Web Development 2

Day 1

Agenda

- Course overview
- HTML forms
- Project 1a
- Project 1b
- Lab time

Response Times for Technical Issues

- If contacting me on a class day, expect a response within a couple of hours
- If contacting me on a non-class day, expect a response within 48 hours between Monday and Friday
- If contacting me during the weekend or a holiday, expect a response on the next business day

Solving Technical Issues

- I am always available to help with any technical or code issues. With that said, I encourage you to do your best to solve problems on your own or with the help of of your classmates
- Steps for troubleshooting
  - Determine what the problem is
  - Check for any error reporting by your development software
    - VS Code -> look for strange colouring of the code or underlines
    - Browser -> Utilize developer tools…Do you notice any CSS being crossed out in dev tools…If the problem is JS based…then check the console
  - Google any error messages or just google the issue you are having
  - Consult your classmates
  - Again…I am also here to help if all else fails…

HTML Forms

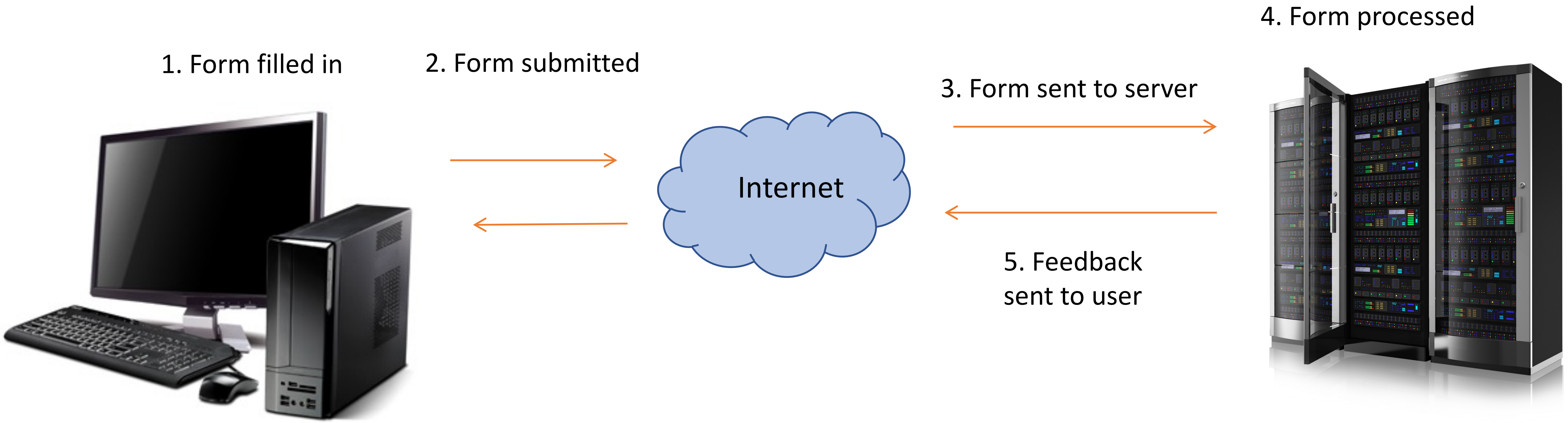
Forms - Introduction

- Forms allow web sites users to interact with your web site
- You can use forms to do the following:
  - Collect data from users
  - Logging into a site
  - To perform calculations
  - Enable users to contact you
  - To pay for items online
  - To search for something online

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

How a Form Works



1. Form filled in

2. Form submitted

3. Form sent to server

4. Form processed

5. Feedback sent to user

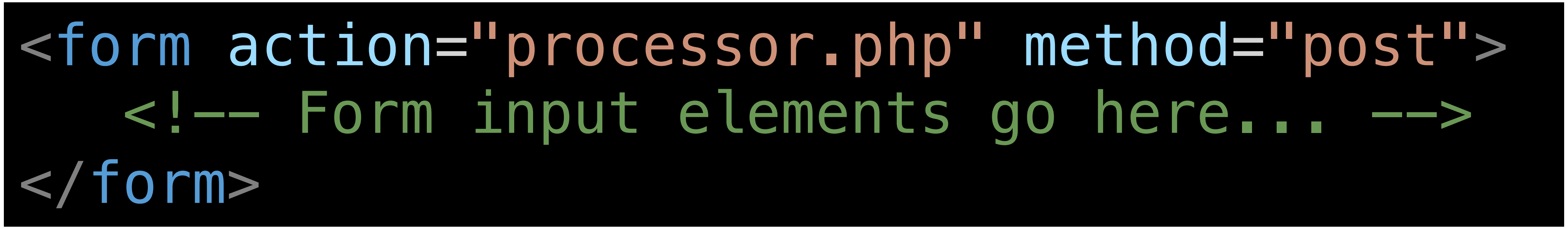
Form Tag

- The form tag wraps form elements and enables the sending of data inputted into a form to a web server
- 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
        - Form processing will 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)
        - post (the most common method for most forms)
  - name
    - Provides a way to identify the form to the server

Form Tag

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

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



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

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
    - For most forms you should use “post”
    - 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[1]

[1] https://www.w3schools.com/tags/ref_httpmethods.asp

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
  - Types of HTML and HTML5 inputs include but are not limited to the following:
    - checkbox, email, tel, radio, text, password, number, button and submit

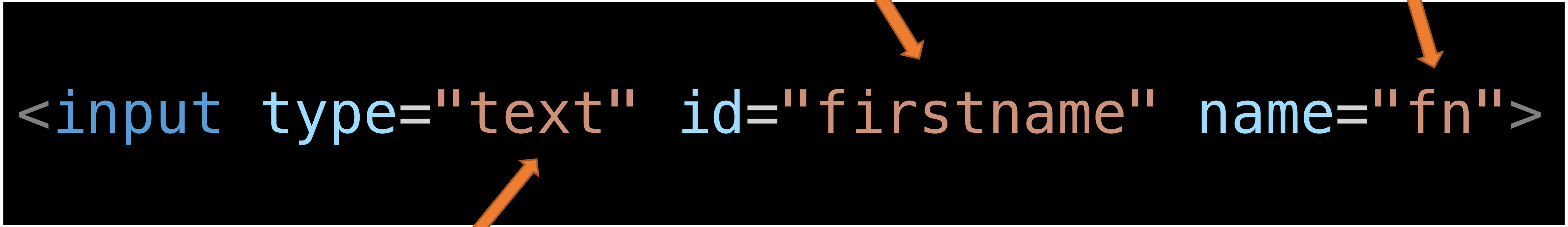
Input Tag

- 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
    - readonly
      - Used to make an input read only and to send a value to the server
    - required
      - Used to tell the browser to require and validate that this input has been filled in

Input Tag

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

The “name” attribute is used to identify the input to the server. The “name” attribute is also used radio and checkbox input types to group them together



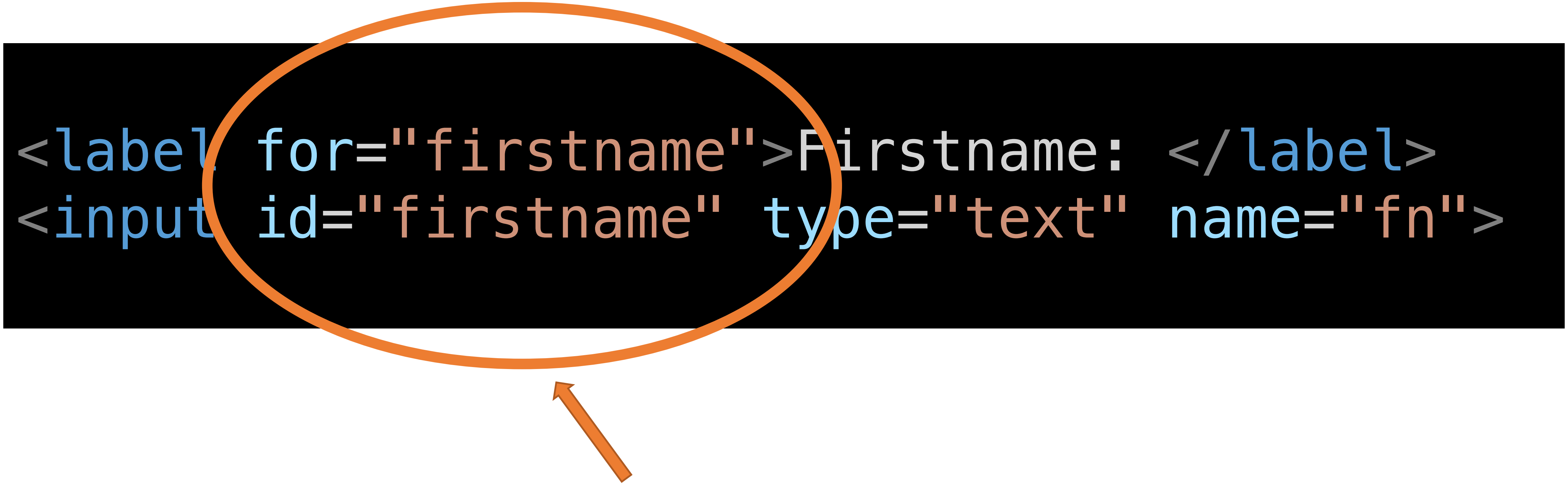
```
<input type="text" id="firstname" name="fn">
```

