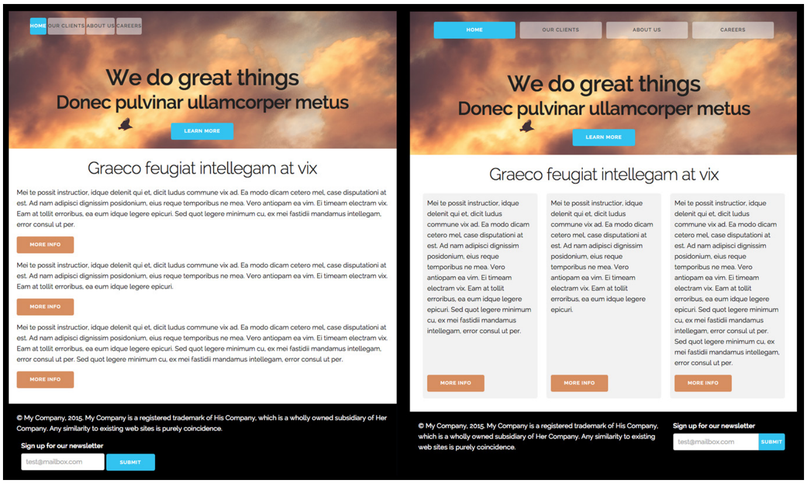
Just a note about the vendor prefixes, you can just create the first style property, for instance, display:flex; and then you can run your code through the Autoprefixer CSS online tool and it will parse your CSS and add the additional vendor prefixes. Please make sure you run your CSS code through the Autoprefixer before you submit your work.
http://autoprefixer.github.io/
Also, you need to use the correct HTML5 semantic markup for this file.
https://www.w3schools.com/html/html5_semantic_elements.asp

Tutorial: Build a Flexbox Layout

In this tutorial, you'll take a linear layout that stacks one div on top of another (Figure 17-17, left) and create columns using flexbox (Figure 17-17, right). In addition, you'll create a few visual effects that are extremely easy with flexbox but nearly impos-sible with other CSS techniques.



We do great things
Donec pulvinar ullamcorper metus

Graeco feugiat intellegam at vix

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

FIGURE 17-17

You can create multi-column layouts without using any floats with flexbox.

Styling the Navigation Bar

This design consists of three flex containers—one for the navigation button, one for the main content area, and one for the footer (Figure 17-18). Each container will hold a variable number of flex items: four buttons in the navigation bar, three content boxes in the main area, and two regions in the footer. The first step is create those flex containers.

1. **In your text editor, open the file *17→flexbox-layout→css→custom.css*.**

   This is an empty CSS file, to which you'll add the CSS for flexbox. The *index.html* page has two style sheets linked to it: this empty file and a file named *base.css*, which contains a lot of styling information to make the basic page elements look good.

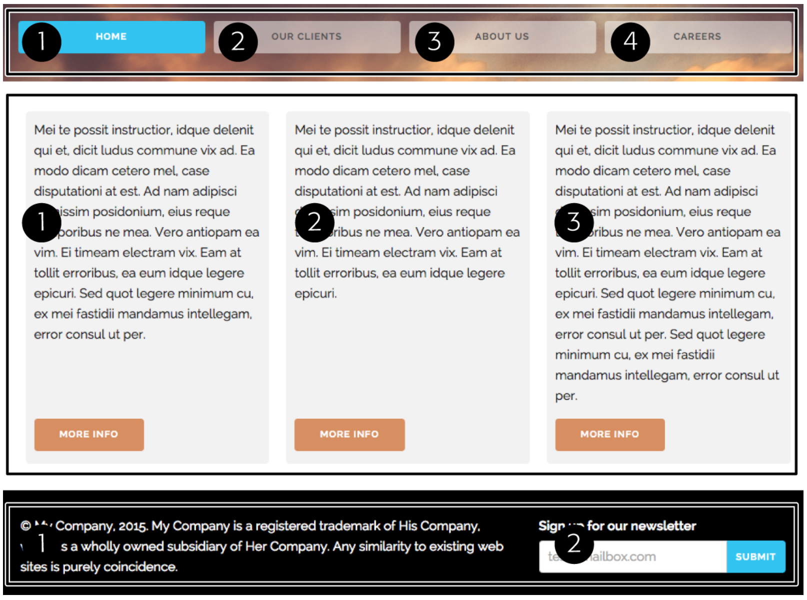
   First, you'll set up the main flex containers.

