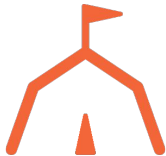


Front End

HTML & CSS



HTML & CSS

HTML = Hypertext Markup Language

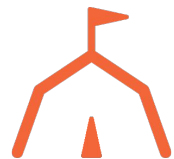
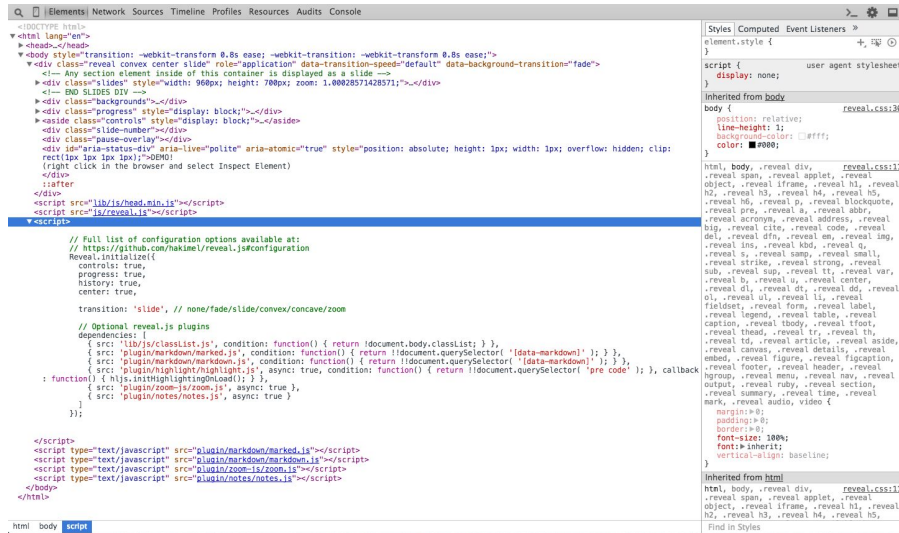
CSS = Cascading Style Sheets

Every website on the internet uses HTML & CSS, and about 96% use JavaScript.



Your new best friend

It's called the Dev
Tools Inspector (right
click in the browser
and select **Inspect
Element**)



Nesting and Indenting

HTML elements "nest" inside of one another.

The element that opens first closes last.

Understanding how to nest elements is crucial to writing clean and maintainable code.



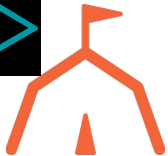
Comments

HTML allows for comments.

Comments are useful for keeping code organized while developing.

The browser engine will ignore them.

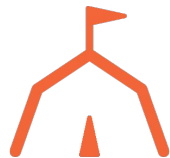
```
<!-- Hello, I am a comment. -->
```



Folder Structure

Folder structure may not be exciting, but it is important.

How a folder is structured will depend on the preferences of whoever is building the project.

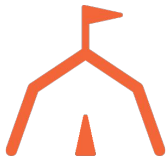


HTML



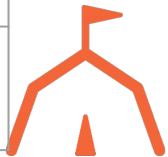
Content Tags

Tag	Description
<code>div</code>	The defacto container element.
<code>p</code>	Used for body copy.
<code>h1-h6</code>	Designating titles/subtitles.
<code>ol</code>	Create a numbered (ordered) list.
<code>ul</code>	Create a non-numbered (unordered) list.
<code>li</code>	List elements.
<code>a</code>	Link to other pages or sites.
<code>link</code>	Import external documents.



Semantic Tags (HTML5)

Tag	Description
<code>section</code>	The “new” div tag, used for page organization.
<code>header</code>	Introductory and navigational content.
<code>main</code>	Content inside should not be repeated.
<code>footer</code>	Footer content (site map, internal links, etc.).
<code>nav</code>	A list of navigation links.
<code>audio</code>	Multimedia tag for playing audio files.
<code>video</code>	Multimedia tag for playing video files.
<code>canvas</code>	Element can be used to draw graphics using JavaScript.



HTML5

HTML5 is a standard that has come to make web development more programmatic and standardized.

This goes along with improvements to CSS, and JS.

Moral of the story: it's a great time to learn web development!

HTML



Tables

Tables are used to display data, NOT organize and position content.

The only exception to this rule is for developing emails, due to client restrictions.



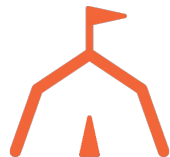
What about breaks, bolds, and horizontal rules?

Any inline styling elements such as ``, `<i>`, or `
` should not be used.

If an element needs to have emphasis (bold or italics), use a `` or `` element.

Anyone on a screen reader or PDA will appreciate this consideration.

If an element needs spacing, use `margin`.



CSS



What is CSS?

CSS is short for Cascading Stylesheets.

It is the language for styling HTML.

CSS can be written to do anything from changing fonts and colors to creating beautiful transitions and animations.



CSS Selectors

The selector instructs the browser to search the page for any HTML element that matches the given criteria.

It applies any applicable declarations to the matched element(s).



CSS Selectors

Selector	Description
tagname	Selector based on the element's tag name.
.	Selector based on a class name.
#	Selector based on an id.



CSS Declarations

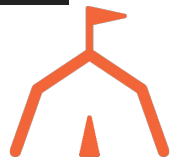
Declarations are made up of the property needing to be altered and the value given to the property.

Declarations can be grouped so that more than one declaration may be applied to a selected element.

Curly brackets must surround declaration groups.

Declarations must end in a semicolon.

```
.selector {  
  background-color: red;  
  color: white;  
  border: 1px solid black;  
  border-radius: 5px;  
}
```



Comments

Just like HTML, CSS offers comments:

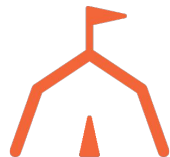
```
/* This is a CSS comment, it can be multi-line */
```



CSS Pseudo-Classes

```
a:link { /* unvisited link */  
  color: aliceblue;  
}  
a:visited { /* visited link */  
  color: darkblue;  
}  
a:hover { /* mouse over link */  
  color: lightblue;  
}  
a:active { /* mouse click link */  
  color: yellow;  
}
```

Used to target specific states of an HTML element.



Other Pseudo-Classes

Pseudo-Class	Description
<code>:first-child</code>	The first direct child of its parent.
<code>:last-child</code>	Any element which is the last child of its parent.
<code>:only-child</code>	Any element which is the only child of its parent.
<code>:empty</code>	Any element which has no content.
<code>:nth-child</code>	Any element which is the nth child of its parent.



Other Pseudo-Classes Demo

<http://jsbin.com/gebeqehewe/1/edit?html,css,output>



nth-Child Syntax

`:nth-child` takes an expression to determine which children to select.

It is common to use `:nth-child` for alternating styles of large groups of data like table rows, columns, or lists.

