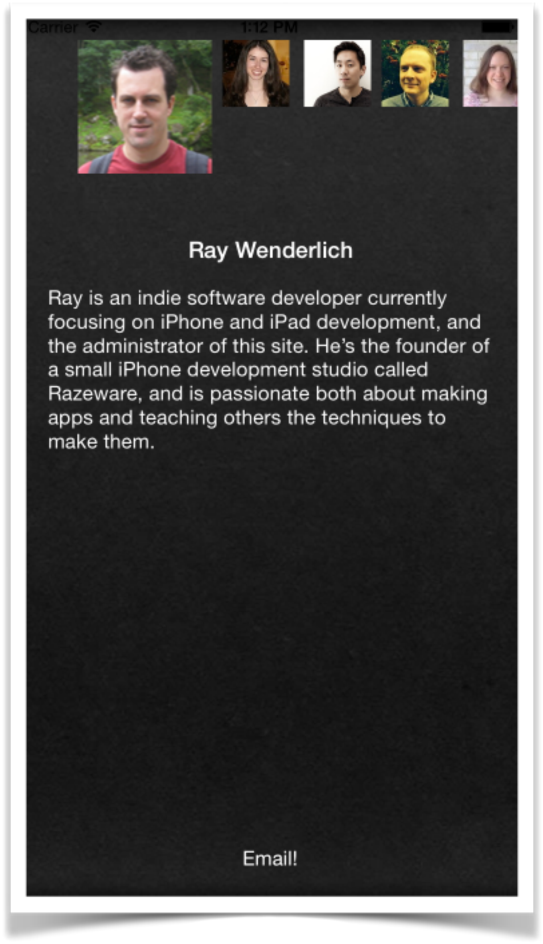
If you are setting your constraints in Interface Builder, in order to change it at runtime, you need to create an outlet to that constraint. Once you have the outlet in place, you can then change the constraint in code.

The key thing to remember – you can only change the **constant** value of the **constraint**.

Here's what you'll build in this challenge:



When the user taps on each of the images, they should see the following:



Ray Wenderlich

Ray is an indie software developer currently focusing on iPhone and iPad development, and the administrator of this site. He's the founder of a small iPhone development studio called Razeware, and is passionate both about making apps and teaching others the techniques to make them.

Also notice, an email button appears on the bottom of the view controller.

Challenge Hint

In order for the pictures to expand and decrease in size, you will need to change the width and height on each of the imageviews. To simplify, only set one dimension (width or height) with a constraint and then use an aspect ratio constraint to control the other dimension. Create an outlet for the constraint and modify the constant when the user taps the image. You’ll need some logic to determine whether the image is large or small.

Challenge Solution

As always, the very first thing that you should do is layout your app.

1) Drag an empty imageview onto the scene. Make sure it fills up the complete view.

2) In the imageview, set the image to be **footer-background-min**.

3) With the imageview selected, press the pin button. Uncheck **constrain to margins** and pin the imageview on all four sides. Add all four constraints.

4) Drag an empty UIView into the scene. In the size inspector, set the following values: **x: 159, y:20, width: 282, height: 100**. In the attributes inspector, set the **Background color** to **clear**.

5) Drag an imageview into the view. In the size inspector, set the following values: **x: 0, y:0, width: 50: height: 50**. In the attributes inspector, set the image to be **ray** and check the **User Interaction Enabled** checkbox.

6) Duplicate the image by pressing **command-D**. In the size inspector, set the following values: **x: 58, y: 0**. In the attributes inspector, set the image to be **vicki** and the tag to **1**.

7) Duplicate the image by pressing **command-D**. In the size inspector, set the following values: **x: 116, y: 0**. In the attributes inspector, set the image to be **greg** and the tag to **2**.

8) Duplicate the image by pressing **command-D**. In the size inspector, set the following values: **x: 174, y: 0**. In the attributes inspector, set the image to be **mic** and the tag to **3**.

9) Duplicate the image by pressing **command-D**. In the size inspector, set the following values: **x: 232, y: 0**. In the attributes inspector, set the image to be **christine** and the tag to **4**.

10) Add a label underneath the view containing all the pictures. In the size inspector, add the following values: : **x: 278, y: 167, width: 45: height: 21**. In the attributes inspector, change the **Font** to **System Bold 17**. Next, set the **text color** to **white**.

11) Add a text view underneath the label. In the size inspector, add the following values: : **x: 20, y: 196, width: 560: height: 358**. In the attributes inspector, clear the default text and set the **text color** to **white**.

12) Add a button at the bottom of the view controller. Give it the name: **Email!**. In the size inspector, add the following values: : **x: 280, y: 562, width: 41: height: 30**. In the attributes inspector, set the **text color** to **white**.

13) Select the **view** containing all the pictures, then **click** the **pin** button. Select the **top I-beam**, then check the **height constraint**. Add the constraints.

14) With the view still selected, click the alignment button and check **Horizontally in Container**. Add the constraint.

15) Select all the photos, then click the pin button. Select **the left, top, and right I-beams**. Also, make sure to click the **width** and **aspect ratio** constraints. Add all the constraints.

Pro-tip: You could add width and height constraints, but then when you want to change the size of the image, you'd have to change both constraints. By adding just width and aspect ratio constraints, you can now change only the width and the height will automatically adjust to keep the aspect ratio.

16) Select the label, and click the pin button. Select the **top and bottom I-beams**. Next, with the label still selected, click on the alignment button and check **Horizontally in Container**. Add the constraint.

17) Select the textview. Click the pin button. Select the **left, right, and bottom I-beams**. Add all the constraints.

18) Finally, select the email button. Click the pin button. Select the **bottom I-beam**. Next, select the alignment button, and check **Horizontally in Container**.

19) Now, you'll have to create an outlet collection for the width constraints. Select the first constraint (on the Ray image), and with the assistant editor open, control drag to the source code. Change the **Connection** to **Outlet Collection** and name it widthConstraints. In the order the images appear, control drag the remaining width constraints to the same outlet.

20) Make sure to create outlets for your label, text view, and button. At this point, you should compare the actual coding portion of the project to the completed project. Once you've created the code, create tap gesture recognizers for each image and attach to the imageTapped(sender:) IBAction.