

# FWDP 1000 – Day 5

Course: Web Development 1  
Instructor: Martha Villa Martin

# Morning Review

- Download the files and open them in your code editor.
- Read the comments in the CSS file and complete the tasks.
- If you finish these tasks, open **products.css** from the **day-05** folder and compare it to your version from day 4.

We will go over these in 15 minutes.

# Agenda

- Connecting to GitHub
- HTML Forms
- Custom Google Fonts
- Styling Best Practices
- Flexbox Extra
- Lab Time

# Connecting to GitHub

# From Local to Online

GitHub is a web-based platform that hosts Git repositories and offers many collaborative features.

Today, we'll cover the basics of setting up your GitHub account and pushing your local repositories online.

Open the PDF “connect-to-GitHub.pdf”

# Practice, practice, practice

From now on, our workflow will be to:

- Adding, commit, and pushing changes

It may be confusing and intimidating now, but the more you do it, the easier it gets!

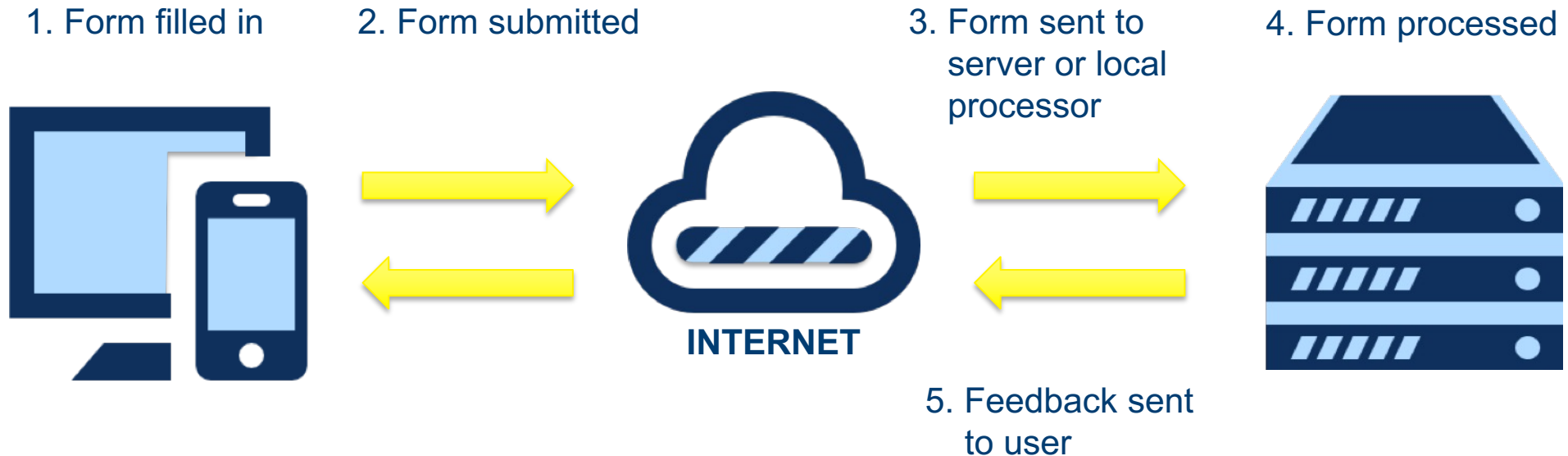
# HTML Forms

# What is a form?

- Forms allow web sites **users** to **interact** with your web site
- The form tag wraps form elements and enables the **sending of data** inputted into a form to a **web server**
- You can use forms to do the following:
  - Collect data from users
  - Logging into a site
  - To perform calculations
  - To contact you
  - To pay for items online
  - To search for something online



# How a Form Works



All images are from [iconmonstr.com](https://iconmonstr.com)

# How a Form Works

1. User fills in the form
2. User submits the form
3. Data from the form is sent to the form processor which is a server script that runs on a web server
  - Not all forms are sent to the server, some forms such as calculators can be processed locally and never require data to be sent to a server
4. The form is processed
5. Feedback can optionally be given to the user to assure them that their information has been received

# Form Tag (syntax)

The form tag wraps form elements and enables the sending of data inputted into a form to a web server

The action attributes tells the browser where to send the form data

The method attribute determines how the data is sent to the server.

The name attribute provides a way to identify the form to the server.

```
<form action="processor.php" method="post" name="contact">
  <!-- Form elements go here... -->
</form>
```

# Form Tag

- The form tag can include the following attributes:
  - **action** - The location of the form processing file
    - Form processing is usually done on the server side (to be covered in PHP)
    - Not all forms require submission to a server
  - **method** - Determines how the data is sent from the client to the server
    - The values are **GET** (useful for search forms) and **POST** (the most common method for most forms).
  - **name** - Provides a way to identify the form to the server

# Should I use GET or POST?

- Setting the method attribute to “get” means the form data will be sent out in the open in the URL address
  - Do not use “get” for sensitive form data like credit cards, usernames, emails, passwords and other personal information
  - Use “get” for search forms if you want to allow the user to bookmark a search query
- Use “post” for most forms
  - Post data is stored in the request body of the HTTP request<sup>1</sup>

<sup>1</sup> [https://www.w3schools.com/tags/ref\\_httpmethods.asp](https://www.w3schools.com/tags/ref_httpmethods.asp)

# Fieldset Tag

- The fieldset tag creates an element that **optionally** groups form controls and labels under a common name or caption.
- It nests a <legend> that captions the <fieldset>.
- Draws a box around the grouped controls and labels.