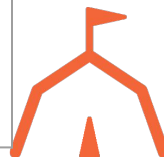
```
p:nth-child(an + b) {  
    property:value;  
}  
  
div:nth-child(3) {  
    color:red;  
}  
  
ul:nth-child:(2n + 5) {  
    color:yellow;  
}
```



Pseudo-Elements

Used to target specific parts of an HTML element.

Pseudo-Element	Description
<code>::after</code>	Inserts something after the element's content.
<code>::before</code>	Inserts something before the element's content.
<code>::first-letter</code>	Selects the first letter of the element's text content.
<code>::first-line</code>	Selects the first line of the element's content.
<code>::selection</code>	Selects a portion of the element that is selected by the user.



CSS Properties

There are hundreds of CSS properties that are available for use.

We will go over a few of the most common, however, the best strategy is to Google if curiosity strikes.



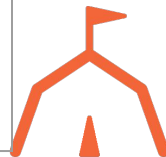
CSS Properties

Property	Description
<code>background-color</code>	Background color for an element.
<code>color</code>	Color of text AND border of an element.
<code>font-family</code>	Typeface for text.
<code>font-size</code>	Size for text (px, %, em, pt).
<code>font-weight</code>	Used to make text bold (if possible).
<code>text-decoration</code>	Used for changing decorations (underline/strikethrough).
<code>height</code>	Specifies the height of an element.
<code>width</code>	Specifies the width of the element.



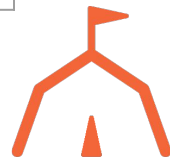
Color in CSS

Method	Syntax	Description
color name	<code>white</code>	One of the 140 predefined colors.
hexadecimal	<code>#FF0000</code>	RGB values in hex form. Each value can be within the range of 0-9 or A-F.
RGB	<code>rgb(255, 0, 187)</code>	RGB values.
RGBA	<code>rgba(255, 0, 187, 0.5)</code>	RGB values with an added alpha (opacity) value.



CSS Units

Method	Syntax	Description
<code>em</code>	<code>1em</code>	Scalable unit based on font size of the parent element.
<code>rem</code>	<code>1rem</code>	Scalable unit based on font size of the html root.
<code>pixels</code>	<code>16px</code>	Fixed number of pixels.
<code>percent</code>	<code>120%</code>	Percent value based on font size of the parent element.

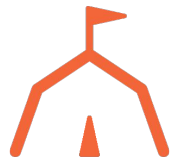


Exercise

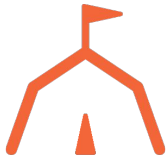
Pick from one of two images linked below.

Try to re-make these images with HTML and CSS.

Be ready to demo! [Option 1](#) / [Option 2](#)

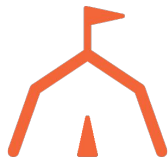


Display Properties



Display

Display	Description
<code>block</code>	Takes up the horizontal space of a line. This will stretch to fill all the space from left to right of its parent container.
<code>inline</code>	Rendered without starting a new line. They appear side by side until reaching the edge of its parent container.
<code>inline-block</code>	Inline element that can have a height and width declared.
<code>flex</code>	Acts like a block element but its direct children can be aligned, justified and flexed.



Display Demos

BLOCK:



DEMO

INLINE:



Block vs. Inline Elements

Inline elements: `<a>`, ``, ``.

Block elements: `<div>`, `<p>`, ``,
``, `<table>`, and just about everything
else.



The Box Model



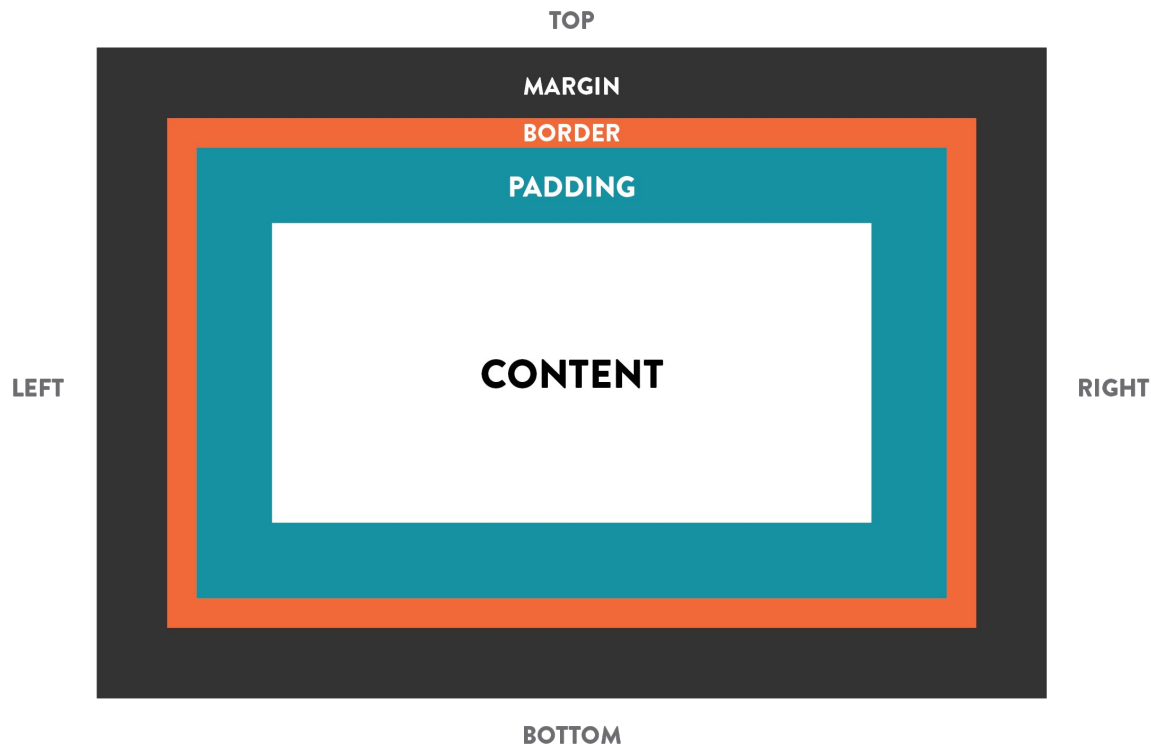
The Box Model

Every HTML element is in a box, regardless of its visible shape.

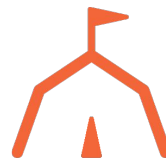
The total size of an element is a combination of the following: **content**, **padding**, **border**, and **margin**.



The Box Model

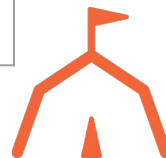


[DEMO](#)



The Box Model

Property	Description
padding	The space between the content within an element and the border. Padding can be specified in units of pixels (px), Ems (em), or points (pt) or percentage (%) of the containing element.
border	The edge around the element. It has elements of thickness, style, and color.
margin	The space outside the element that separates it from other elements. It is 'outside the fence' in terms of its relationship to the border.



