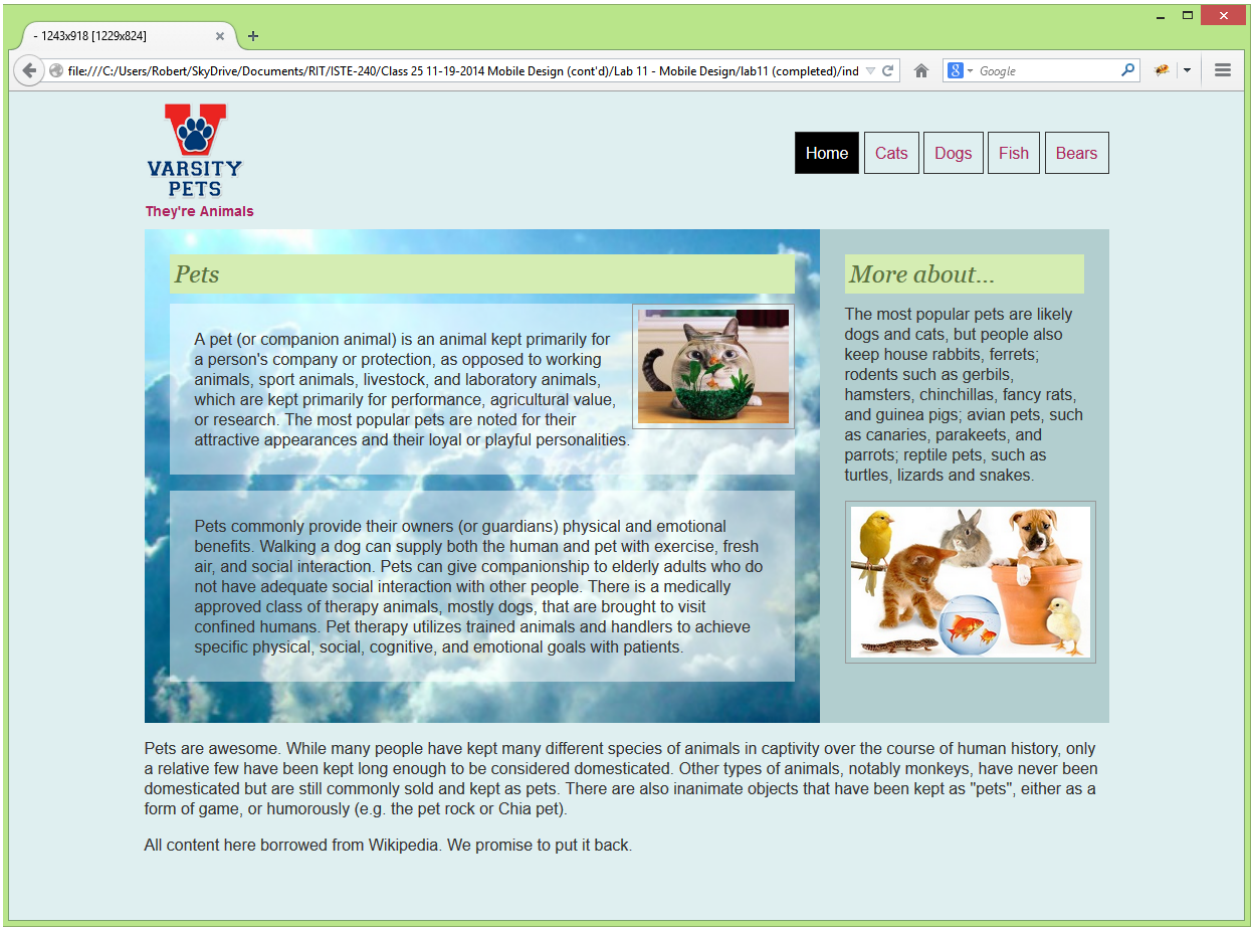
CSC 170 Lab 12: Mobile Design

Step 0: Setup

- From the Lab12 Lab Assignment in our section in Blackboard, download the file: **starter-files.zip** and extract the folder named **lab12** to your desktop or thumb drive.
- Open the **index.html** file in a web browser.
- Open the **index.html** file in a code editor.
- Open the **css/styles.css** file in a code editor.