Choose your starter Pokemon

- ☐ Squirtle
- ☐ Charmander
- ☐ Bulbasaur

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/fieldset>

# Input Tag

- The input tag creates an element that can be used to input data by a user of a web site
- The input tag can have several “types” which determine the type of data that should be entered into the input element
- Some input types:
  - checkbox, email, tel, radio, text, password, number, button and submit

## Sign in

BCIT email

Password

**SIGN IN**

[Forgot your BCIT email](#) or [password?](#)

# Input Tag Attributes

- Some common input tag attributes include:
  - **type** - Determines the input type
  - **name** - Identifies the input for the server
    - Used by radio and checkbox input types to group them together
  - **disabled** - Used to disable an input and to not submit any value to the server
  - **required** - Used to tell the browser to require and validate that this input has been filled in

[Read more about input tag attributes](#)



# Input Tag Syntax

The “type” attribute determines the type of input

Used to identify the input to the server. It is also used radio and checkbox input types to group them together

```
<input type="text" id="firstname" name="fn">
```

The “id” attribute is used to link a label to an input

# Label Tag

- The label tag provides a label for an input element
  - The label tag can be linked to a form element by using the “for” attribute
- ! The “for” attribute value should match the “id” attribute value of the input element that it is a label of

[Read more about input tag attributes](#)

# Label Tag Syntax

```
<label for="firstname">Firstname: </label>  
<input id="firstname" type="text" name="fn">
```

The “for” attribute value must match the “id” of the input element that it is a label of

# Input Type - Password

- Used for password inputs
- Text entered by the user is hidden visually by the browser

```
<label for="pw">Password: </label>  
<input id="pw" type="password" name="pw">
```

Password: .....

# Textarea Tag

- The “textarea” tag is used for inputs where you need to give the user the ability to enter multiple lines of text or a large amount of text.
- Useful for comments

```
<label for="comments">Comments: </label>  
<textarea id="comments" name="comments"></textarea>
```

# Input Type - Submit

- An input with the “type” of “submit” is used to submit the form data
- When the “submit” input is clicked on the browser will send the form data to the location specified by the “action” attribute on the “form” element
- The “value” attribute determines what text appears in the button

```
<input type="submit" value="Submit">
```

# Input Type - Checkbox

- Allows the user to select multiple options with a "checkbox" interface
- Use "checkbox" when you want to give the user the ability to select **multiple options**
- ⚠ To create a "checkbox" group make sure the each "checkbox" input has the same "name" attribute value

# Input Type – Checkbox

To create a “checkbox” group make sure the each “checkbox” input has the same “name” attribute value

```
<input type="checkbox" name="a-features" id="b-pro" value="balance protection">  
<label for="b-pro">Balance Protection </label>  
  
<input type="checkbox" name="a-features" id="a-alerts" value="account alerts">  
<label for="a-alerts">Account Alerts </label>
```

- ☐ Balance Protection
- ☐ Account Alerts



# Input Type - Radio

- Allows the user to select one option from a group of options with a "radio" button interface
- Use "radio" when you want to give the user the ability to **select only one option** from a group of options
- ⚠ To create a "radio" group make sure the each "radio" input has the same "name" attribute value

# Input Type - Radio

To create a “radio” group make sure the each “radio” input has the same “name” attribute value

```
<input type="radio" name="account-type" id="chequing" value="chequing">  
<label for="chequing">Chequing </label>  
  
<input type="radio" name="account-type" id="savings" value="savings">  
<label for="savings">Savings </label>
```

- ☐ Chequing
- ☐ Savings

# Input Type - Hidden

- An input with the type of “hidden” will not be displayed visually by the browser
- Hidden inputs form data is sent to the server for processing
- Useful to send form data to a server that you do not need or want the user to enter into a form

```
<input type="hidden" name="user-id" value="abc12345">
```

# Select List

- Creates a select list form element
- Can be configured to allow the user to select only one option or multiple options
- The size attribute allows you to configure the number of options that are visible to the user before they activate the dropdown
- ⚠ Will look different depending on the browser and the operating system
- ⚠ Difficult to fully customize the appearance, but some styling is possible

# Select List

Select list in the closed state (Chrome Browser, macOS)

Doctor: Dr. Smith ▼

```
<label for="doctor">Doctor: </label>
<select id="doctor" name="doctor">
  <option value="Smith">Dr. Smith</option>
  <option value="Chou">Dr. Chou</option>
  <option value="Abbas">Dr. Abbas</option>
  <option value="Garcia">Dr. Garcia</option>
  <option value="Nair">Dr. Nair</option>
</select>
```

Doctor: ✓ Dr. Smith  
Dr. Chou  
Dr. Abbas  
Dr. Garcia  
Dr. Nair

Select list in the open state  
(Chrome Browser, macOS)

# Focus State

- One use of the “:focus” pseudo selector is style form elements when a user clicks into or focuses into a form input element
- Changing the style of the input that the user has focused into helps inform the user where they are in the form

[Read more about pseudo classes](#)