Margin

```
p {  
  margin: 10px; /* 10 pixels  
    of margin on all sides */  
}
```

```
p {  
  margin: 10px 5px; /* 10 px  
    on top/bottom & 5px on  
    left/right */  
}
```

```
p {  
  margin-top: 5px;  
  margin-right: 10px;  
  margin-bottom: 15px;  
  margin-left: 20px;  
}  
  
p {  
  margin: 5px 10px 15px 20px;  
  /*top, right, bottom, left*/  
}
```

Centering with Margin

`margin: 0 auto` can be used on an element that has a set width to center the element.

This method only works for horizontal centering.

[DEMO](#)



margin auto vs. text-align center

`margin: 0 auto` centers the given element within its parent.

`text-align: center` centers the text that is inside the given element.

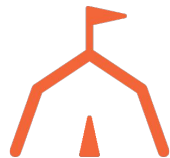


Centering on X and Y

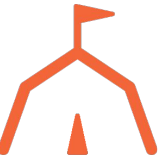
A solid choice for centering (both horizontally and vertically) is using a translate.

```
.vertical {  
  position: relative;  
  top: 50%;  
  transform:  
translateY(-50%);  
}
```

```
.horizontal {  
  position: relative;  
  left: 50%;  
  transform:  
translateX(-50%);  
}
```



Positioning



CSS Positioning

Every HTML element has a property called `position`, which dictates how that element flows on a document.

This property can be set to many different values, each of which behaves slightly differently.

Positioning gives a web developer more significant control over the design and overall look of a page.



Positioning

Position	Description
<code>static</code>	Default for all HTML elements. Adheres to the previously discussed behavior of block and inline elements.
<code>relative</code>	Relative positioned elements appear in the normal flow of the document but can be offset by using the top, bottom, left and right properties.
<code>absolute</code>	Absolutely positioned are removed from the normal flow. They appear relative to their nearest positioned parent element, using offsets.
<code>fixed</code>	Fixed positioned elements are removed from the normal flow. They appear relative to the viewport or the nearest transformed parent.



Absolute Positioning

When an absolutely positioned element is inside another positioned element, it is positioned relative to that container, rather than the whole page.

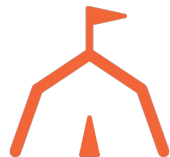
[DEMO](#)



Exercise

You try! Make a web page with 4 elements. Heads up: This will look bad.

1. One should have a header tag with a background color of your choice, 100% width, and fixed positioned to the top, right of the viewport.
2. Give the body a viewport height of 200vh.
3. Make a div (give it a width and a height) under the header and give it a border of 1px solid black.
4. Put a paragraph tag with your name inside the div and absolutely position it to the bottom, right of the div.
5. Add an image of your choice without any positioning (this is your static positioned item).



More Properties of Interest



Float

Floating takes an element in the normal flow and pushes it as far to the left or right of its parent element as possible.

When an element is floated, other elements will wrap around it.

To float an element, you must specify a direction to float.



Float

none	hidden	dotted	dashed
solid	double	groove	ridge
inset	outset		

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat

non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



Clear

Clear specifies on which side of an element other elements cannot appear.

Clears can be applied to left, right, or both.

[DEMO](#)



Z-Index

When elements are moved out of the normal flow of content (i.e., not position static), they can overlap.

Z-index can be used to define the order of overlapping elements.

The element with the highest z-index goes on top.

Z-index will NOT work on an element with position static.

[DEMO](#)



@font-face

The @font-face property allows the use of non-default fonts on websites.

Using external fonts is quite easy. Follow these steps:

1. Download your font
2. Place the font file in your web site
3. Create a @font-face CSS rule

```
@font-face {  
    font-family: "myCoolFont";  
    src:  
    url("path/to/myCoolFont.ttf"  
);  
}  
  
div.demo {  
    font-family: "myCoolFont";  
}
```



Exercise

1. Download a free font from font-squirrel.
2. Place the font file in your pre-work, this cannot be a CDN.
3. Create a `@font-face` CSS rule and use it somewhere in your project.



CSS3 Properties and Prefixes



Vendor Prefixes

Not all browsers support all of the newer CSS3 properties, which means additional rules must be created for specific browsers.

[Great Prefix Resource](#)

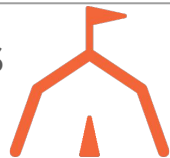
The order matters, so make sure the non-prefixed property goes last.

```
-webkit- /*Android*/  
-webkit- /*Chrome*/  
-webkit- /*iOS*/  
-webkit- /*Safari*/  
  
-o- /*Opera*/  
  
-moz- /*Firefox*/  
  
-ms- /*Internet Explorer*/
```

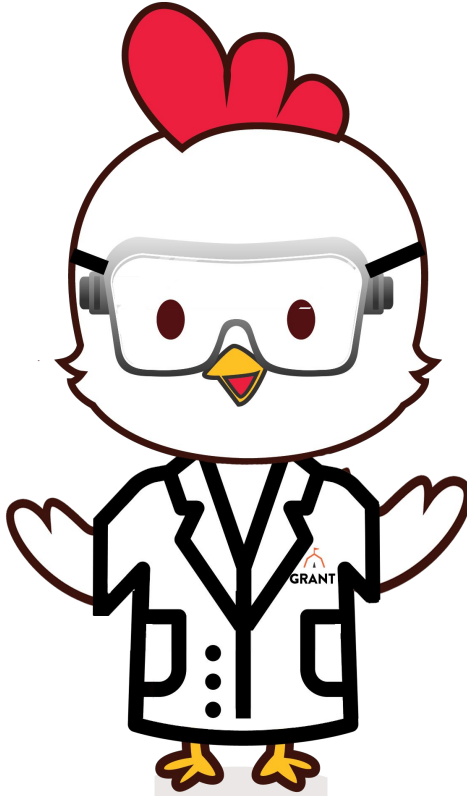


CSS3 Properties

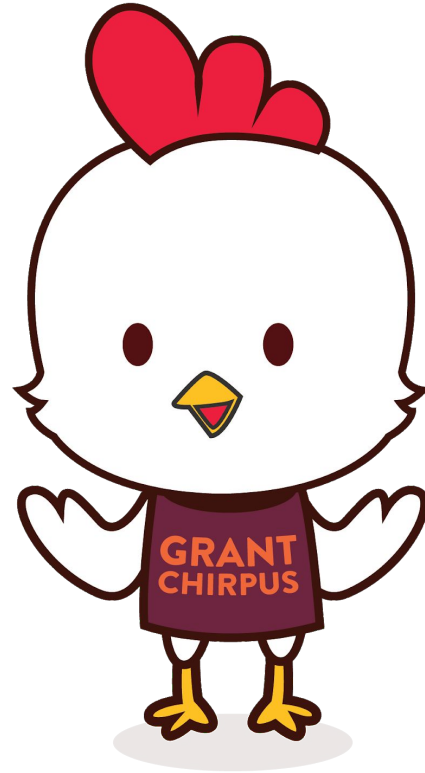
Property	Description
<code>border-radius</code>	Creates rounded corners or circular borders for elements.
<code>opacity</code>	Controls the transparency of an element.
<code>box-shadow</code>	Adds a calculated shadow effect to an element.
<code>text-shadow</code>	Adds a calculated shadow effect to text.
<code>transforms</code>	Adds the ability to transform HTML elements (rotation, scale, translate).
<code>animations</code>	Allows for one or more property changes to be applied as an animation.