2. **In the *custom.css* file add a group selector rule:**

```
nav, .boxes, footer {
  display: flex;
 flex-flow: row wrap;
}
```

   Each of those class names corresponds to a class that is applied to the three regions outlined in Figure 17-18. This basic rule turns those divs into flex containers and sets them up to display as a row of flex items that can wrap into additional rows. The `-webkit` vendor prefix is included so these styles will work in Safari as well as Chrome, Firefox, Opera, and Internet Explorer 11.

Next, you'll do something that's usually difficult to achieve. You'll spread out the navigation buttons evenly so that the leftmost button is on the left side of its container, the rightmost button touches the right side of the container, and the other two buttons are distributed evenly between.



Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci issim posidonium, eius reque poribus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci sim posidonium, eius reque ribus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci sim posidonium, eius reque ribus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

© Company, 2015. My Company is a registered trademark of His Company, s a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign for our newsletter

te ailbox.com

FIGURE 17-18

The design for this page includes three flex containers (the top, middle, and bottom rows), which are divided into separate flex items (numbered elements).

3. **Add another style following the one you added in the last step:**

```
nav {
  justify-content: space-between;
}
```

The `justify-content` property applied to the flex container controls how the flex items are placed inside the container. In this case, the `space-between` value distributes all the flex items inside it evenly across the container. This is really cool, and usually difficult to do using other CSS techniques.

You can also give those buttons flexible widths by giving them percentage width values, so try that next.

4. **Style the links by adding one more style to the style sheet:**

```
nav a {
  width: 23%;
}
```

This style makes each link take up 23% of the flex container's width. As the container shrinks, the buttons get narrower. The remaining space is evenly distributed between each button, thanks to the style you added in the previous step.

There you have it. A navigation bar made up of evenly distributed, variable-width buttons, and all done without floats or other difficult CSS.

Adding Three Columns

Now it's time to tackle the main content area. Currently, there are three divs, each containing a More Info button, stacked on top of each other. By adding a `flex` property, you can place them side by side. Remember that in step 1 on page 559, you turned the div that holds these three divs into a flex container. Now you'll turn the divs into flex items.

1. **At the bottom of the custom.css file add the following rule:**

```
.boxes div {
  flex: 1 1 250px;

 }
```

This makes each of the divs take up an equal area of their flex container. They're each the same width, but currently they touch each other, and it's a bit hard to read that way (and the fact that the text is in Latin doesn't help either). The design should now look like the top image in Figure 17-19. To make the columns easier to read, you'll add a background color and a bit of space.

2. **Edit the style you added in the last step by adding four more properties to it (additions are in bold):**

```
.boxes div {
  flex: 1 1 250px;
  margin: 10px;
  border-radius: 5px;
  padding: 10px 10px 0 10px;
  background-color: rgba(0,0,0,.1);
}
```

The columns should now look like the middle image in Figure 17-19. Notice how each of the columns are the same height, even though the content inside them is of different lengths. This is another magic flexbox moment: By default, columns in a row are all the same height. That's really hard to do with regular CSS, and some designers even resort to using JavaScript to get this effect.

However, the buttons, which are so prominent in the design, are staggered at different vertical positions, which is distracting. It would be great if there were a way to force those buttons to the bottom of each column. Thanks to flexbox, there is!

Margins work slightly differently with flex items. For example, you can add `auto` margin to the top of each button to force it down to the bottom of its column. However, you can only use this margin magic on flex items. At this point, only the columns themselves are flex items; the buttons and paragraphs inside are regular elements.

Fortunately, the flexbox model lets you assign more than one role to a flex item. The columns can be flex items within their outer container (the row), *and* they can be flex containers holding flex items (the paragraph and button) inside them.