## Book an Appointment

### Your Info

First name

Michael

Last name

Email

Submit

This input has additional styles applied when the input is focused using the “:focus” pseudo selector

# Focus State Syntax

```
input[type="text"]:focus,  
input[type="email"]:focus {  
  background-color: #fdfdd6;  
  outline: none;  
  box-shadow: 0 0 1px 2px #13b613;  
  border-radius: 3px;  
}
```

The “:focus” pseudo selector applies when a user clicks into or focuses into an input element

# Placeholder

- The placeholder attribute should be used to provide a short hint (a word or short phrase) intended to aid the use with data entry when the control has no value<sup>1</sup>
- Use the placeholder attribute with caution (see next slide)
- The placeholder attribute is NOT a substitute for label elements (see next slide)
- Give the below linked article a read to learn about some of the issues with the placeholder attribute
  - <https://www.smashingmagazine.com/2018/06/placeholder-attribute/>



# Placeholder Attribute

```
<label for="fn">First name</label>  
<input type="text" id="fn" name="fn" placeholder="First name...">
```

First name

First name...

# Placeholder Attribute as Label



**Note:** Hiding labels and using placeholders inside form input elements as labels should be used with caution. Read the Nielsen Norman Group article [Placeholders in Form Fields Are Harmful](#) for reasons to perhaps not use this technique.

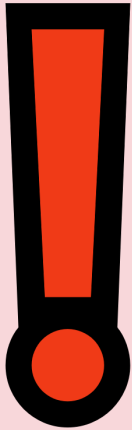
# Placeholder Attribute as Label



**Warning:** Never omit label tags in your markup, even if you are using this placeholder as label technique. People using screen readers rely on labels to use forms properly.

**Placeholders are NOT a substitute for labels in your HTML.** Use CSS to visually hide your labels (DO NOT use `display: none;`). Use a visually hidden style such as that used by WebAim "sr-only" class. See the next slide for a CSS code snippet that will allow you visually hide an element but still make the element accessible to screen readers.

# Placeholder Attribute as Label



**DO NOT BE A LAZY DEVELOPER!!!** Always use label tags and tie them to their corresponding inputs with a ***for*** attribute on the label tag that matches the ***id*** attribute on the corresponding input element, even if your labels are not visible.

# WebAIM sr-only Class


You can also use this code snippet from WebAIM to visually hide screen reader only elements.

WebAIM is an excellent resource for accessibility tools and tips.

**I will share a more robust sr-only style on day 9.**

Code snippet from this article:

<https://webaim.org/techniques/css/invisiblecontent/>



```
.sr-only {  
  position: absolute;  
  width: 1px;  
  height: 1px;  
  top: auto;  
  left: -10000px;  
  overflow: hidden;  
  border: 0;  
}
```

# Styling the Placeholder Attribute

Styling the Placeholder can be tricky as different browsers support a slightly different syntax for the “::placeholder” pseudo selector

First name




First name...

HTML input element with a placeholder attribute styled to with a red text color

# Styling the Placeholder Attribute


CSS syntax for styling the placeholder attribute for Chrome, Firefox and Chromium based Microsoft Edge



```
input::placeholder {  
  color: #d84f69;  
  /* Firefox fades the text using opacity  
     – If you want a consistent look between  
        browsers, set the opacity to 1  
  */  
  opacity: 1;  
}
```

# Styling the Placeholder Attribute

CSS syntax for styling the placeholder attribute  
for non-Chromium based Microsoft Edge




```
input::-ms-input-placeholder {  
  color: #d84f69;  
}
```



# Styling the Placeholder Attribute

CSS syntax for styling the placeholder attribute  
for IE 10+



```
input:-ms-input-placeholder {  
  color: #d84f69;  
}
```

# HTML Form Validation

- HTML5 introduced simple HTML form validation without any need for JavaScript
- You can use the "required" attribute on an input to make sure the user enters data on an input element
- If you want more custom looking form validation then you will need to turn to JavaScript
- **Note:** HTML form validation using the built-in HTML validation or using custom JavaScript validation is not secure. Always validate the data a 2<sup>nd</sup> time on the server when receiving data from an HTML form

# HTML Form Validation

```
<label for="city">City</label>  
<input type="text" id="city"  
name="city" required>
```



### Shipping Info

#### Your Address

Street Address

City

Province

Newfoundland and Labrador ▾

Postal

Please fill out this field.

Submit

# Form Validation with the Pattern Attribute

- Using the required attribute alone allows for basic validation to make sure the user at least inputted something into an input
- If you require something a bit more custom, you can use the “pattern” attribute to make the browser validate an input against a regular expression
  - A **regular expression** is a pattern that the computer uses to test a string of text against
- With a pattern attribute you can validate for Credit Card numbers, postal codes, phone numbers and other common types of data
- For some pre-written HTML5 pattern values visit this web site: <http://html5pattern.com/>
- Use the title attribute to provide helper text to the user if they input invalid data

# Form Validation with the Pattern Attribute

The pattern attribute tells the browser to test the input entered by the user against the regular expression set in the pattern attribute. The pattern here will test for a Canadian postal code (A1A 1A1)

```
<input type="text"
       id="postal-code"
       name="postal-code"
       pattern="[A-Za-z][0-9][A-Za-z](\s)?[0-9][A-Za-z][0-9]"
       title="Format: A1A 1A1"
       required>
```

You can use the title attribute to display a helper message to the user if they enter incorrectly formatted data

# Form Validation with the Pattern Attribute

Postal Code

ABC 123



Please match the requested format.