HTML/CSS Lab part 1



**NEW
CONTENT
AHEAD!**



Responsive Development



What is responsive?

For a site to be responsive, it has to be able to be viewed on multiple screen sizes and screen orientations (landscape or portrait).

With so many different types of devices out there, all of them must be recognized and coded for (welcome to front-end development).

This is done by creating fluid layouts using flexbox, css grids and media queries.



Tools for Responsive Development

Flexbox for responding to parent and siblings in one dimensional layouts

Grid for responding to parent and siblings in two dimensional layouts

Media Queries for responding to device differences



Flexbox

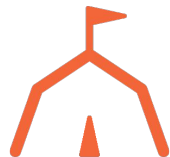


Exercise

A client wants you to make a site for their new dog walking business.

The client's head developer wants you to build our navigation using floats because that is what is being used on the rest of their site and she is most familiar with it.

The designer gives you the .png in the next slide, so you should make sure it looks like what the client wants!



Exercise

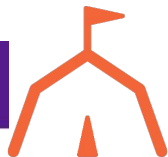
Remember - You can only use `floats` and `display: inline-block` right now.

The font they want you to use is `Lato`.

The background color of the navigation is `#551A8B`.

The font color is `rgba(255, 255, 255, 0.8)`.

Good luck!



Flexbox

Think of a world where people do not have to use floats or inline-blocks for their layout!

Flexbox is a display property within CSS that allows a developer to plan and visually display their site's content.

There are tons of values that can be leveraged with flexbox.



Benefits of Flexbox

1. Eliminates the need for calculating the heights of elements in JavaScript to set equal heights of divs.
2. No more floats or need for clear fixes.
3. Flexbox works beautifully for responsive layouts.
4. Flexbox calculates the widths of each element and sets the margins for us.

[Here's](#) using flexbox to accomplish equal heights through CSS!

[Here's](#) using flexbox eliminates the need for clear fixes.



How to Flex



The Parent Container

To use flexbox, wrap the items needing to be flexed in a container.

The container is often referred to as the **flex container** or as the **parent container**.

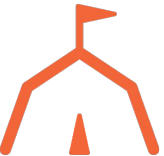
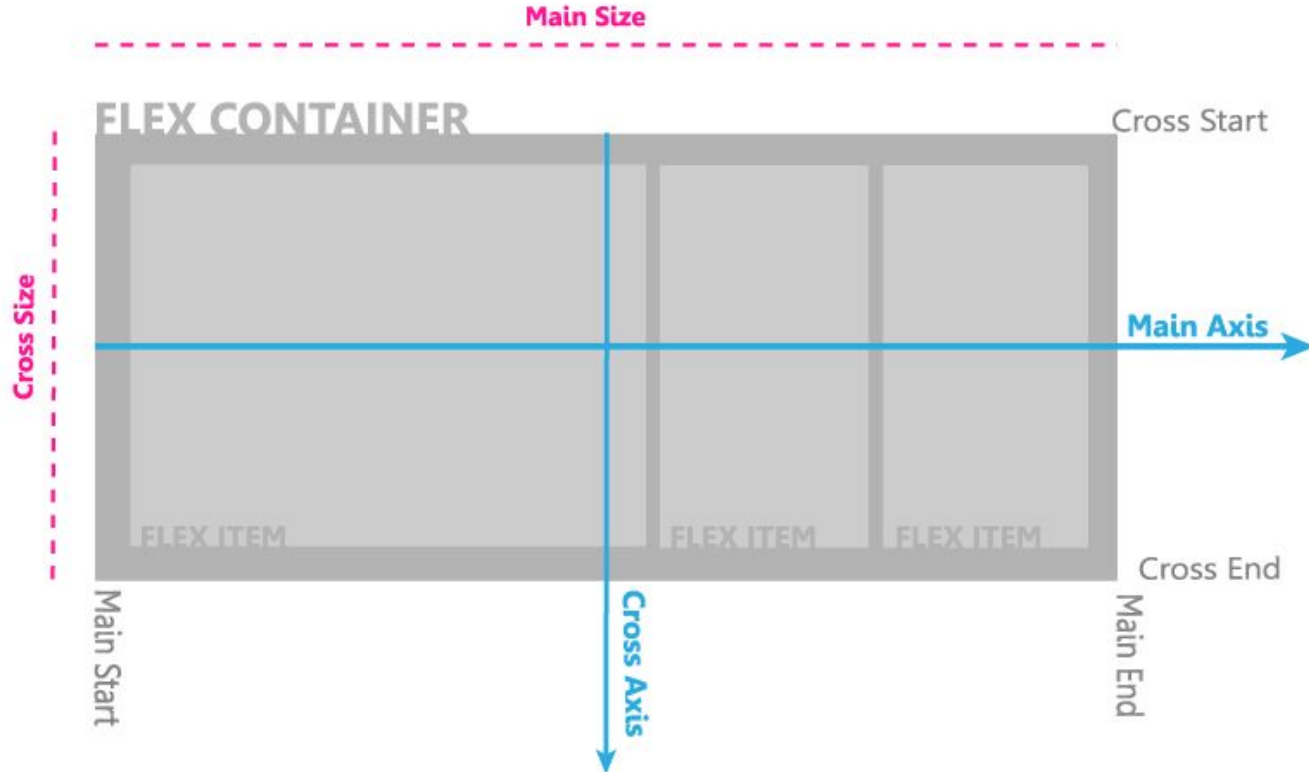
Do this by giving the display property a flex value. It should look like this:

```
.parent-container {  
  display: flex;  
}
```

Here's how we do it.



Cross Axis and Main Axis



Cross Axis

The cross axis is the vertical axis along the flex container.

This allows for elements to move between the top, middle or bottom of the container.

The `align-items` property is what controls this axis.

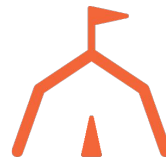
The `align-items` property goes on the parent container and will be applied to child items.