3. **Add four more lines of code to the `.boxes div` style:**

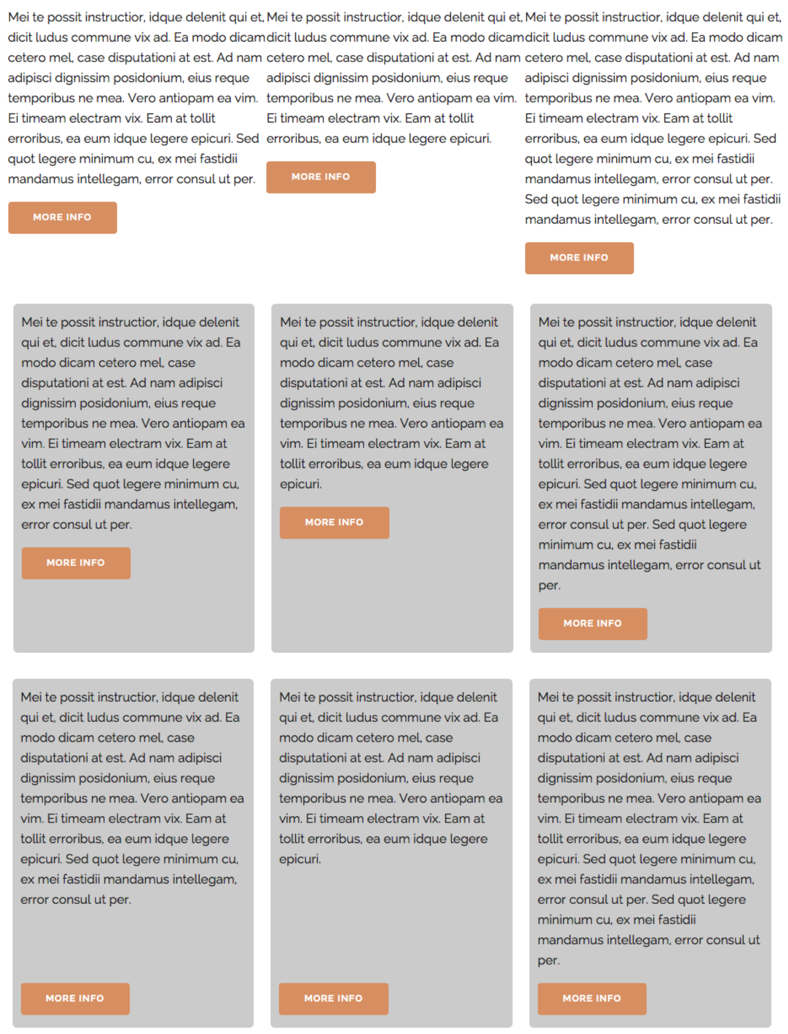
```
.boxes div {
  -webkit-flex: 1 1 250px;
  flex: 1 1 250px;
  margin: 10px;
  border-radius: 5px;
  padding: 10px 10px 0 10px;
  background-color: rgba(0,0,0,.2);
  display: -webkit-flex;
  display: flex;
  -webkit-flex-flow: column;
  flex-flow: column;
}
```

This defines the divs as flex containers, and it sets the flow direction of the flex items inside to `column`, which stacks the paragraph and buttons on top of each other. You may be saying to yourself, "Hey self, they're already stacked on top of each other, why do I need to do this?" Flex items can take advantage of auto margins, so you can force those buttons to the bottom of each div.

4. **Add another style after the `.boxes div` rule:**

```
.boxes .more {
  margin-top: auto;
}
```

Each button has the class of `more` applied to it, so this selector selects the buttons inside the columns. The top margin is set to `auto`, which tells the browser to automatically add whatever empty space is left inside the flex item above the top of the button. This auto-margin has the effect of pushing all of the buttons to the bottom of each column (Figure 17-19, bottom).



Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

MORE INFO

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

MORE INFO

FIGURE 17-19

Flex items have many virtues: Columns are, by default, the same height. That's normally very tricky to manage. In addition, you can distribute empty space automatically so that the browser can perform some magic—like aligning the buttons to the bottom of each column (bottom).

Formatting the Footer

You've worked your way to the bottom of this page. There's just the footer left to format. In the footer, there's a copyright notice and a signup form. This content would look good divided into two columns.

1. **Add two new styles to the bottom of the *custom.css* file:**

   ```
   footer .copyright {
        flex: 2 1 500px;
        margin-right: 30px;
   }

   footer .signup {
        flex: 1 1 250px;
   }
   ```

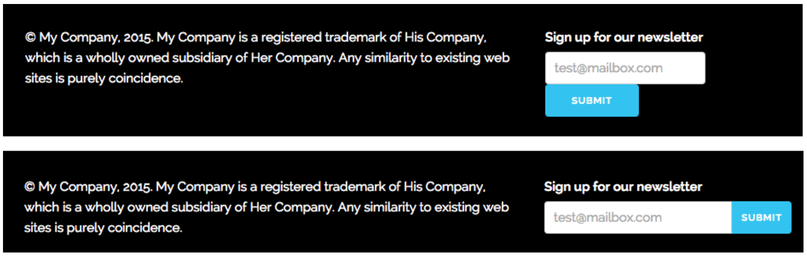
   These styles set the copyright area to be around twice as wide as the signup form. A bit of right-margin helps to push the signup form away from the copyright notice (Figure 17-20, top).