The comps below shows how the provided webpage will look after you follow the instructions in this document.

Desktop view (as provided)



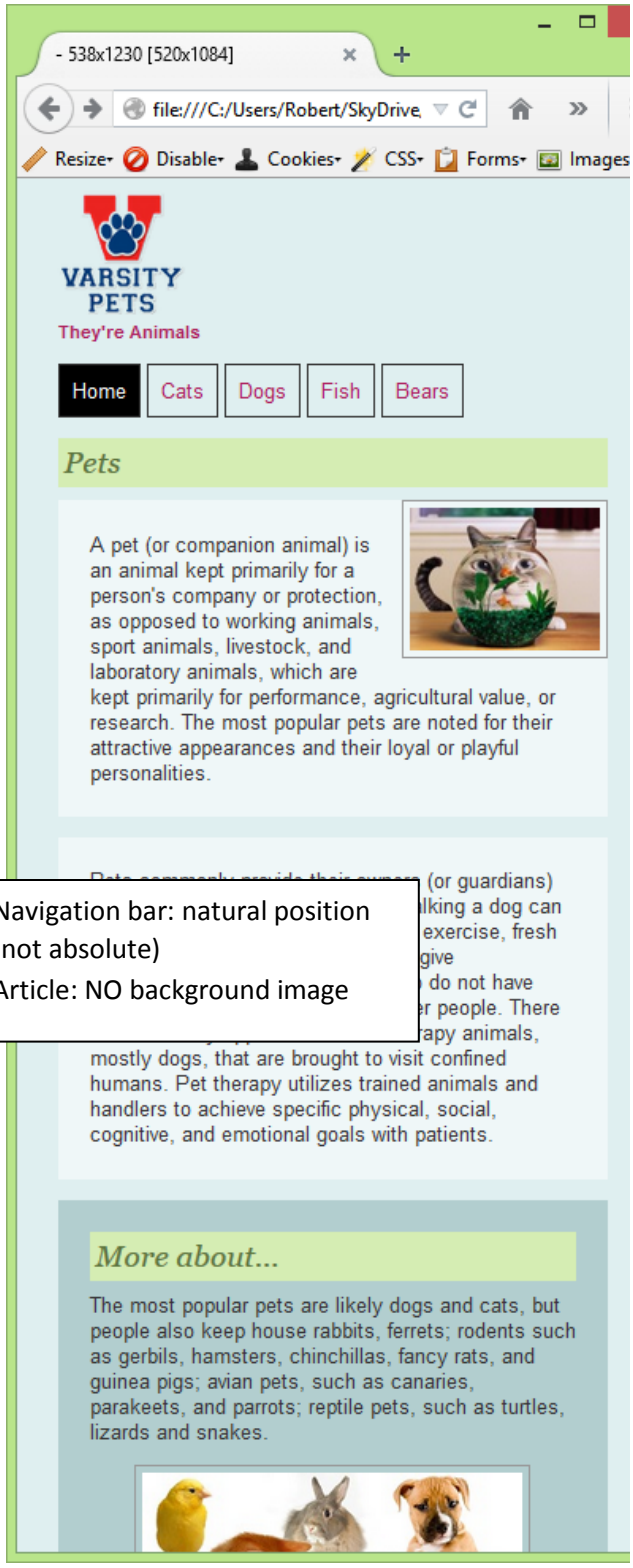
Notable desktop styles:

- Navigation bar: positioned using absolute
- Navigation list items: side-by-side using inline-block
- Article and Aside, side-by-side using table-cell
- Image in Article: floated right.
- Article: background image
- Image in Aside: scales to 100% of the aside's width