Align-Items

Value	Description
<code>flex-start</code>	Items will line up at the top of the container.
<code>flex-end</code>	Items will line up at the bottom of the container.
<code>center</code>	Items will be centered in the container.
<code>baseline</code>	Items will be lined up along their baseline.
<code>stretch</code>	Items will stretch as tall as the container. (default)

[DEMO](#)



Main Axis

The main axis is the horizontal axis along the flex container.

This allows for elements to move between the left, middle or right of the container.

Spacing can also be applied either around or between these items.

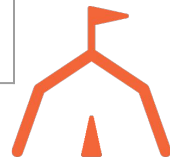
The `justify-content` property is what controls this axis.

The `justify-content` property goes on the parent container and will be applied to child items.

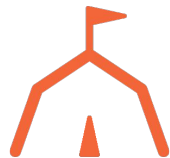


Main Axis

Value	Description
<code>flex-start</code>	Items will line up at the top of the container. (default)
<code>flex-end</code>	Items will line up at the bottom of the container.
<code>center</code>	Items will be centered in the container.
<code>space-between</code>	Flex items are evenly distributed between items.
<code>space-around</code>	Flex items have space around all sides.



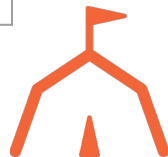
DEMO



Flex-Direction

Flexbox items can go in two directions, rows and columns.
The direction can also be reversed. [DEMO](#)

Value	Description
<code>row</code>	Items will flow left to right. (default)
<code>row-reverse</code>	The order of items will reverse row order.
<code>column</code>	Items will stack from top to bottom.
<code>column-reverse</code>	Items will stack in a reversed order from top to bottom.

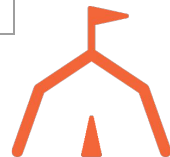


Flex-Wrap

The `flex-wrap` property controls whether the flex container is single-line or multi-line, which determines the direction new lines are stacked in.

Value	Description
<code>nowrap</code>	The default value. Items fall on a single line, even if they start to overlap each other.
<code>wrap</code>	Items fall on another line if there isn't enough room on the current line.
<code>wrap-reverse</code>	Items go on top of the previous line if there isn't enough room on the current line.

[DEMO](#)



Exercise

Oh no!

It turns out the head developer took a flexbox course and realized the magic of flexbox.

Convert your old code to use flexbox, but still achieve the same look as before.

Here's the image of the navigation again for you to reference.

Grant's Dog Walking!

[Home](#)

[Prices](#)

[Contact Us](#)



Exercise

Great job!

The client was super happy with your flexibility and knowledge of flexbox.

Less code for them to implement in their project does not hurt either.



Reference

<https://cssreference.io/flexbox/>

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Fun Game!

<https://flexboxfroggy.com/>



Media Queries

One of the ways to accomplish having responsive websites is by using media queries.

Media queries can detect a bunch of properties about the device used to view the site.

It allows the browser to use specifically developed CSS styles for the different screen sizes that it detects.



Media Queries

Media queries are written like this:

```
@media [not|only] type  
[and] (expression) {  
    /* rules */  
}
```

Types of media can be all, screen, print, and speech. [Here's](#) an example of how to use media queries.



Min and Max Width

`min-width` applies the styles to anything greater than or equal to the min-width

`max-width` applies the styles to anything less than or equal to the max-width

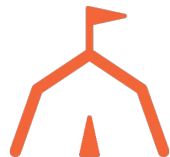


You Try It!

Create a container element with three paragraph tags that have the same font size value.

Make their font size get more prominent by increasing the font-size as the viewport increases using two media query breakpoints.

Things to think about: will you need to use max-width or min-width for this situation?



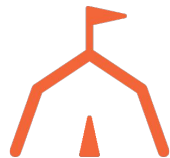
Viewport Meta Tag

The following code must be included within the `<head>` element.

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

The previous tag tells the browser to render the width of the page at the width of its screen.

If that screen is 320px wide, the browser window will be 320px wide, rather than way zoomed out.



Mobile-First Development



Mobile-First Development

In this class, we will emphasize coding all sites mobile-first.

Mobile-first means any default styles will go above media queries, and any overriding code will go within media queries.

Typically, min-width is used for mobile-first development.



Mobile-First

In 2019, it is expected that there will be 4.68 billion mobile users.

It is far easier to build up than to scale down.

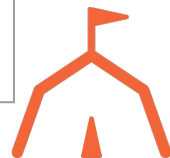
Mobile-first is also great for quicker load times and has a much higher SEO (Google ranks mobile-first sites higher).



Mobile-First

The standard device breakpoints to use are:

Breakpoint	Device Types
480px	Larger phone screens.
768px	Most tablets.
992px	Large tablets to small laptops.
1200px	Anything larger than a tablet, like desktop screens.



Exercise

Let's go through how a mobile-first developer would typically code and test their site as they work.

Let's come up with a wireframe of what we want our site to look like together going from mobile to desktop.

We should minimally include a title, an about us section with two facts.



Exercise

Grant's Dog Walking Company has been doing great since you built that impressive navigation.

However, now, it has gotten so popular that people on their phones are trying to access it.

No one can use it because we did not develop it mobile-first!

Our client wants us to re-do our navigation (still using flexbox) but to work responsively.



Exercise

Here is the design.

They also want the name of the company to go on the right side on bigger screen sizes without changing the HTML structure.

Plan for them to change at the 768px breakpoint.



Additional Resources

Here are some additional readings to check out:

[Flex-Flow Property](#)

[Flexbox Cheat Sheet](#)

[Mobile-First Importance](#)

[More Media-Query Resources](#)



HTML Forms



Forms

Forms are used to send data back to a server from user inputs.



Form Tags

```
<form action="some-url" method="get">  
    <!-- Form inputs and other html -->  
</form>
```



Methods and Actions

Action is the URL you want the data to be sent to.

Method is the HTTP methods used to transfer data.

- GET
- POST
- And others



Form Tags

```
<input type="text" name="firstname"  
placeholder="Name?">
```

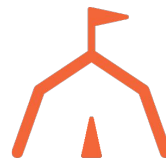


Types of Inputs

- Text (type="text")
- Radio Button (type="radio")
- Checkbox (type="checkbox")
- Password (type="password")
- Date (type="date")
- Time (type="time")
- Date & Time (type="datetime")
- Email (type="email")
- Submit
- `<input type="submit"/>`
- `<button type="submit"/>`



Demo

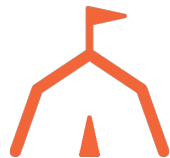


Text

```
<input type="text"/> Starts out empty.
```

```
<input type="text" value="Hello"/> Starts out  
with 'Hello' typed in.
```

The 'value' attribute determines what text it starts out with.