The “type” attribute determines the type of 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

Label Tag

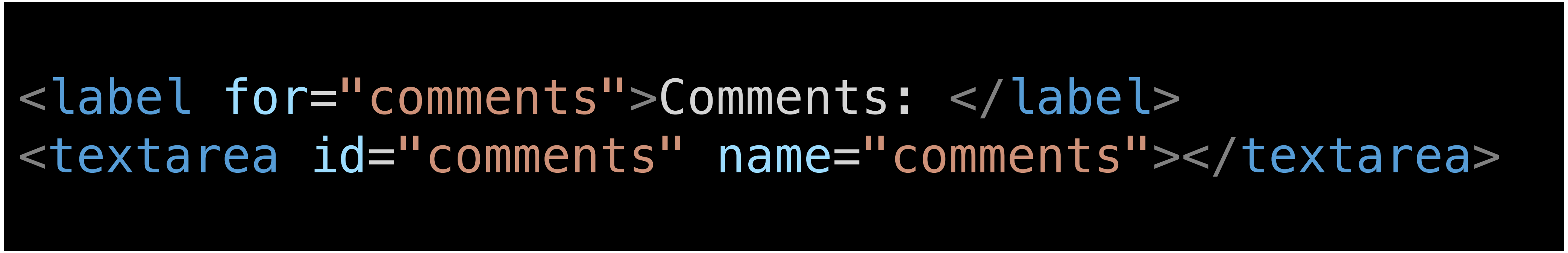


```
<label for="firstname">Firstname: </label>
<input id="firstname" type="text" name="fn">
```

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

Textarea Tag

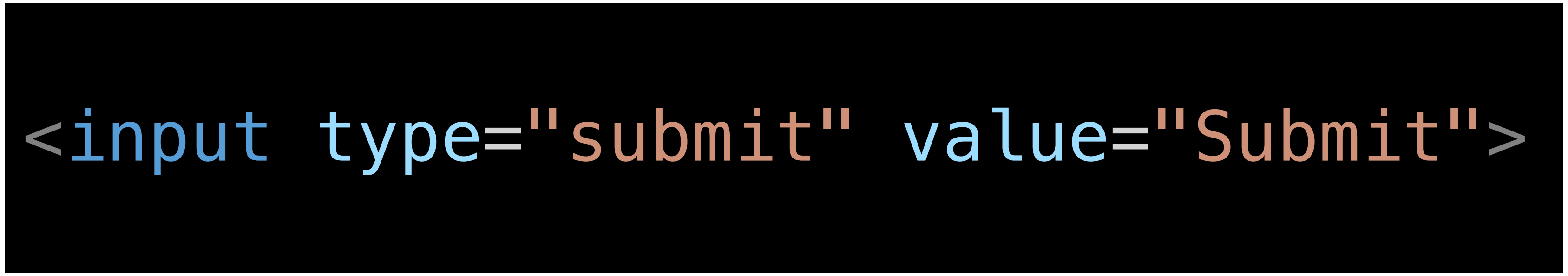
- The “textarea” tag is used for inputs where you need the give the user the ability to enter multiple lines of text or a large amount of text
- Useful for comments



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

Input Type - Submit

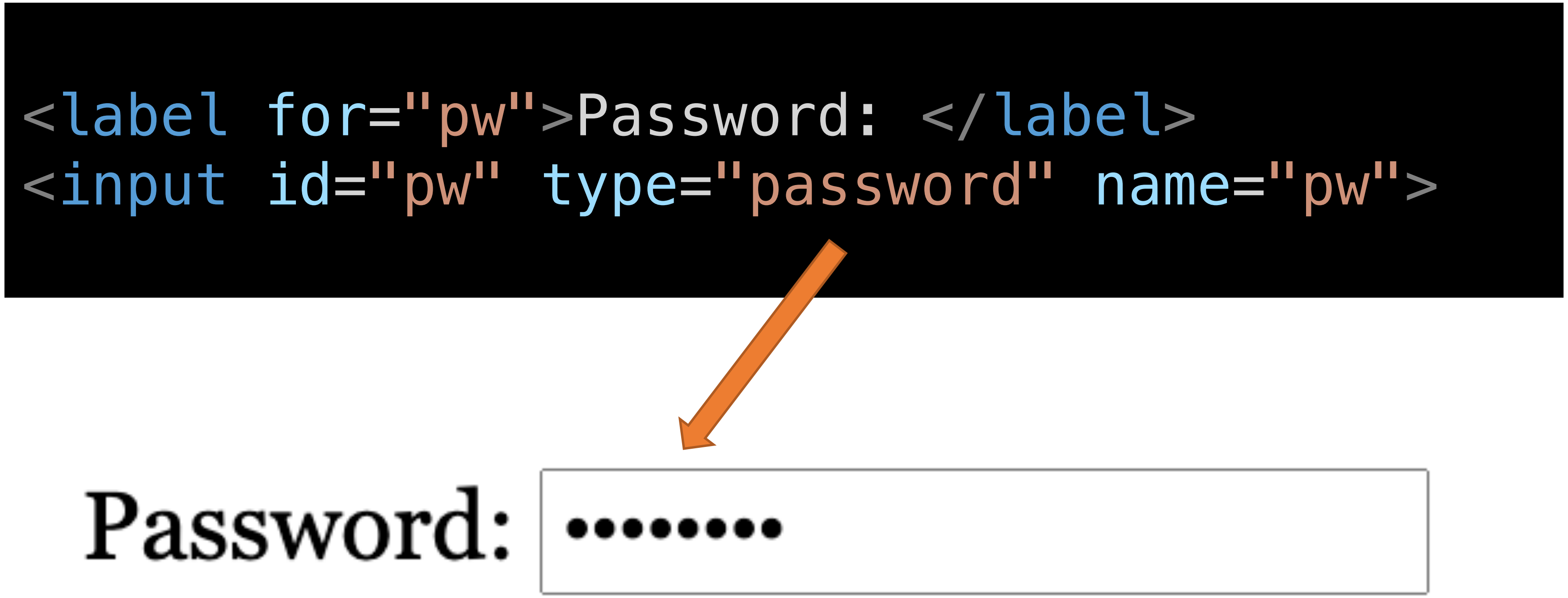
- An input with the “type” of “submit” is used to submit the form data
- When the “submit” input is clicked on the browser will send the form data to the location specified by the “action” element on the ”form” element
- The “value” attribute determines what text appears in the button



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

Input Type – Password

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



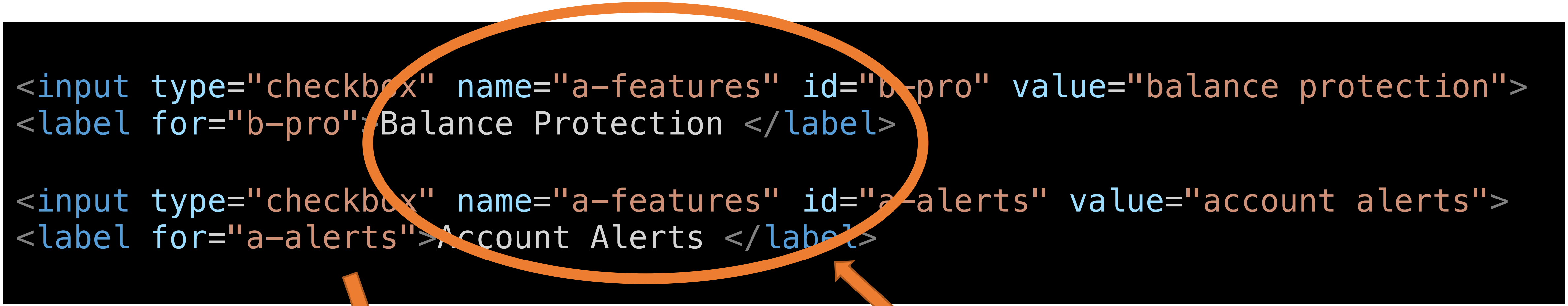
```
<label for="pw">Password: </label>
<input id="pw" type="password" name="pw">
```