Format: A1A 1A1

This input is being validated against a custom pattern set via the “pattern” attribute on the input element. This input is being validated for a Canadian postal code (A1A 1A1).

- Visit <http://html5pattern.com/> to get some common pattern values for things such as telephone numbers, credit cards, postal codes and other common types of information

The text “Format: A1A 1A1” comes from the title attribute on the input element

# Recap

Which **method** would you use for a user login form?

GET      POST

**TRUE or FALSE:** You can omit the label tag when you use a placeholder in your input.

# Recap

What is wrong with this code snippet?

```
<input type="checkbox" name="dogs" id="dogs" value="dogs">  
<label for="dogs">Dogs</label>  
  
<input type="checkbox" name="cats" id="cats" value="cats">  
<label for="cats">Cats</label>
```



# Custom Fonts

# Web Font File Formats

- **WOFF2**: The best format. It works in the latest versions of all major browsers.
- **WOFF**: It works in older versions of all major browsers and Internet Explorer.
- **TTF** or **OTF**: These are not optimized for the web but can work in most browsers.
- **SVG**: Used by Safari on old iPhones.
- **EOT**: Used by old Internet Explorer.

# Variable Fonts

A newer version of fonts are called **Variable Fonts**.

Variable fonts will include all weights in a single file.

Variable fonts typically only require two font files:

- Normal styles (containing all weights)
- Italic styles (containing all weights)

# Font Licenses

When choosing custom fonts, make sure you are following the license!

Some fonts, like those hosted by Google, are free to use on any website.

Other fonts require you to purchase the fonts or a license to use the fonts.

# Hosted Fonts

**Google Fonts (Free)** - <https://fonts.google.com/>

- You can download the fonts or let Google host them.
- Google is blocked in China so if you have users in China you should use downloaded fonts.

**Adobe Fonts (Paid)** - <https://fonts.adobe.com/>

- Included with Creative Cloud Subscription.

# Hosted Fonts Syntax

The online font resource you use will provide you a `<link>` element to add inside of your `<head>` element.

## Adobe

To use these fonts on a web page, copy this code into the `<head>` tag of your HTML.

```
<link rel="stylesheet" href="https://use.typekit.net/hah3ecb.css">
```

If you'd like to use fonts in HTML email, use the [@import link](#).

## Google

Use on the web

To embed a font, copy the code into the `<head>` of your html

☒ `<link>` ☐ `@import`

```
<link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap" rel="stylesheet">
```

CSS rules to specify families

```
font-family: 'Roboto', sans-serif;
```

# Hosted Font Practice

Let's use Google Fonts to add a font to our About Us page.

Playfair Display [Download family](#)

Select styles Glyphs About License Pairings

Regular 400  
Almost before we knew it, we had left the ground. — Remove this style

Regular 400 italic  
*Almost before we knew it, we had left the ground.* + Select this style

Medium 500  
Almost before we knew it, we had left the ground. + Select this style

Medium 500 italic  
*Almost before we knew it, we had left the ground.* + Select this style

Semi-bold 600  
Almost before we knew it, we had left the ground. + Select this style

Semi-bold 600 italic  
*Almost before we knew it, we had left the ground.* + Select this style

Bold 700  
Almost before we knew it, we had left the ground. — Remove this style

Bold 700 italic  
*Almost before we knew it, we had left the ground.* + Select this style

Selected family

Review

Playfair Display ^

Regular 400 ⊖

Bold 700 ⊖

[Add more styles](#) [Remove all](#)

Use on the web

To embed a font, copy the code into the <head> of your html

☒ @link ☐ @import

```
<link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;700&display=swap" rel="stylesheet">
```

CSS rules to specify families

```
font-family: 'Playfair Display', serif;
```

[API docs](#) [Download all](#)


# Using the Custom Font

Once you have imported a custom font, you can use the `font-family` you defined.


```
body {  
    font-family: 'Roboto', Arial, sans-serif;  
}
```



The primary font from Google Fonts included in your CSS file.



A fallback font. Define as many as you want or none.



A fallback font type. It's good to always have one here.



# Recap

What is the optimal web font file format?

WOFF      TTF or OTF      SVG      WOFF2      EOT

If using an externally hosted font, which is the optimal way to load the font?

- a) Using a `<link>` element in the HTML.
- b) Using an `@import` rule in the CSS.

# Forms and Font Group Activity

- Collaborate as a team and complete the steps listed in the day-05-group-activity PDF.
- Aside from the predefined max-width, the form does not have to be colour and pixel-perfect.

This is an optional exercise.

The solution will be provided to you on Day 6, but if you want me to check your work, let me know.

# Styling Best Practices

# Number of Fonts

General rule of thumb is to have 1 or 2 fonts per site.

Many designers use one font for headings and one font for everything else.

## Our Story

We are a local company based out of *beautiful* Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within

# Decorative Fonts

Decorative fonts and very stylized fonts should never be used for paragraphs of text.

This is okay.

## *Our Story*

We are a local company based out of *beautiful* Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within

This is bad.