Checkbox

```
<input type="checkbox" checked/> I'm checked.
```

```
<input type="checkbox"/> I'm NOT checked.
```

The presence or absence of a 'checked' attribute determines whether it starts out checked.



Radio Buttons

```
<input type="radio" name="group-name" checked/> I'm  
checked.
```

```
<input type="radio" name="group-name"/> I'm NOT checked.
```

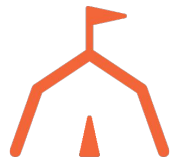
Radio buttons that have the same 'name' are grouped together. Only one of the buttons in a group can be checked at a time.



Select (Drop-Down)

```
<select>  
<option>Red</option>  
<option selected="">White</option>  
<option>Blue</option>  
</select>
```

One of the options can have the 'selected' attribute, which pre-selects that option.



Labels

```
<!-- input nested inside label -->
```

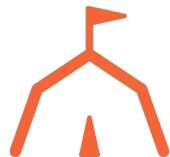
```
<label><input type="text"/> Describe the input</label>
```

```
<!-- The for attribute matches the id attribute, binding the two  
together -->
```

```
<label for="color">Color</label>
```

```
<input type="text" id="color"/>
```

It's good to associate a label with an input.
Here are two ways.



Labels

```
<!-- input nested inside label -->
```

```
<label><input type="checkbox" checked/> Click Me!</label>
```

```
<!-- The for attribute matches the id attribute, binding the two  
together -->
```

```
<label for="hasDog">Has Dog</label>
```

```
<input type="checkbox" id="hasDog"/>
```

It's especially important to give checkboxes and radio buttons a label. Then the user can click either the label or the checkbox.



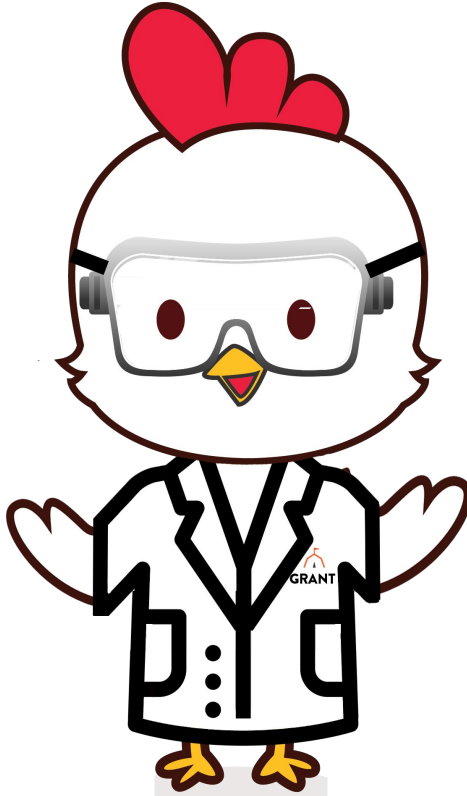
Forms Activity

Open a new JS Bin

- Add a form.
- Add a checkbox. Make sure it has a clickable label.
- Add a group of two or more radio buttons. Make sure they have labels.
- Add a dropdown (select) with at least two options.



HTML/CSS Lab Part 2



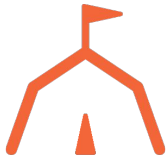
Wireframing and UX/UI



UX/UI (source)



Terminology



What is UX?

UX means user experience. It relates to the process of creating products that will provide personal and meaningful experiences.



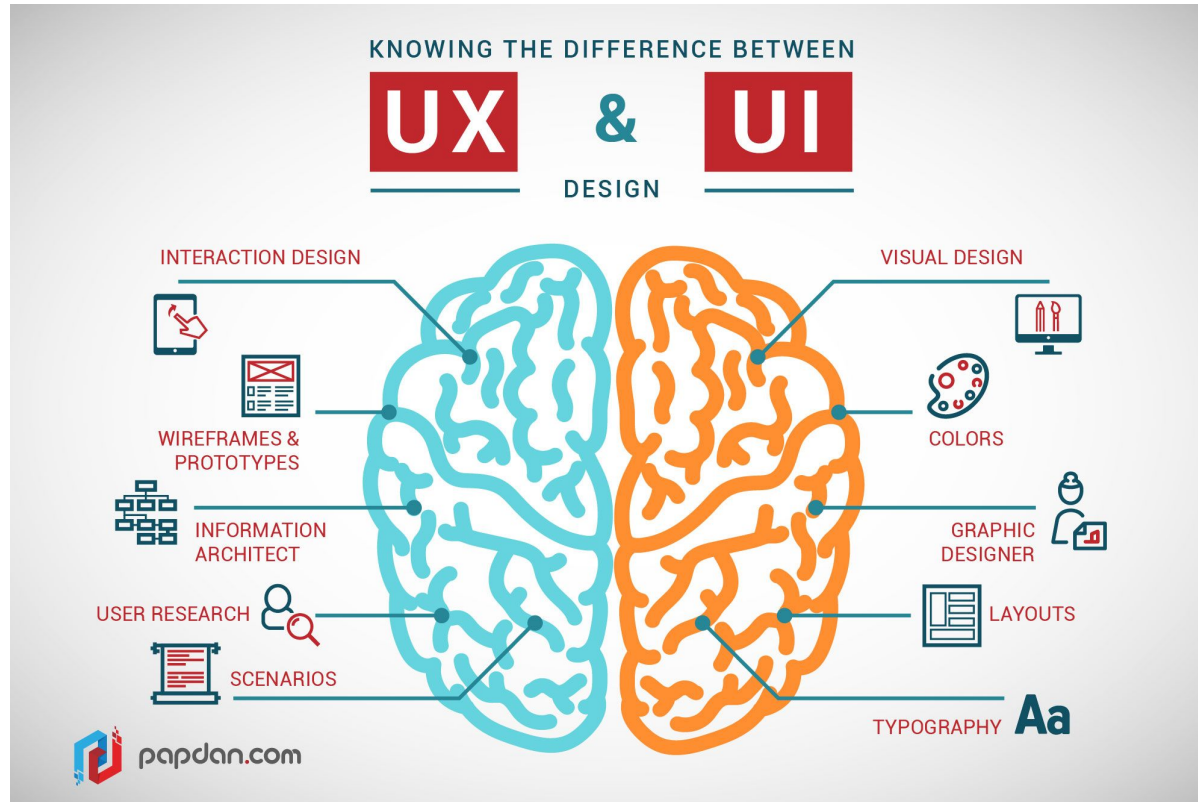
What is the UI?

UI refers to the user interface, it relates to the process of designing user interfaces for software or machines, such as the look of a mobile app, with a focus on the ease and enjoyment of the user.

It usually refers to the graphical interface and how the product is laid out.



What's the difference?



What's the difference?

UI is how things look, UX is how things work.