Password: •••••••••

Input Type - Checkbox

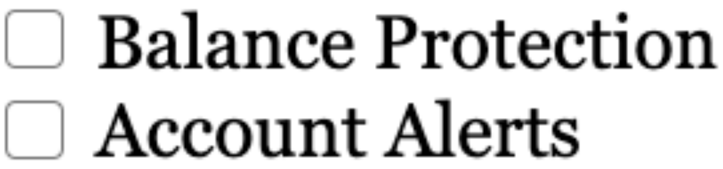
- Allows the user to select multiple options with a ”checkbox” interface
- Use “checkbox” when you want to give the user the ability to select multiple options
- To create a “checkbox” group make sure the each “checkbox” input has the same “name” attribute value

Input Type - Checkbox



```
<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

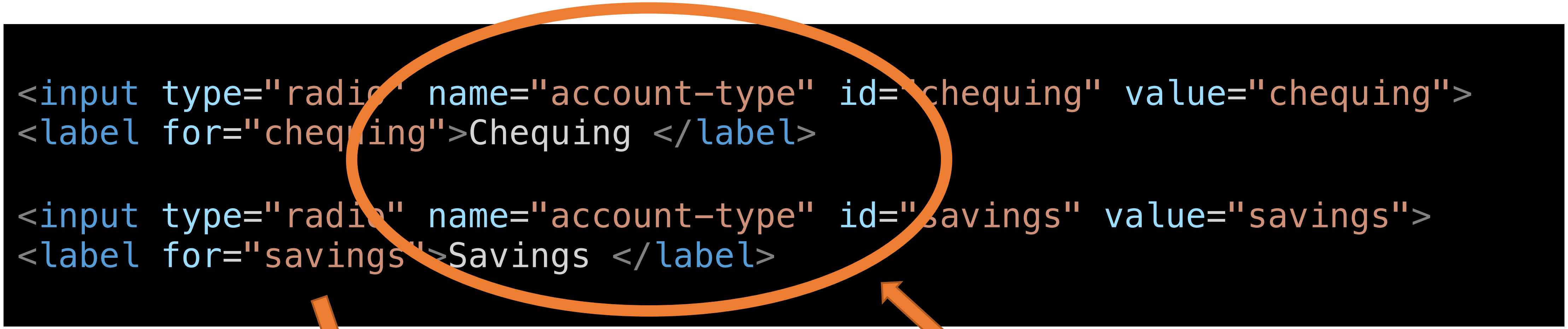
☐ Account Alerts

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

Input Type - Radio

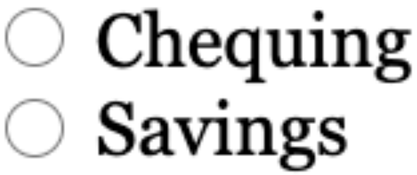
- Allows the user to select one option from a group of options with a ”radio” button interface
- Use “radio” when you want to give the user the ability to select only one option from a group of options
- To create a “radio” group make sure the each “radio” input has the same “name” attribute value

Input Type - Radio



```
<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>
```

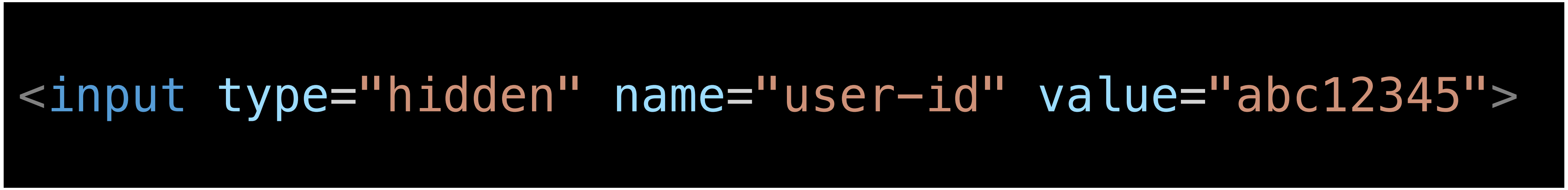


○ Chequing
○ Savings

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

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



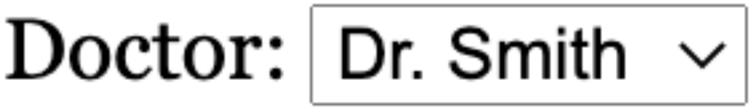
```html
<input type="hidden" name="user-id" value="abc12345">
```

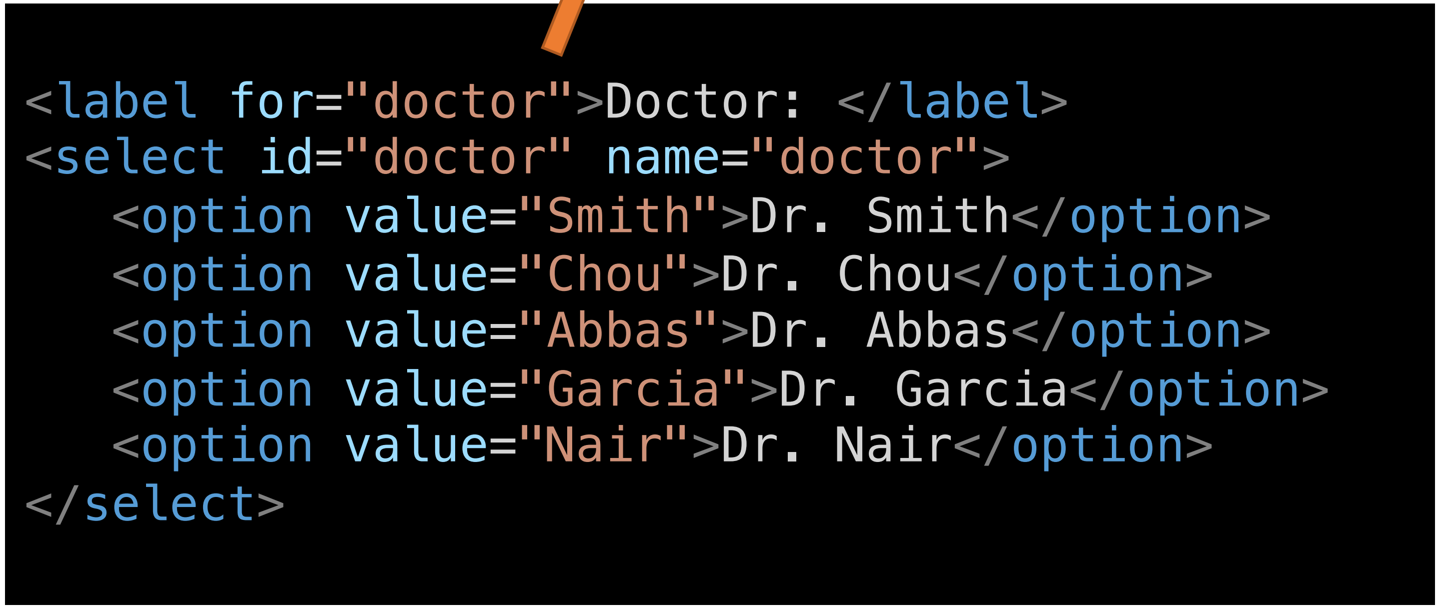
Select List

- 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 amount 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

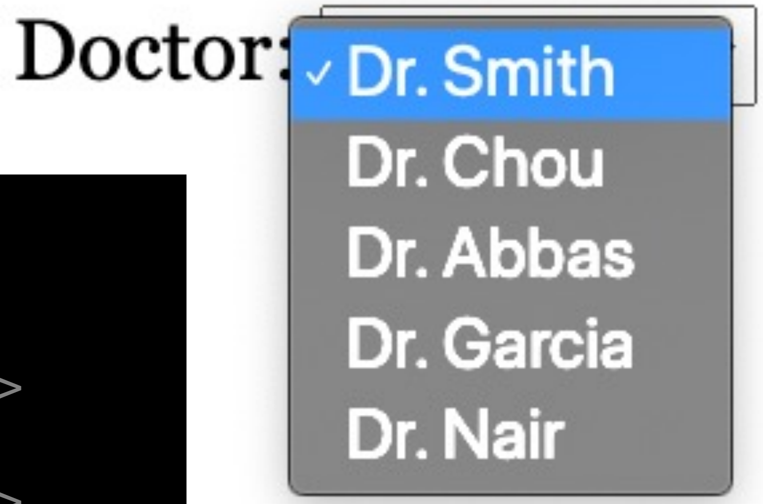
Select list in the closed state (Chrome Browser, macOS)





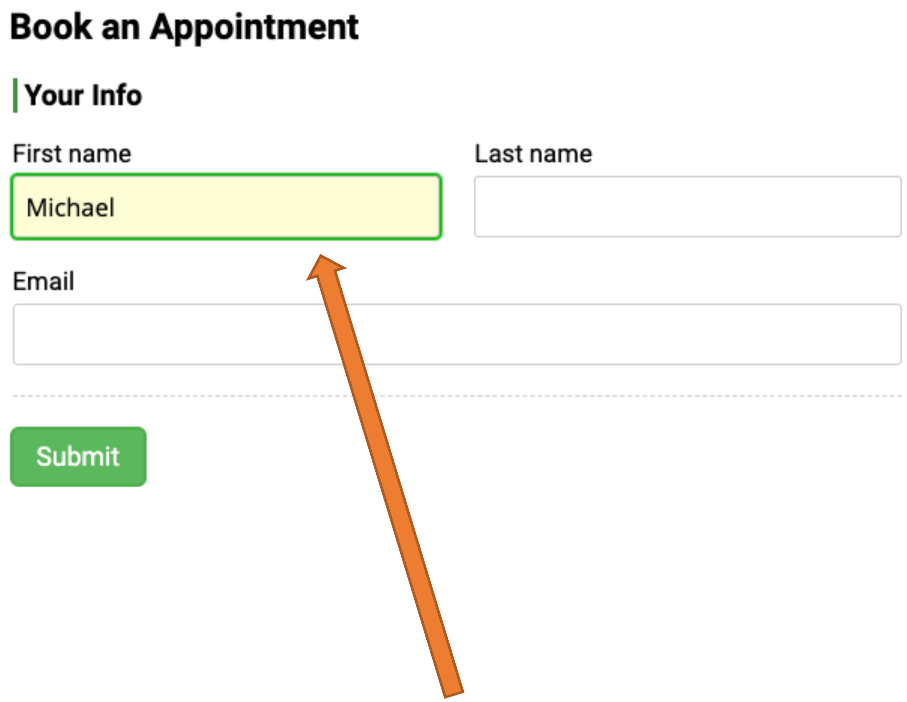
```
<label for="doctor">Doctor: </label>
<select id="doctor" name="doctor">
   <option value="Smith">Dr. Smith</option>
   <option value="Chou">Dr. Chou</option>
   <option value="Abbas">Dr. Abbas</option>
   <option value="Garcia">Dr. Garcia</option>
   <option value="Nair">Dr. Nair</option>
</select>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