## **Our Story**

*We are a local company based out of beautiful Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within the framework of this experience we can provide the best results for our customers so that they can reach the potential they truly desire. We never turn down a job! No job is too small and every job is too big for us to complete so we will never let you down with a medium to infinite sized job.*

# Line Lengths

Line length should be roughly 45 to 85 characters per line.

- MANY sites break this rule, look at news sites for better examples.

Screen width and font size will determine this.

Use the **max-width** property instead of setting a fixed **width**.

# Line Length Example

Smashing Magazine uses a max-width for its content area making it easy to read...

## THE IDEAL MEASURE: 45 TO 75 CHARACTERS

We have multiple “rules” for facilitating a horizontal reading motion, one of which is to set text at a reasonable measure. As James Craig wrote in his book *Designing With Type* (originally published in 1971, now it its fifth edition):

*Reading a long line of type causes fatigue: the reader must move his head at the end of each line and search for the beginning of the next line.... Too short a line breaks up words or phrases that are generally read as a unit.*

If a casual reader gets tired of reading a long horizontal line, then they’re more likely to skim the left edge of the text. If an engaged reader gets tired of reading a long horizontal line, then they’re more likely to accidentally read the same line of text twice (a phenomenon known as “doubling”).

# Line Length Example

Wikipedia does not use max-width for its content area making it more difficult to read...

## Web accessibility <sup>[ edit ]</sup>

See also: [Web accessibility](#)

## International standards and guidelines <sup>[ edit ]</sup>

There have been a few major movements to coordinate a set of guidelines for accessibility for the web. The first and most well known is The [Web Accessibility Initiative](#) (WAI), which is part of the World Wide Web Consortium (W3C). This organization developed the [Web Content Accessibility Guidelines](#) (WCAG) 1.0 and 2.0 which explain how to make Web content accessible to everyone, including people with disabilities. Web "content" generally refers to the information in a Web page or Web application, including text, images, forms, and sounds. (More specific definitions are available in the WCAG documents.<sup>[69]</sup>

The WCAG is separated into three levels of compliance, A, AA and AAA. Each level requires a stricter set of conformance guidelines, such as different versions of HTML (Transitional vs Strict) and other techniques that need to be incorporated into coding before accomplishing validation. Online tools allow users to submit their website and automatically run it through the WCAG guidelines and produce a report, stating whether or not they conform to each level of compliance. [Adobe Dreamweaver](#) also offers plugins which allow web developers to test these guidelines on their work from within the program.

The [ISO/IEC JTC1 SC36 WG7 24751](#) Individualized Adaptability and Accessibility in e-learning, education and training series is freely available and made of 3 parts: Individualized Adaptability and Accessibility in e-learning, education and training, Standards inventory and Guidance on user needs mapping.

Another source of web accessibility guidance comes from the [US government](#). In response to [Section 508 of the US Rehabilitation Act](#), the Access Board developed standards to which U.S. federal agencies must comply in order to make their sites accessible. The U.S. General Services Administration has developed a website where one can take online training courses for free to learn about these rules.<sup>[70]</sup>

## Web accessibility features <sup>[ edit ]</sup>

Examples of accessibility features include:

- [WAI-AA compliance with the WAI's WCAG](#)
- [Semantic Web markup](#)
- [\(X\)HTML Validation from the W3C for the pages content](#)
- [CSS Validation from the W3C for the pages layout](#)
- [Compliance with all guidelines from Section 508 of the US Rehabilitation Act](#)
- [A high contrast version of the site for individuals with low vision, and a low contrast \(yellow or blue\) version of the site for individuals with dyslexia](#)
- [Alternative media for any multimedia used on the site \(video, flash, audio, etc.\)](#)
- [Simple and consistent navigation](#)
- [Device independent](#)



# Line Height

Give your text space between lines with **line-height**.

Start with a line-height of 1.5 and adjust from there.

line-height: 1;

## Our Story

We are a local company based out of *beautiful* Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within the framework of this experience we can provide the best results for our customers so that they can reach the potential they truly desire. We **never** turn down a job! No job is too small and every job is too big for us to complete so we will never let you down with a medium to infinite sized job.

line-height: 1.6;

## Our Story

We are a local company based out of *beautiful* Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within the framework of this experience we can provide the best results for our customers so that they can reach the potential they truly desire. We **never** turn down a job! No job is too small and every job is too big for us to complete so we will never let you down with a medium to infinite sized job.

# Font Size

16px (or 1rem) is the default font size for browsers and you should rarely set your font-size lower than this.

Remember, you can always use media queries to change the font size on your site from mobile to desktop... or better yet, use CSS [clamp\(\)](#)

# CSS clamp()

The CSS clamp() function lets you set a preferred value with a minimum and maximum value. For example:

```
font-size: clamp(1rem, 2.5vw, 2rem);
```

It is great for fonts, but can be used for other values too!

<https://developer.mozilla.org/en-US/docs/Web/CSS/clamp>

<https://ishadeed.com/article/responsive-design/#fluid-sizing>

# Font Alignment

Don't overuse `text-align: center;` because it is harder to read.