UI makes interfaces *beautiful*, UX makes interfaces *useful*.



User Interface



UI Examples

Inspiration UI



UI Core Principles

Clarity: Things need to be apparent to the user- like what happened, what they are, what they can do and what will happen if they do it.

Flexibility: Designing something that looks good in all situations.

Familiarity: Using familiar items and patterns in your app.



More UI Core Principles

Efficiency: User finishes main task in the most efficient way possible.

Consistency and Structure: Fonts and alignment are consistent and shareable elements are reused.



UI Design Tools

[Sketch](#) / [Adobe XD CC](#)



UI Frameworks/Kits

[Bootstrap](#) / [Materialize](#) / [UI Kit](#) /
[Bulma](#) / [Semantic UI](#)

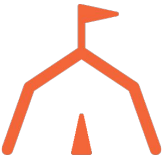


UI Activity

Make a contact form using Bootstrap styles! Begin with the HTML template from the Get Started guide on Bootstrap's website. The contact form should include the following:

1. 2 text inputs
2. two radio buttons
3. A text area
4. A submit button
5. Any other components of your choice (At least one)

Be ready to demo!



User Experience



History of UX

"When we go into a new project, we believe in it all the way. We have the confidence in our ability to do it right." -Walt Disney (One of the original UXers)



History of UX

You could say UX started during the machine age.

The easier it was to use the machines, the quicker the work would get done.



History of UX

A good example of experience design is Disneyland (opened in 1955)

Walt Disney was a pioneer connecting experience and emotion.



UX Examples

[Pendar Yousefi Portfolio](#) / [Kathy Li](#)



UX Core Principles

Useful: Should be original and fulfill a need.

Usable: Must be easy for everyone to use.

Desirable: Image, identity, brand and other design elements are used to evoke emotion and appreciation.



More UX Core Principles

Findable: Content needs to be navigable and locatable on and offsite.

Accessible: Should be accessible to people with disabilities.

Credible: Users need to trust and believe what you tell them.



UX Tools

Moqups / Invision



UX Process



UX Process

User Centered Design

Design based on understanding users, tasks, and environments.

Driven by user-centered evaluation.

Addresses the whole user experience.



UX Process

1. User Research - Know and understand users.
2. Analysis - Identify design opportunities.
3. Design - Conceptualize and explore.
4. Prototype - Implement ideas and receive feedback.
5. User testing - Evaluate.



UX Process: User Research

What do your users want to accomplish?

What are their goals?

Who are the users?



Types of User Research

- Online Surveys
- Interviews
- Focus Groups
- Analytics



UX Process: Analysis

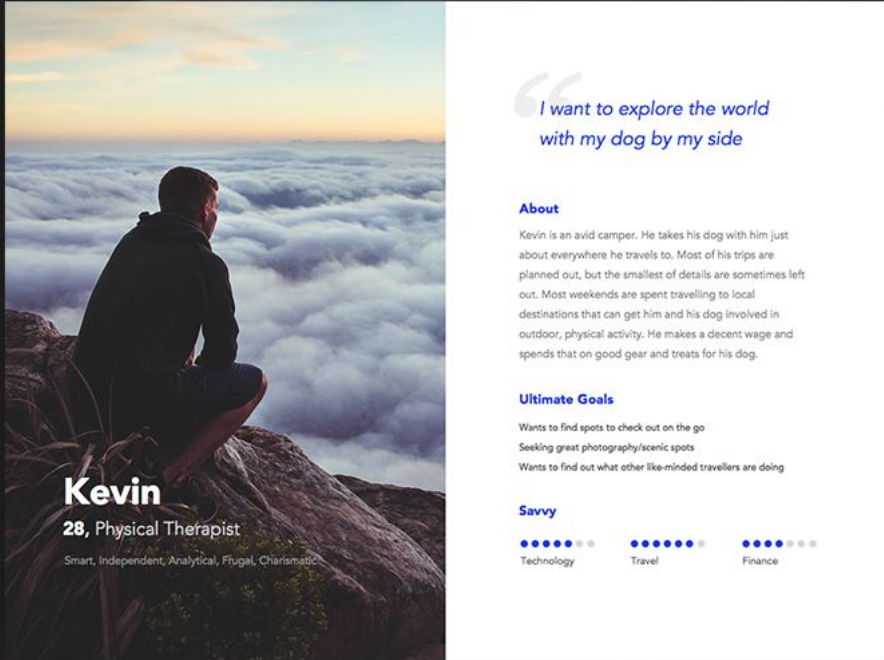
Analyzing user research to create user personas

“The purpose of personas is to create reliable and realistic representations of your key audience segments for reference”.

(usability.gov)



User Persona Example



The user persona card for Kevin is presented on a dark background. On the left, a photograph shows a man sitting on a rocky peak, looking out over a vast sea of clouds under a sunset sky. The text 'Kevin' is prominently displayed in white, followed by '28, Physical Therapist' and a list of traits: 'Smart, Independent, Analytical, Frugal, Charismatic'. On the right, a quote in blue script reads 'I want to explore the world with my dog by my side'. Below this, the 'About' section describes Kevin as an avid camper who takes his dog everywhere. The 'Ultimate Goals' section lists three aspirations: finding spots to check out, seeking great photography/scenic spots, and finding out what other like-minded travellers are doing. The 'Savvy' section features three progress bars for Technology, Travel, and Finance, each with five dots, where the first four dots are filled blue and the last is grey.

Kevin
28, Physical Therapist
Smart, Independent, Analytical, Frugal, Charismatic

"I want to explore the world with my dog by my side"

About
Kevin is an avid camper. He takes his dog with him just about everywhere he travels to. Most of his trips are planned out, but the smallest of details are sometimes left out. Most weekends are spent travelling to local destinations that can get him and his dog involved in outdoor, physical activity. He makes a decent wage and spends that on good gear and treats for his dog.