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



Book an Appointment

Your Info

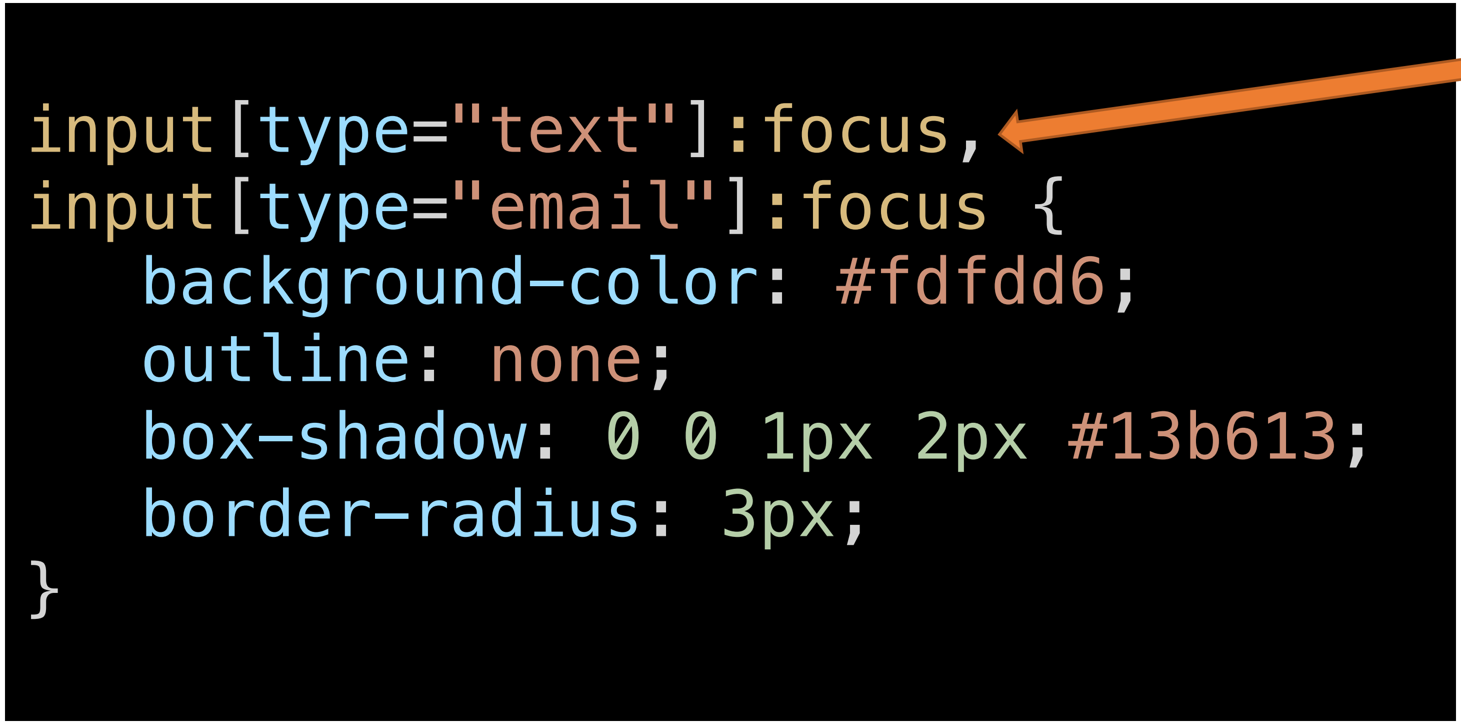
First name: Michael

Last name:

Email:

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

Focus State



```
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 Attribute

- 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[1]
- 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/

[1] https://www.smashingmagazine.com/2018/06/placeholder-attribute/

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.

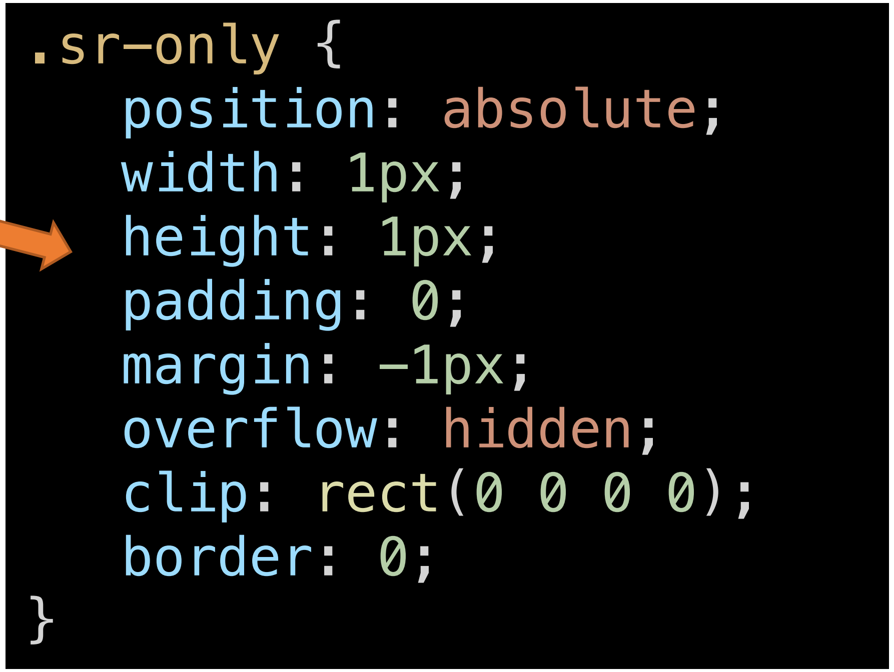
Placeholder Attribute as Label

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 Bootstrap's "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.

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.

Bootstrap 3 sr-only Class

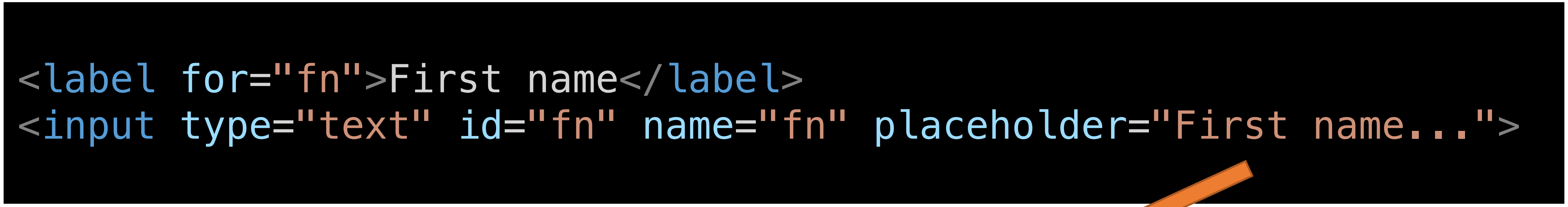
- Use this CSS code snippet from Bootstrap 3 to visually hide an element but still make it accessible to screen readers



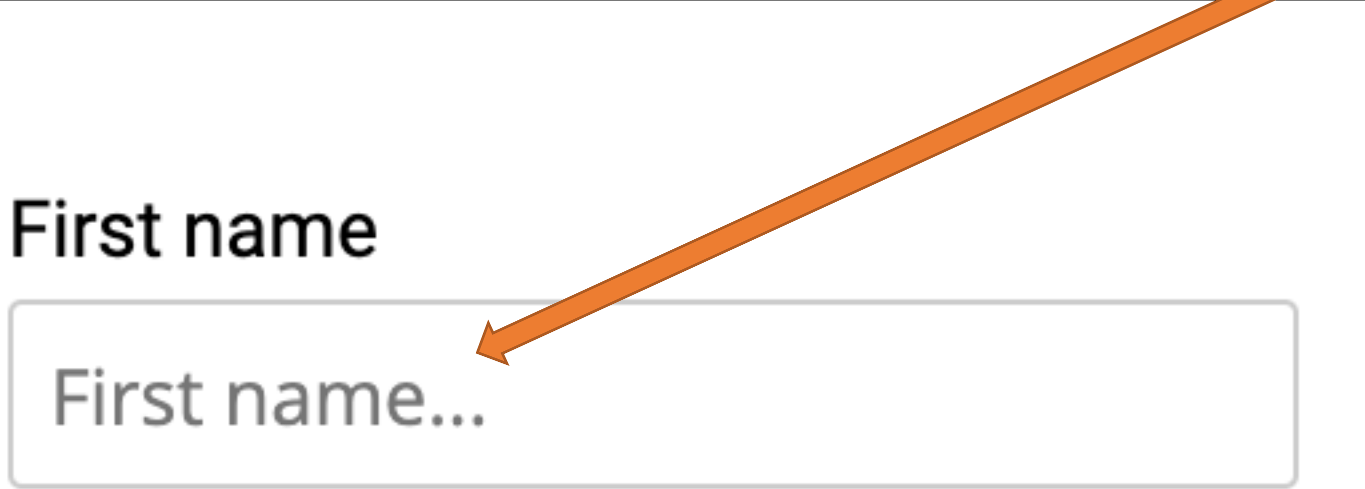
```
.sr-only {
   position: absolute;
   width: 1px;
   height: 1px;
   padding: 0;
   margin: -1px;
   overflow: hidden;
   clip: rect(0 0 0 0);
   border: 0;
}
```

Code snippet from this stackoverflow question and answer: https://stackoverflow.com/a/19340317

Placeholder Attribute

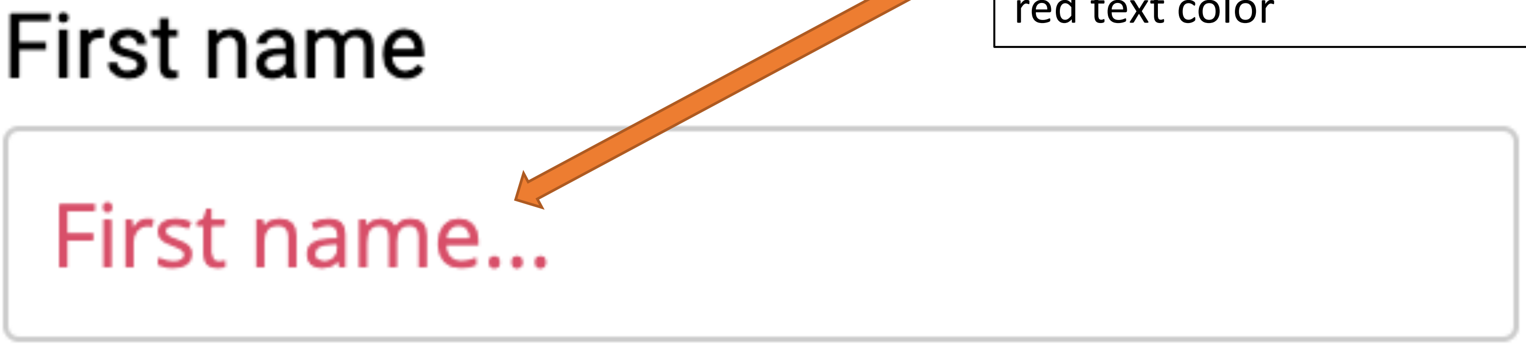