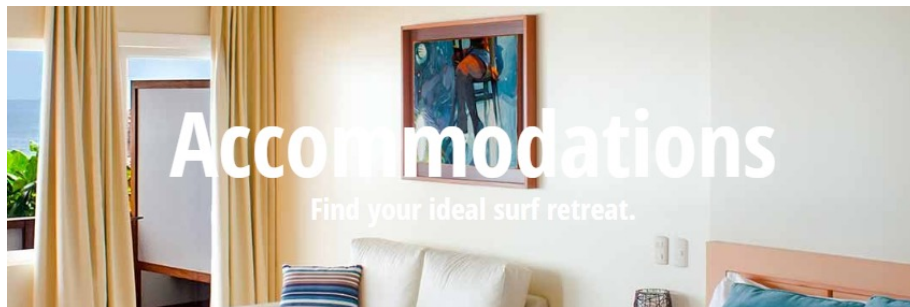
Align text to the left as a default, use center alignment for text that is only 1 or 2 lines long.

# Color Contrast

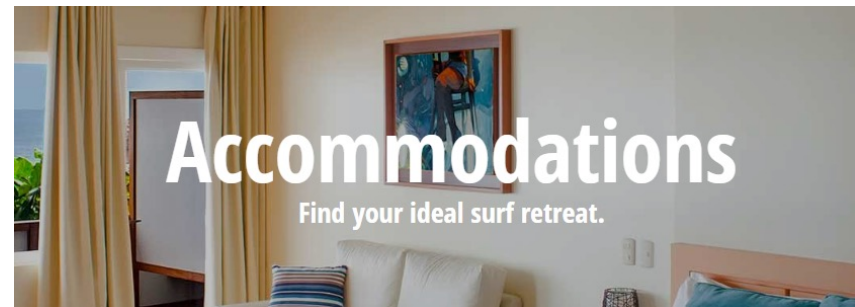
Make sure your font and background colors have sufficient contrast. <https://webaim.org/resources/contrastchecker/>

Follow the same philosophy with text over images:

This is bad.




This is okay.



# Side Padding

Make sure you have some padding next to your text for small screens.

No padding  
on the sides.



## Definitions of the SERP Features

There are [more SERP features beyond these three](#), and Google (and other web-wide search engines) constantly iterates on them. Google changes the design of its results page frequently, so the SERP-feature designs you see may differ from the designs covered here, depending on the time of reading.

There are two main reasons you should care about the design of these SERP features:

- **SEO** (search engine optimization) is a main source of qualified traffic for most websites. To succeed on the Internet, you must know how people use search and how they interact with SERP features, as shown by independent user research.
- **Search on your own website** (or across your intranet) will be heavily influenced by search on public search engines due to [Jakob's law of the Internet user experience](#), which says that people spend most of their time on websites other than yours. Specifically, users' [mental models](#) for how search and search features work will be shaped by their experience with Google (or with the dominant search engine in their country), so your own site-search features should cater to the expectations created by the big search engines.

This is better.



## Definitions of the SERP Features

There are [more SERP features beyond these three](#), and Google (and other web-wide search engines) constantly iterates on them. Google changes the design of its results page frequently, so the SERP-feature designs you see may differ from the designs covered here, depending on the time of reading.

There are two main reasons you should care about the design of these SERP features:

- **SEO** (search engine optimization) is a main source of qualified traffic for most websites. To succeed on the Internet, you must know how people use search and how they interact with SERP features, as shown by independent user research.
- **Search on your own website** (or across your intranet) will be heavily influenced by search on public search engines due to [Jakob's law of the Internet user experience](#), which says that people spend most of their time on websites other than yours. Specifically, users' [mental models](#) for how search and search features work will be shaped by their experience with Google (or with the dominant search engine in their country), so your own site-search features should cater to the expectations created by the big search engines.

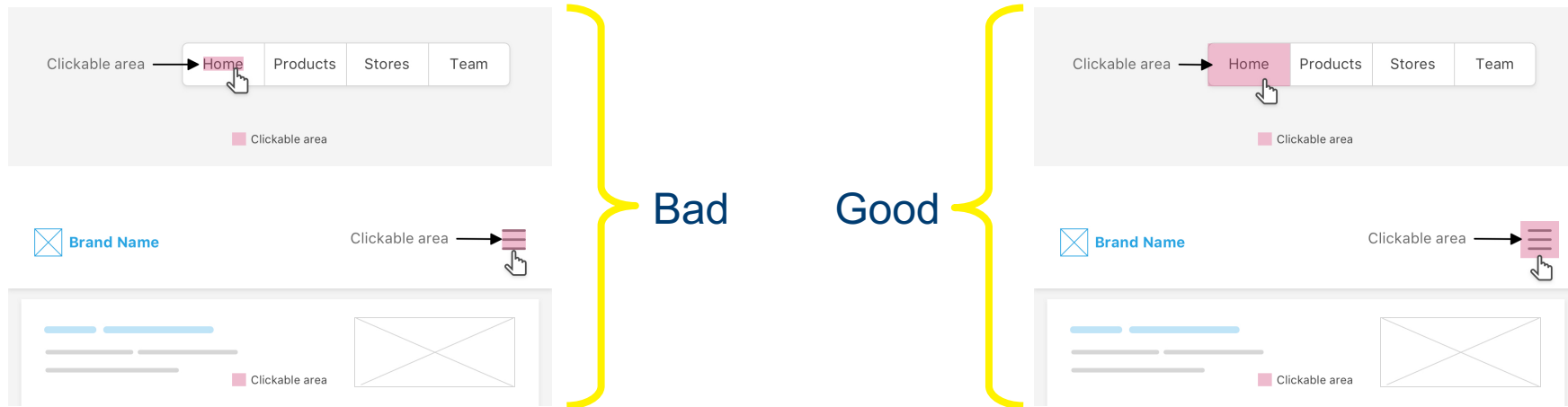
# Whitespace

Give your content space between other pieces of content by adding “whitespace” with margin and padding values.