   The last thing to do is to organize the form elements. It would be great if the Submit button sat right next to the text field, perfectly aligned with it. You can do that with flexbox, but first you must turn the `<form>` tag into a flex container. Then, all of the elements inside—the label, the text field, and the button—can act like flex items.

2. **At the top of the *custom.css* file, update the style to include the selector** `footer form` **(changes in bold):**

   ```
   nav, .boxes, footer, footer form {
         display: flex;
         flex-flow: row wrap;
   }
   ```

   This selector makes the form inside the footer a flex container, and turns the label, the text field, and the button inside the form into flex items. Next, you'll make the label span the entire width of the form.


© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

FIGURE 17-20

Using flexbox, it's easy to align form fields, like the text field and submit button in the bottom image. To see other great examples of using flexbox with forms, visit http://philipwalton.github.io/solved-by-flexbox/demos/input-add-ons/.

3. **Add a style to the end of the *custom.css* file to make the label fill the column:**

```
.signup label {
  width: 100%;
}
```

In the previous step, you turned the form into a flex container that's set to wrap. Setting the width of the label in this rule to 100% forces the two input fields to wrap to another row. However, the text field and button are not sitting side by side. A couple of styles will fix that.

4. **Add two more rules to the end of the *custom.css* style sheet:**

```
.signup input[type="email"] {
  border-radius: 4px 0 0 4px;
   flex: 1;
}
.signup input[type="submit"] {
  border-radius: 0 4px 4px 0;
  padding: 0 10px;
}
```

These rules use attribute selectors (page 59) to identify the text field for receiving the user's email address and the button for submitting the form. Setting the email field's flex value to 1 means the field grows to fill all the available space that's not occupied by the Submit button.

In addition, by using `border-radius` and rounding only the left corners of the text field and the right corners of the Submit button, you've created what looks like a unified form widget. The result looks like the bottom image in Figure 17-20.

Changing the Navigation Bar for Mobile

Thanks to the flexibility of flex items, the design you've created works really well at different browser widths. Go ahead and save the *custom.css* file; open the *index.html* file and resize the browser window. You'll see that the design shrinks to fit the browser window: The three columns in the middle of the page turn into two columns, and eventually one column.

The last thing worth adding is a simple media query to convert the navigation bar from side-by-side buttons to stacked buttons at small screen sizes.

1. **At the bottom of the *custom.css* file, add a media query:**

```
@media (max-width: 500px) {

}
```

This media query works when the page is 500 pixels wide or less. In other words, the styles you place inside it will apply only to devices whose screens are narrower than 501 pixels.

Inside this media query, add a style to change how the flex items (the nav buttons) are displayed inside the navigation bar.

2. **Add the following style to the media query you added in the last step (additions are in bold):**

```
@media (max-width: 500px) {
nav {
     flex-flow: column;
  }
}
```

This changes the flow direction of the flex items. Instead of sitting next to each other in a row, the navigation buttons will sit one on top of the other in a column.

Finally, you can make the buttons fill the container.