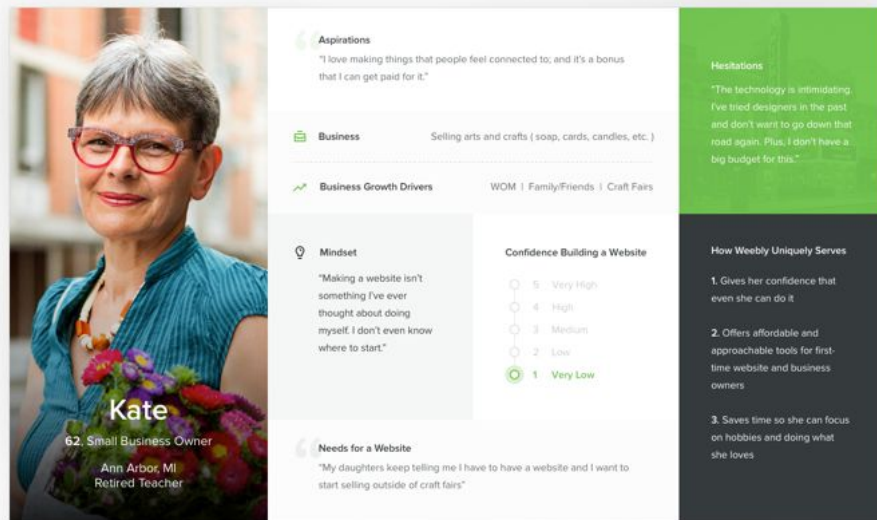
Ultimate Goals
Wants to find spots to check out on the go
Seeking great photography/scenic spots
Wants to find out what other like-minded travellers are doing

Savvy

Technology: ●●●●●○
Travel: ●●●●●○
Finance: ●●●●●○



User Persona Example



A user persona card for Kate, a 62-year-old small business owner in Ann Arbor, MI, who is a retired teacher. The card is divided into several sections: Aspirations, Business, Business Growth Drivers, Mindset, Confidence Building a Website, Needs for a Website, Hesitations, and How Weebly Uniquely Serves. Kate is shown in a photo on the left, holding a bouquet of flowers.

Kate
62, Small Business Owner
Ann Arbor, MI
Retired Teacher

Aspirations
"I love making things that people feel connected to, and it's a bonus that I can get paid for it."

Business
Selling arts and crafts (soap, cards, candles, etc.)

Business Growth Drivers
WOM | Family/Friends | Craft Fairs

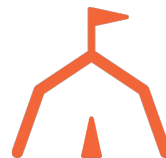
Mindset
"Making a website isn't something I've ever thought about doing myself. I don't even know where to start."

Confidence Building a Website
5 - Very High
4 - High
3 - Medium
2 - Low
1 - Very Low

Needs for a Website
"My daughters keep telling me I have to have a website and I want to start selling outside of craft fairs"

Hesitations
"The technology is intimidating. I've tried designers in the past and don't want to go down that road again. Plus, I don't have a big budget for this."

How Weebly Uniquely Serves
1. Gives her confidence that even she can do it
2. Offers affordable and approachable tools for first-time website and business owners
3. Saves time so she can focus on hobbies and doing what she loves



User Needs

Identify why users will use your product.
What do they want to accomplish? What
benefit will it provide?



User Needs ("Why" not "How")

This is NOT what *features* they need (e.g. blog, calculator, search).

This is what *outcomes* they need (e.g. find what hours the shop is open, identify pros & cons of different shipping options)



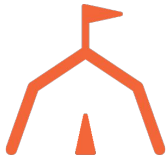
UX Process: Design

Sketching

Creating wireframes

User Flow

Feedback



UX Process: Prototyping

Paper Prototyping (Lo-Fi)

Digital Prototyping (Hi-Fi)

[Prototyping Resource](#)



Prototyping

Digital Prototyping



UX Process: User Testing

Analytics (web traffic and sales analytics)

Test design

Offer solutions



UX Activity

A client wants to create a product to help dog owners find playmates for their pet.

What are some possible user needs for this product the client has not thought of?

Split into groups and brainstorm at least six possible user needs on post-it notes.



UX Activity

Prioritize the eight most essential user needs that should be addressed throughout the product.



Wireframing



Wireframe

A wireframe is a skeleton of what we want our site to look like.

Think of it as the blueprint or sketch of the layout the site should include.

These are handy for us developers to use so we can visualize how sites are going to look before we start to code them.



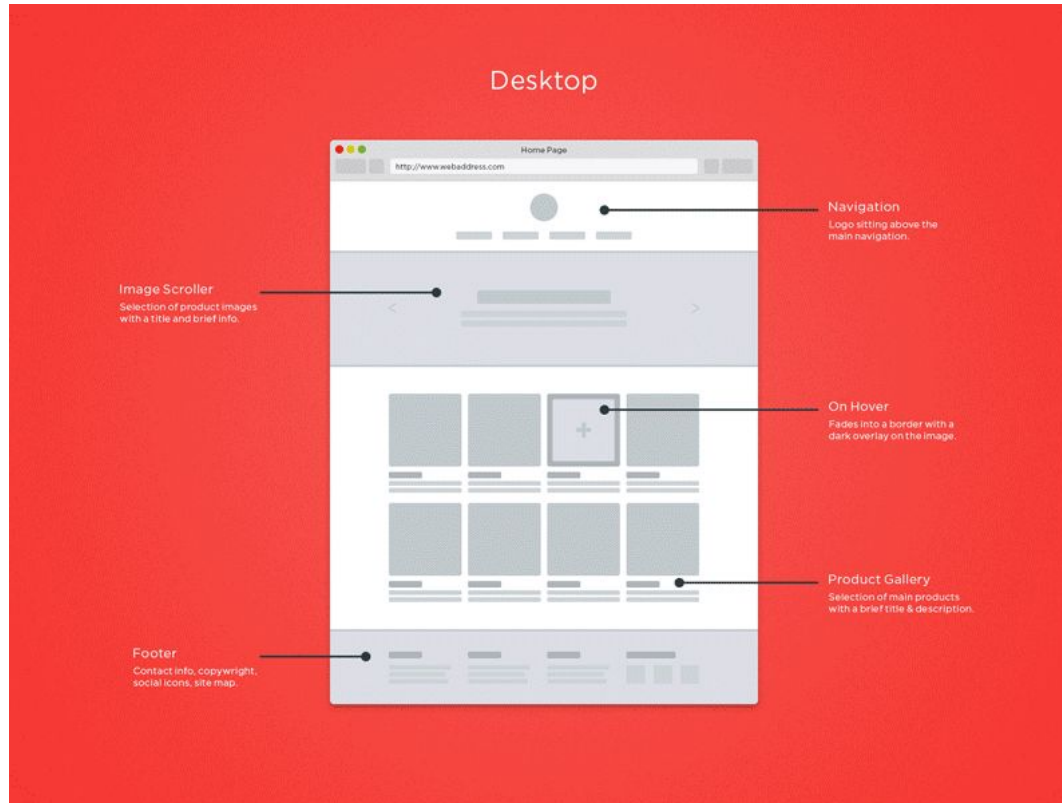
Wireframe

We use wireframes so we can plan and architect our code before we even touch the keyboard.

Strategizing this way helps to eliminate future problems by thinking through possible issues in the beginning.



Wireframe Example



Wireframe Activity

We are responsible for the creation of an event planning website.

Remember, we need a wireframe for mobile, tablet, and desktop.

What are some interactive features we can include?



Wireframe Activity

Let's go back to our dog matching site.

Based on our priorities, let's come up with a mobile and desktop wireframe.

Each group should take 10 minutes to sketch their interpretation of the home page structure based on user needs.