For example, a margin between sections indicates to users that they are distinct from one another.

# Clickable Areas

Make clickable elements easier to click with padding.



<https://ishadeed.com/article/clickable-area/>



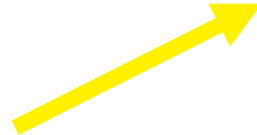
# Capitalization

Use **text-transform** to make text lowercase or uppercase.

If you type your content in all caps, screen readers can think it is an acronym and may read each individual letter.

```
<h1>About us</h1>
```

```
h1 {  
  text-transform: uppercase;  
}
```



**ABOUT US**

**Our Story**

We are a local company based out of *beautiful* Vancouver, British Columbia, Canada. It is our passion for creating that allows us to thrive and gives our customers the benefits of our boundless energy. Within

# Accessibility

A lot of these best practices have to do with making your site accessible to as many users as possible.

While working on your projects, start implementing these best practices that you have already learned.

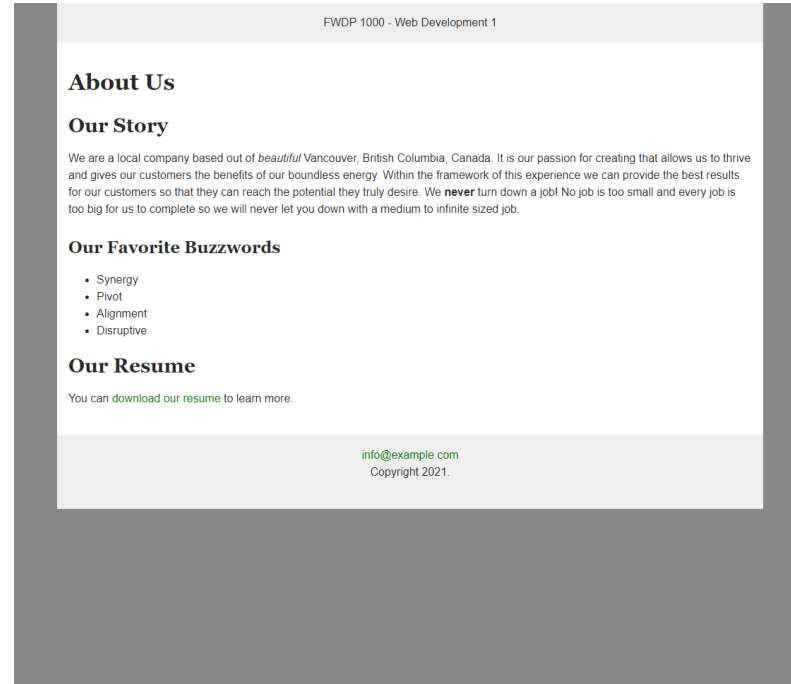
We will cover accessibility throughout the course but this is a great resource to get you started: <https://a11y-101.com/>

# Flexbox Extras

# Short Content = Short Height

Pages with very little content might be shorter than the viewport height.

Flexbox can make this look better though by using “flex-grow” property for full height.



# Viewport and Percentage Units

Using a percentage as a CSS value is based on the parent and grandparent and great-grandparent, etc.

```
main { width: 100%; height: 100%; }
```

Using the viewport units is based on the size of the viewport and not its ancestors. (This usually includes the scrollbar.)

```
main { width: 100vw; height: 100vh; }
```

# Viewport and Percentage Units

General rules...

- For width, min-width, max-width: % is better than vw
- For height, min-height, max-height: vh is better than %
- Avoid setting “height” on elements

<https://bitsofco.de/viewport-vs-percentage-units/>

# Flexbox and Images

Images that are flex items can become distorted if the other content in the flex container is taller.



Lorem ipsum dolor sit amet consectetur adipiscing elit. Voluptas repellendus perspicatis impedit qui odit libero quam laudantium nulla sed voluptate fugiat, quis ad at sint dignissimos assumenda iusto quas quasi. Lorem ipsum, dolor sit, amet consectetur adipiscing elit. Incidunt, pariatur ex ea sapiente ut nostrum deserunt quidem laboriosam perspicatis mollitia officiis dolorem cupiditate corporis soluta aut quia magnam ullam eos? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Animi deleniti hic voluptates, veritatis unde cum? Quaerat aut natus eius ipsa expedita, quam debitis iste ipsum, nam deserunt tempora ratione vitae. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla, labore? Nemo in facilis accusamus consectetur consequuntur, a quibusdam sed. Nobis accusamus rem debitis quidem, ad molestiae possimus facere fugit repellat. Lorem ipsum dolor sit amet consectetur adipiscing elit. Vel ea pariatur at quisquam corporis, laboriosam labore. Placeat voluptatum laboriosam maxime ipsam! Velit ducimus incididunt esse id laborum sit perferendis architecto! Lorem ipsum dolor, sit amet consectetur adipiscing, elit. In nihil cupiditate architecto adipisci nulla illum necessitatibus modi quia, doloribus laborum dicta exercitationem, cumque voluptatem vero molestiae ex odio quas inventore? Lorem, ipsum dolor sit, amet consectetur adipiscing elit. Voluptate atque, obcaecati odio quis assumenda vitae quas tenetur ex ratione blanditiis omnis quo similique molestias soluta enim tempora qui autem molestiae.