3. **Add one last style to the media query so it looks like this:**

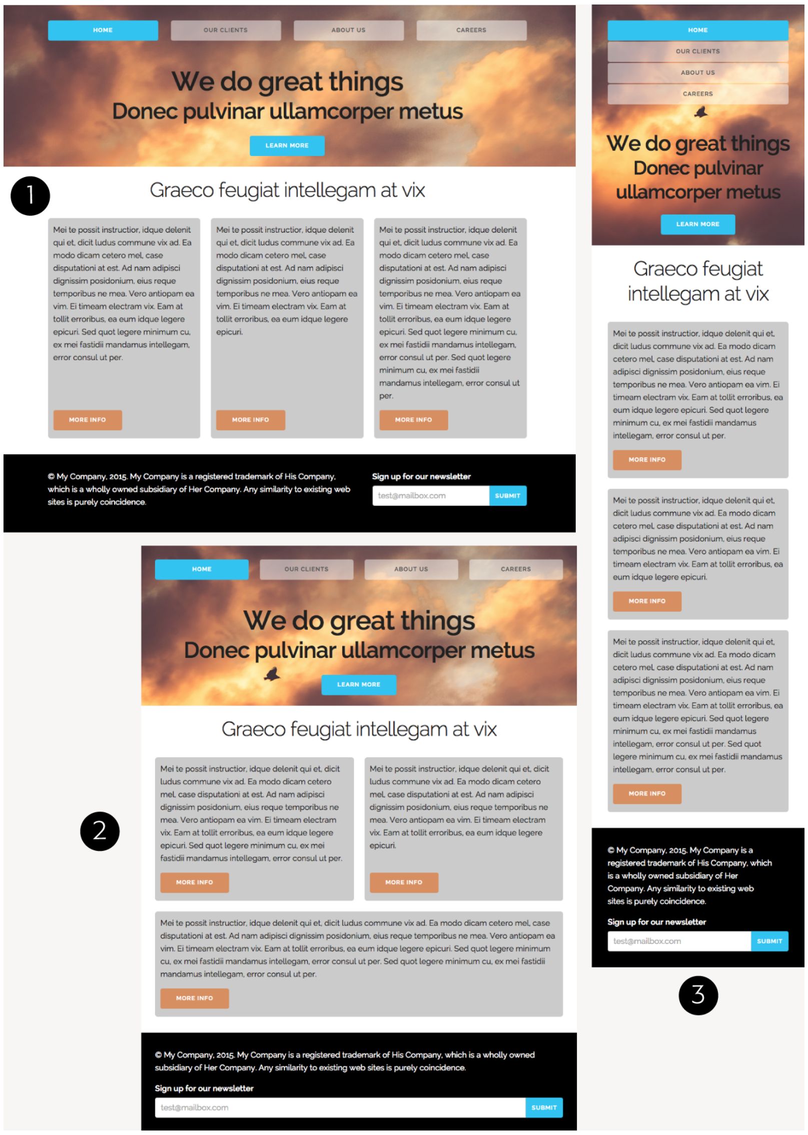
```
@media (max-width: 500px) {
  nav {
  flex-flow: column;
  }
  nav a {
    width: 100%;
    margin-bottom: 2px;
  }
}
```

This last style forces each link to fill the width of the flex container. It also adds a bit of space below each link to add visual separation.

4. **Save the *custom.css* file and preview the *index.html* file in a browser. Resize the browser, first dragging it to fill your screen, and then dragging it to make it narrower.**

See how the design changes as you narrow the browser window (Figure 17-21)? First, at full width (#1) the main content area has three columns, and the footer has two. Then, at a slightly narrower size, the three columns turn into two, and the footer elements stack on top of each other (#2). Finally, at a width of less than 500 pixels, all the content appears in a single column, and the navigation bar turns into a series of stacked buttons (#3).

Flexbox is a fun and exciting addition to CSS. It makes many previously difficult (or almost impossible) layout chores simple. It simplifies responsive design and reduces the math you need to create proportionally designed columns. Its biggest drawback is that not all browsers understand the flexbox syntax, and simply ignore flex properties. However, with support in all current browsers including Internet Explorer, there's no reason not to start experimenting with flexbox in your web designs.



We do great things
Donec pulvinar ullamcorper metus

Graeco feugiat intellegam at vix

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

test@mailbox.com

We do great things
Donec pulvinar ullamcorper metus

Graeco feugiat intellegam at vix

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

test@mailbox.com

We do great things
Donec pulvinar ullamcorper metus

Graeco feugiat intellegam at vix

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri.

Mei te possit instructior, idque delenit qui et, dicit ludus commune vix ad. Ea modo dicam cetero mel, case disputationi at est. Ad nam adipisci dignissim posidonium, eius reque temporibus ne mea. Vero antiopam ea vim. Ei timeam electram vix. Eam at tollit erroribus, ea eum idque legere epicuri. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per. Sed quot legere minimum cu, ex mei fastidii mandamus intellegam, error consul ut per.

© My Company, 2015. My Company is a registered trademark of His Company, which is a wholly owned subsidiary of Her Company. Any similarity to existing web sites is purely coincidence.

Sign up for our newsletter

test@mailbox.com

FIGURE 17-21

The CSS flexbox display mode lets you create designs that "flex," or change based on the width of the browser window. Used with media queries, flexbox lets you easily create responsive designs for all screen widths.