Tablet width

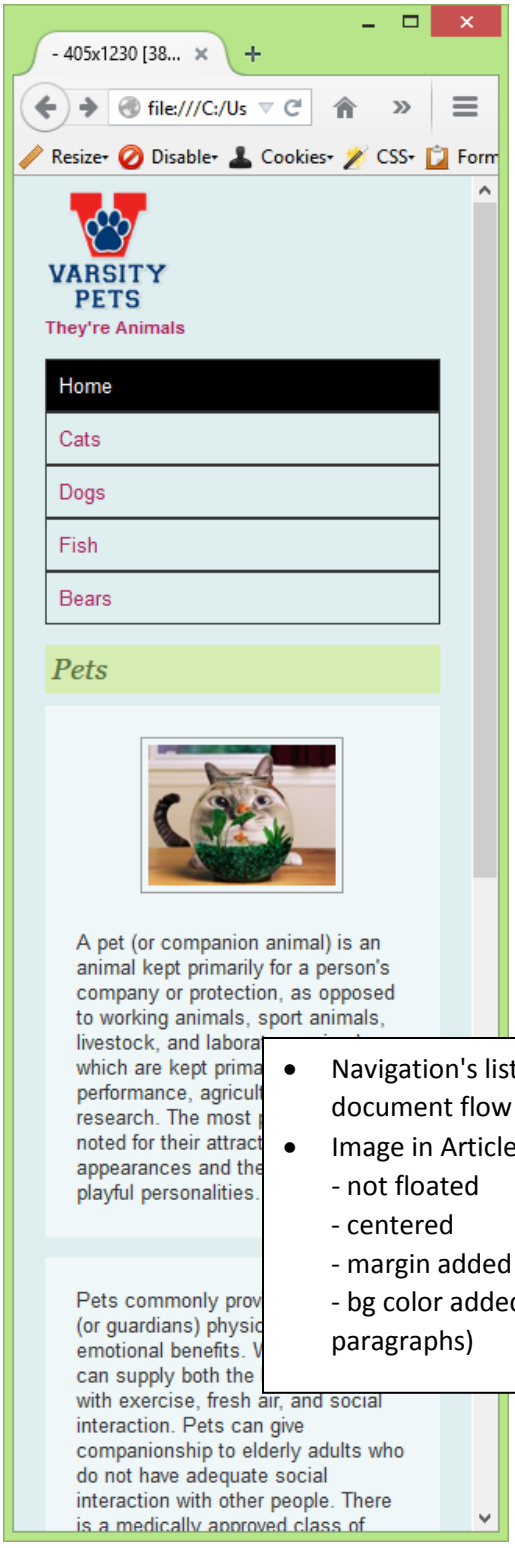
- Article and Aside: natural document flow (block)
- Aside: image centered; does not grow more than 300px

Large Smart Phone



- Navigation bar: natural position (not absolute)
- Article: NO background image

Smart Phone, Portrait



- Navigation's list items: natural document flow (block)
- Image in Article:
  - not floated
  - centered
  - margin added to the top
  - bg color added (to match the paragraphs)

Step 1: Override *Apple Safari for iPhone*'s default viewport

- ❑ In the **index.html** file, add the appropriate **meta** tag to set the initial scale of the browser to 100% of the default font as determined by the device's width.

  In the <head>, enter…

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Step 2: Set the webpage's overall content container to be responsive

As you work, be sure to check what you're doing in a web browser. Try expanding and contracting the width of the browser to see the effects.

- ❑ In the **styles.css** file, find the **.container** class and change it from pixel units to a proportional unit (a percentage) that is appropriate for what the container is supposed to do.

  in the CSS file, in the .container class, change to…

```
width: 80%;
```

- ❑ Add to the **.container** class, a maximum width that is appropriate to keep the container from growing too wide on wide-screen monitors.

  in the CSS file, in the .container class, add…

```
max-width: 960px;
```

Step 3: Set static elements to be responsive

- ❑ Look through the CSS. Any CSS that sets a pixel **width** (we don't care about height) to a block element must be changed to use proportional units (like a percentage). Set the units to something appropriate for what it's supposed to do.

  in the CSS file, in the **article** class, change to…

```
width: 70%;
```

  in the CSS file, in the **aside** class, change to…

```
width: 30%;
```

Note: there's a limit to what you have to set proportional. Only structural blocks need to have proportional units. Dimensions that just add a little "elbow room" side-to-side, probably *don't* need to be made responsive.