# Flexbox and Images

This can be resolved by...

- wrapping the `<img>` element in a `<div>` or `<figure>` element

...or...

- using `align-items` on the flex container

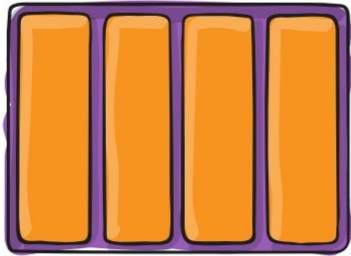


Lorem ipsum dolor sit amet consectetur adipiscing elit. Voluptas repellendus perspicatis impedit qui odit libero quam laudantium nulla sed voluptate fugiat, quis ad at sint dignissimos assumenda iusto quas quasi. Lorem ipsum, dolor sit, amet consectetur adipiscing elit. Incidunt, pariatur ex ea sapiente ut nostrum deserunt quidem laboriosam perspicatis mollitia officiis dolorem cupiditate corporis soluta aut quia magnam ullam eos? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Animi deleniti hic voluptates, veritatis unde cum? Quaerat aut natus eius ipsa expedita, quam debitis iste ipsum, nam deserunt tempora ratione vitae. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla, labore? Nemo in faciliis accusamus consectetur consequuntur, a quibusdam sed. Nobis accusamus rem debitis quidem, ad molestiae possimus facere fugit repellat. Lorem ipsum dolor sit amet consectetur adipiscing elit. Vel ea pariatur at quisquam corporis, laboriosam labore. Placeat voluptatum laboriosam maxime ipsam! Velit ducimus incidunt esse id laborum sit perferendis architecto! Lorem ipsum dolor, sit amet consectetur adipiscing, elit. In nihil cupiditate architecto adipisci nulla illum necessitatibus modi quia, doloribus laborum dicta exercitationem, cumque voluptatem vero molestiae ex odio quas inventore? Lorem, ipsum dolor sit, amet consectetur adipiscing elit. Voluptate atque, obcaecati odio quis assumenda vitae quas tenetur ex ratione blanditiis omnis quo similique molestias soluta enim tempora qui autem molestiae.

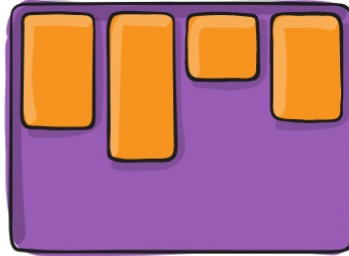


# Align Items Values

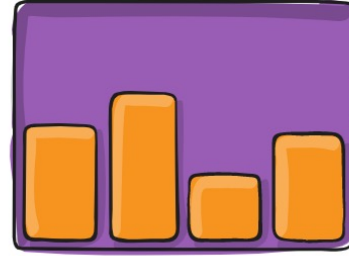
stretch



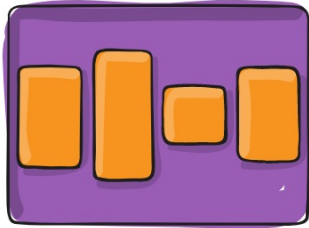
flex-start



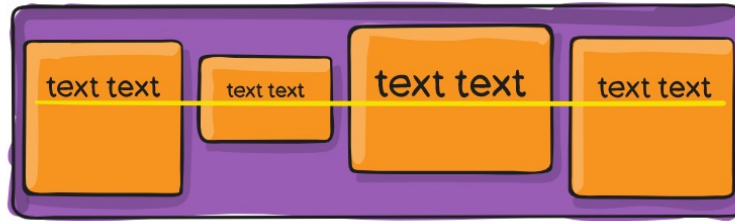
flex-end



center



baseline



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

# Lab Time

# Projects

No assignment today.

Use the rest of the day to work on the optional group activity and/or your Projects:

- Landing Page – Due May 10
- Country Website – Due May 26

# VS Code – Live Server

If you are using the Live Server extension in Visual Studio Code to view your files in the browser, incorrect file paths in your HTML code can still work.

To check your file paths are correct, open your HTML files directly at least once and check the console for any 404 errors.

# Peer Review

Before submitting your Landing Page, consider asking a classmate to look at the page and your code.

Do the same for them.

Give each other suggestions and see if you can find any issues with the code, design, functionality, accessibility, etc.

# Practice Git & GitHub

Practice adding and committing until it becomes a habit.  
Git/GitHub is incredibly helpful especially for more complex projects.

Practicing it even in simple projects make it less daunting!

# Resources

## Forms (MDN)

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form>

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input>

## Pseudo Classes (MDN)

<https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes>

# Resources

## Accessibility of Placeholder Attribute

<https://www.boia.org/blog/is-placeholder-text-essential-for-form-accessibility>

<https://webaim.org/techniques/css/invisiblecontent/>

## Fonts

<https://fonts.google.com/>

<https://fonts.adobe.com/>



---

# QUESTIONS & ANSWERS