```html
<label for="fn">First name</label>
<input type="text" id="fn" name="fn" placeholder="First name...">
```



Styling the Placeholder Attribute

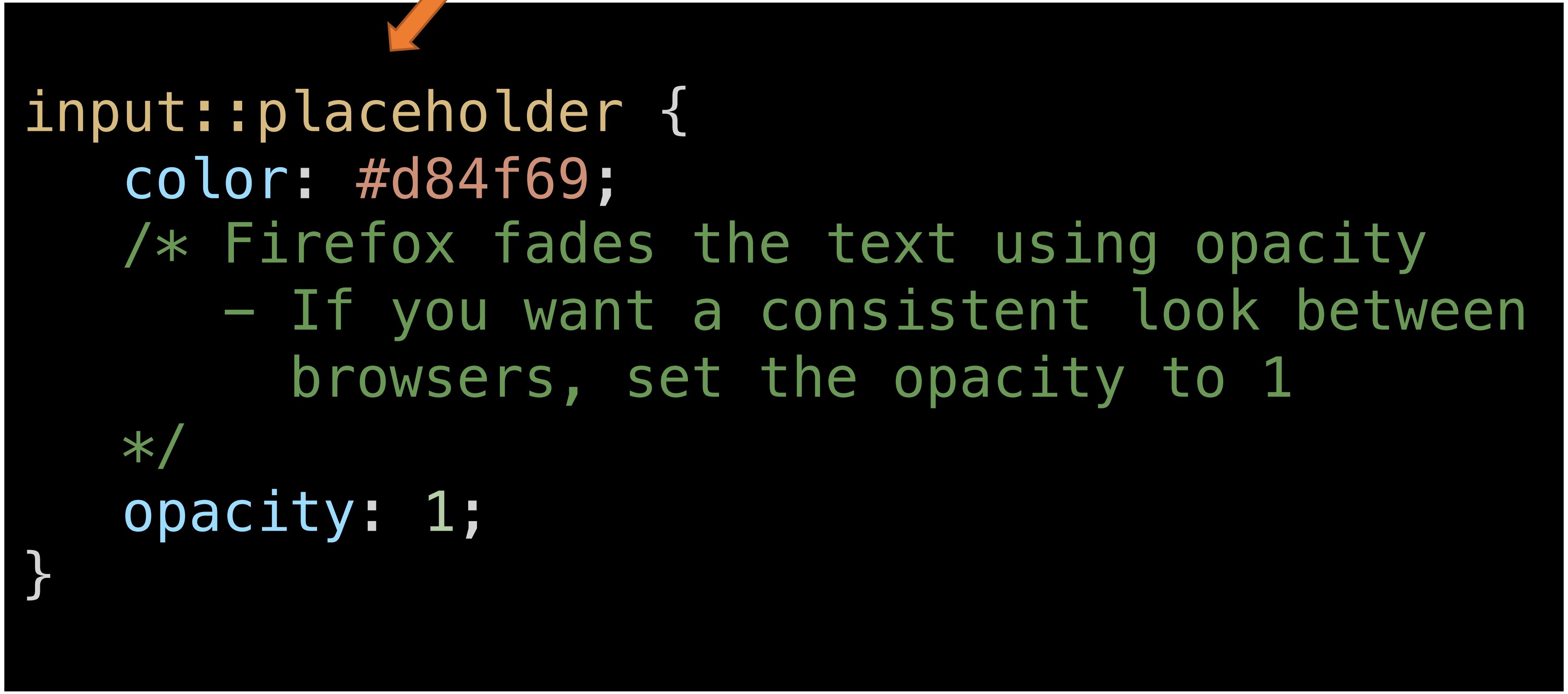
- Styling the Placeholder can be tricky as different browsers support a slightly different syntax for the "::placeholder" pseudo selector



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

Styling the Placeholder Attribute

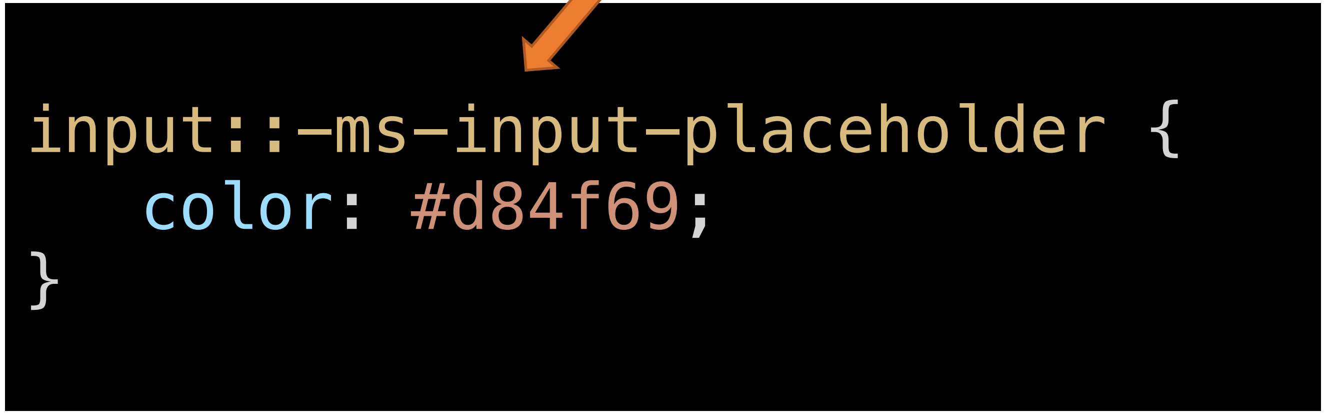
CSS syntax for styling the placeholder attribute for Chrome, Firefox and Chromium based Microsoft Edge



```
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

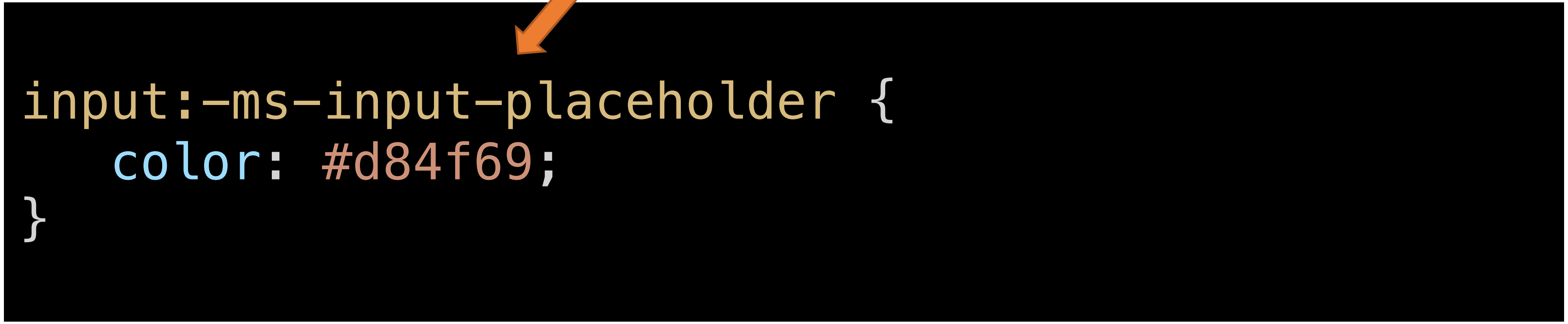
CSS syntax for styling the placeholder attribute for non-Chromium based Microsoft Edge



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

Styling the Placeholder Attribute

CSS syntax for styling the placeholder attribute for IE 10+



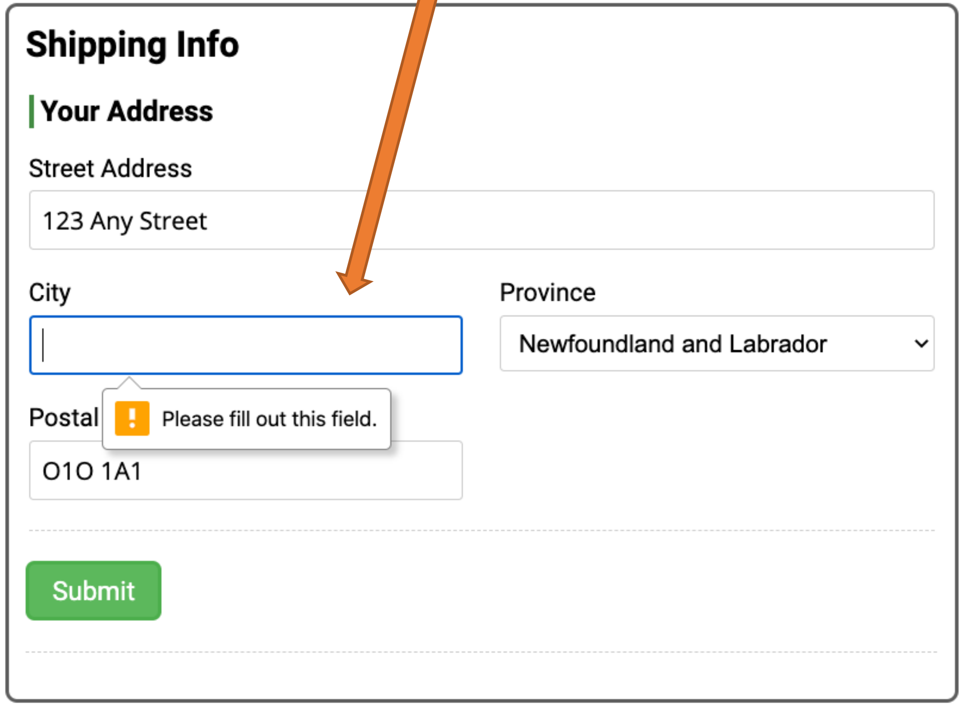
```
input:-ms-input-placeholder {
   color: #d84f69;
}
```

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 2nd 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: 123 Any Street

City:

Province: Newfoundland and Labrador

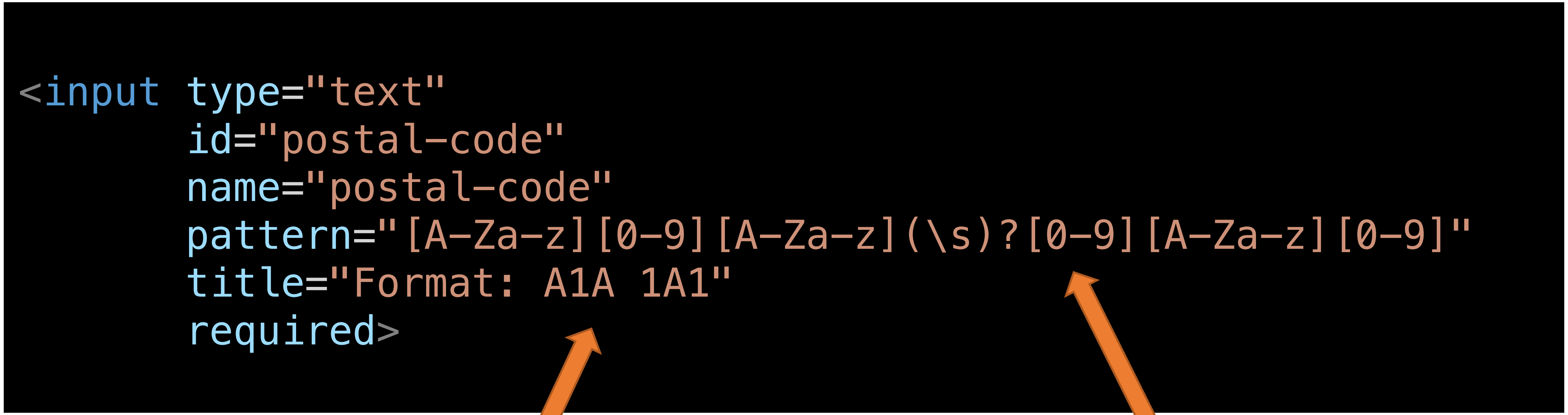
Postal: O1O 1A1

Please fill out this field.

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

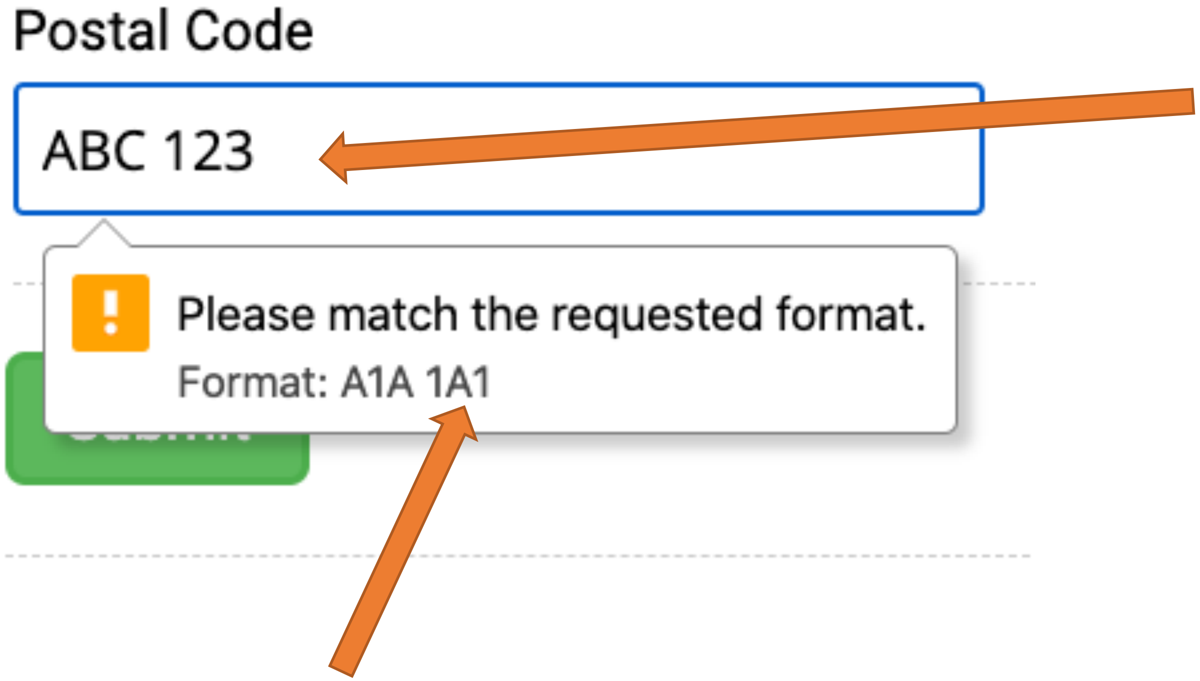


```
<input type="text"
       id="postal-code"
       name="postal-code"
       pattern="[A-Za-z][0-9][A-Za-z](\s)?[0-9][A-Za-z][0-9]"
       title="Format: A1A 1A1"
       required>
```

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

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)

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).

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

- 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

Project
1a and 1b

Project Overview

- Research a new web technology as part of a group and present what you have learned to the class
- Create a website that provides an overview of the technology that you researched
- This is a group project
  - The instructor will assign groups
  - Groups will be the same for both projects 1a, 1b, and 2
- Due December 4 – 6
  - Depends on the selected presentation date
- More details can be found on the instruction sheet for this project

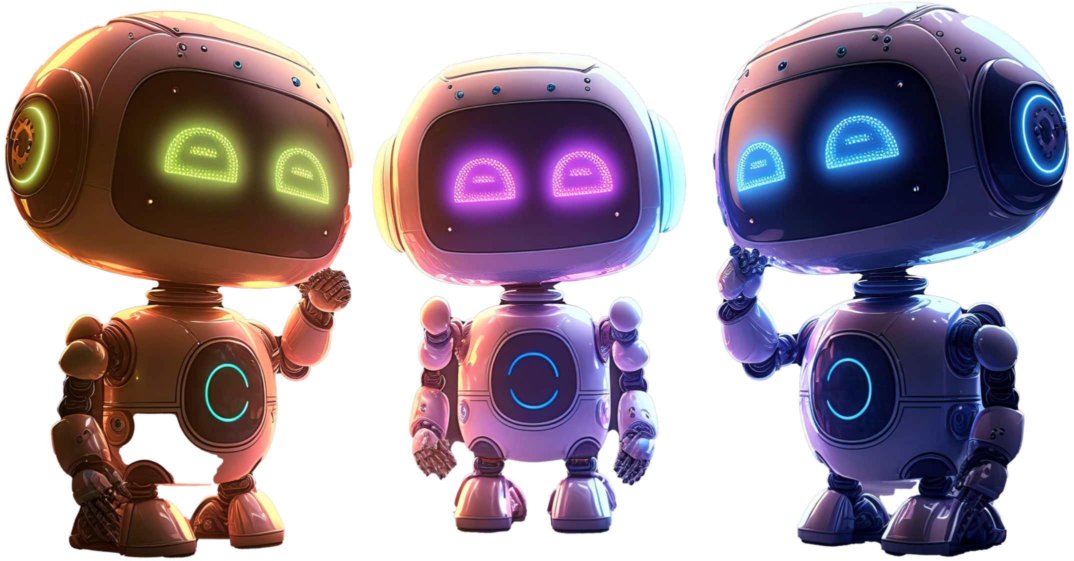
Project Objectives

- **Group Collaboration:** Learn the essentials of effective teamwork, contributing equitably to a shared goal.
- **Modern Web Tools:** Gain hands-on experience with contemporary web tools like Git for version control and Sass for CSS pre-processing.
- **Exploratory Research:** Investigate emerging trends or technologies in web development and summarize findings.
- **Knowledge Sharing:** Share acquired insights with the class via a presentation and website, fostering a collaborative learning environment.
- **Presentation Skills:** Develop the ability to convey technical information clearly and compellingly.
- **Communication Skills:** Build essential communication skills vital for succeeding in the professional world.

Note: The above points generated by ChatGPT from rough objectives provided by Michael Whyte

Groups

- Groups will be randomly selected in class
- Group size will be 4 – 5 students
- Groups will be the same for all projects for the Web Development 2 course



Benefits of Group Work

- **Diverse Perspectives:** Access to a variety of viewpoints that can lead to more comprehensive solutions.
- **Skill Enhancement:** Opportunity to learn from peers, thus improving one's skill set.
- **Resource Pooling:** Sharing of resources, be it time, expertise, or tools, to more efficiently achieve project goals.
- **Shared Responsibility:** Workload distribution among team members can lead to better time management and less stress.
- **Accountability:** Regular group meetings and peer evaluations can foster a sense of responsibility.
- **Conflict Resolution:** Learning to resolve disagreements constructively is a valuable workplace skill.
- **Social Skills:** Interaction within a group helps improve interpersonal skills, including communication and empathy.