Step 4: Set images that should be responsive to be responsive

- ❑ Look through the HTML. REMOVE **height** and **width** attributes from all images. (Generally, you should never use them on any IMG elements.)

Look at the webpage in a web browser. Identify any images that *should be* responsive.

- Only the “pets2.jpg” image needs to be responsive.
- The other "pets" image can remain at its natural size.
- And generally speaking, logos should never be scaled, especially if they have words in them (like this one does).

- ❑ In the HTML, apply a class named **full** to the image that needs to be responsive like this…

```
<img class="full img-frame" src="images/pets2.jpg" alt="pets">
```

- ❑ In the CSS, create a class named **.full** and write the CSS to make the image(s) responsive.
  - A good place to put this class in the CSS file would be: at the bottom of the "Modules" section, with the other classes.

```
.full {
      width: 100%;
      max-width: 300px;
}
```

Again, as you work, be sure to check what you're doing in a web browser. Try expanding and contracting the width of the browser to see the effects.

The two **pets** images need to be centered when on smaller displays. To do this, they need to be placed into their own container.

- ❑ In the HTML, add a DIV around each of the two **pets** images like this:

```
<div>
      <img class="right img-frame" src="images/pets.jpg" alt="pets">
</div>
```

…and this:

```
<div>
      <img class="full img-frame" src="images/pets2.jpg" alt="pets">
</div>
```

- In the HTML, add a class called **image-holder** to both the new DIVs you created in the step, above like this…

```
<div class="image-holder">
      <img class="right img-frame" src="images/pets.jpg" alt="pets">
</div>
```

…and this:

```
<div class="image-holder">
      <img class="full img-frame" src="images/pets2.jpg" alt="pets">
</div>
```

- In the HTML, add a another class called **light-bg** to *only* the new DIV with the "**pets.jpg**" image (not the **pets2.jpg** image)…

```
<div class="image-holder light-bg">
      <img class="right img-frame" src="images/pets.jpg" alt="pets">
</div>
```

- In the CSS, create a class named **.image-holder** and write the CSS to make the DIVs centered.
    - A good place to put this class in the CSS file would be: at the bottom of the "Modules" section, with the other classes.

```
.image-holder { text-align: center; }
```

Step 5: Set breakpoint(s) using media queries

- Create media queries in your webpage's CSS like this.
    - (A good place to put this media query in the CSS file would be: at the bottom of the file, after all other CSS.)

@media (*max-width: 720px*) {

}

- INSIDE the media query with a breakpoint set at max-width: 720px, override the **article** and **aside**'s display and width settings like this…

```
.article, aside {
      display: block;
      width: auto;
}
```

Check it out in a web browser. Expand and contract the width of the browser to see the effects.

- ❑ Create another media query in your webpage's CSS like this.
  @media (*max-width: 540px*) {
  }

- ❑ INSIDE the media query with a breakpoint set at max-width: 540px, override the **article** and **nav** like this…

```
article {
      background-image: none;
      padding: 0;
}
nav {
      position: static;
}
```

Check it out in a web browser! The ARTICLE and ASIDE will jump top-over-bottom when you reduce the width of the viewport (browser window).

- ❑ Create a third media query in your webpage's CSS like this.
  @media (*max-width: 390px*) {
  }

- ❑ INSIDE the media query with a breakpoint set at max-width: 390px, override the **nav li** and **.right** class like this…

```
nav li {
      display: block;
}
.right {
      float: none;
      margin-left: 0;
      margin-top: 25px;
}
```

Step 6: Check and Upload your Work

Save all your work and check to see if it replicates the comps shown at the end of this document as you resize the width of your web browser.

When you are done with your webpage, close everything and use an FTP tool (like *WinSCP*) to access your account on **urcsc170.org** and upload your files:

- ❑ In a web browser (any), go to this address to check your handiwork:
  www.urcsc170.org/*accountname*/lab12/index.html
  (where "*accountname*" is your account name)

Step 7: Report your work

- In our Blackboard section, in Lab 12, post a link to your webpage to receive credit for this Lab.