Note: The above edited points were originally generated by ChatGPT

Topic Selection

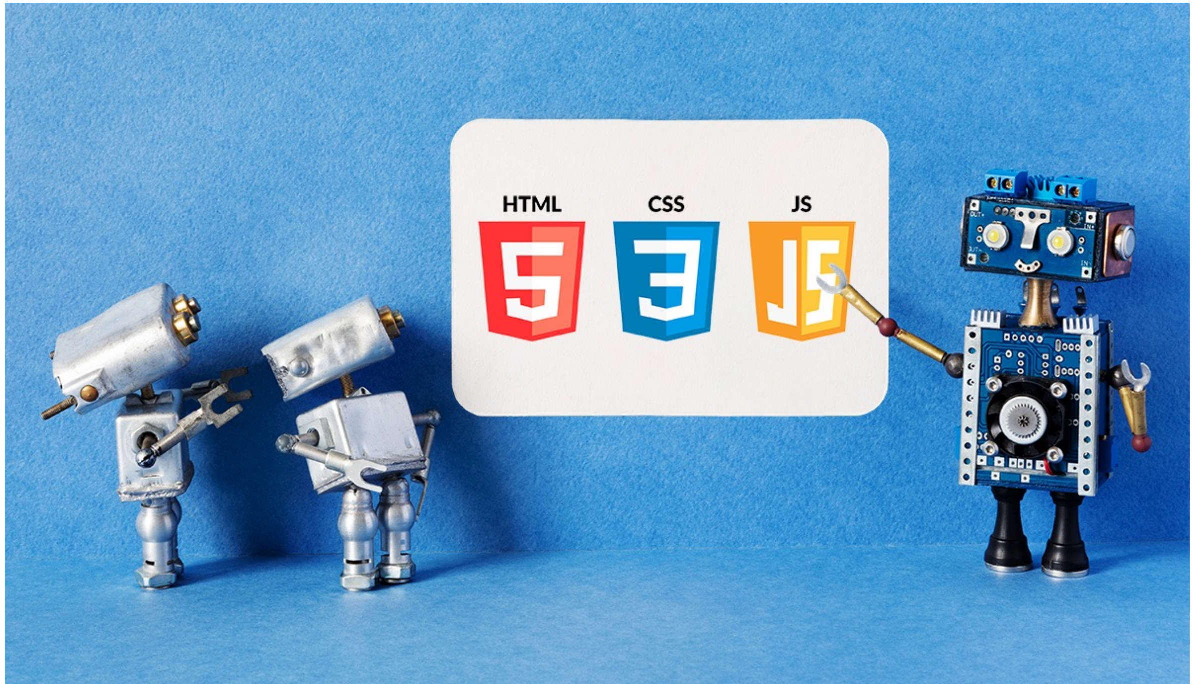
- Each group must select a unique topic
- Topics are selected on a first-come-first-serve basis
  - When your group has selected a topic, notify your instructor

Topic Selection

- Available Topics
  - CSS Custom Properties (CSS Variables)
  - CSS Transitions, Transforms, and Animations
  - CSS Selectors Beyond Class, ID, and Element Selectors
  - 3 - 5 New or Up-and-Coming CSS Technologies
  - SVG
  - Markdown
  - Bootstrap or Similar Framework
  - Web APIs such as Geolocation, Fetch, and Web Storage
  - Static Site Generators
  - Progressive Web Apps
  - Other Topics with Instructor Approval

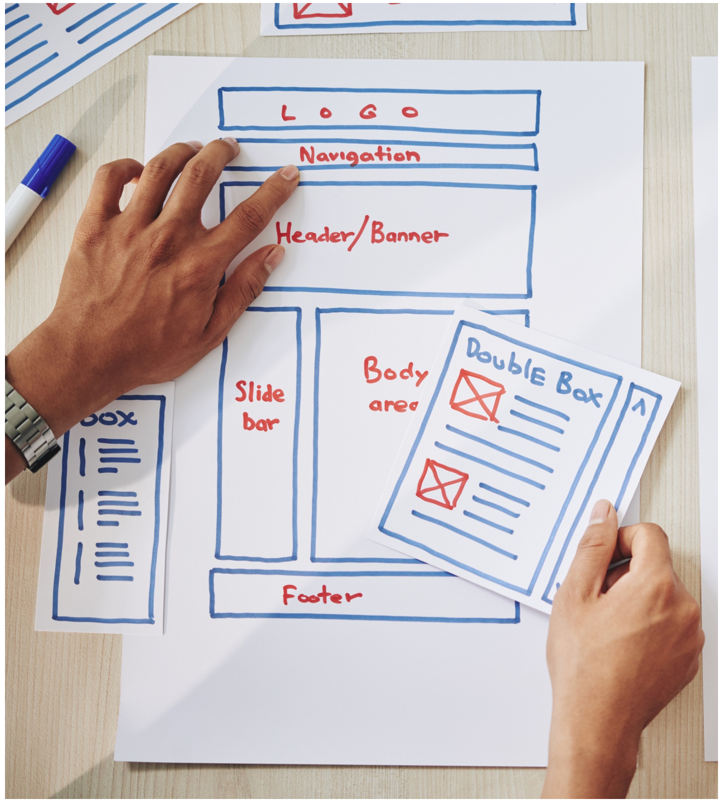
Project 1a - Presentation

- Duration 10-20 minutes
- Content Requirements
  - Technology Overview
    - What is it
    - What does it do
    - Why would you use it
    - Where would you use it
    - Pros and Cons
  - A code demo or technology walk-through



Project 1b - Website

- Technical Requirements
  - One or more page website
  - Hosted and shared using GitHub’s Pages feature
    - See this link for instructions on how to create a GitHub Pages site:
      - https://docs.github.com/en/pages/getting-started-with-github-pages/creating-a-github-pages-site
  - Share the URL to your site with the class in the Slack channel for the Web Scripting 3 course
- Content Requirements
  - Like the presentation, answer the what, why, where, and how of your chosen topic



Project 1a – Presentation Rubric

- Excellent level for the evaluation criteria
  - **Topic Understanding**: Demonstrate a comprehensive understanding of your selected web technology and articulate its use cases and applications well.
  - **Code Demo and Walk-Through**: Provide an exceptionally clear and insightful code demonstration that not only works but significantly contributes to the understanding of the technology.
  - **Presentation Skills**: Deliver a presentation that is exceptionally clear, engaging, and well-organized, capturing and maintaining the audience's attention throughout.
  - **Group Collaboration**: Exhibit clear evidence of equitable and effective collaboration among all group members, where everyone has contributed significantly to the project.
  - **Relevance and Originality**: Go beyond the basic requirements by adding unique perspectives or elements that are not only relevant but offer original insights into the topic.
  - **Time Management**: Adhere perfectly to the 10–20-minute guideline, utilizing the time effectively to cover all important points without rushing or stretching the content.

Project 1b – Website Rubric

- Excellent level for the evaluation criteria
  - **Content Quality:** Present a comprehensive and insightful overview of the selected technology that fulfills all specified content requirements, leaving no gaps in information.
  - **Design and Usability:** Create an exceptionally user-friendly and visually appealing design that complements and enhances the content, ensuring a seamless user experience.
  - **Code Quality:** Develop clean, efficient code that adheres to industry best practices, demonstrating mastery over the technologies used.
  - **GitHub Pages:** Maintain a well-organized public repository and properly set up the GitHub Pages site. The URL should be shared as instructed, and the repository should facilitate easy navigation.

Project Submission

- Live In-Class Presentation
- One group member should submit a URL to the group’s GitHub Pages URL

Project Penalties

- Presentation component
  - 10% for not being in attendance for other group presentations
- Website component
  - 5% per to a maximum of 50%
  - No projects accepted that are more than 14 days late

Strategies for Effective Group Work

- **Clear Role Assignment**: Define roles and responsibilities for each group member to ensure a balanced workload.
- **Communication Channels**: Establish a regular communication channel such as Slack, Microsoft Teams, or regular meetings to keep everyone in the loop.
- **Timelines and Milestones**: Develop a project timeline with milestones and deadlines to keep the team on track.
- **Conflict Resolution**: Agree on a method for resolving conflicts or making decisions, whether by majority vote or consensus.

Note: The above edited points were originally generated by ChatGPT

Strategies for Effective Group Work

- **Version Control**: Utilize version control tools like Git to manage changes and avoid work duplication.
- **Regular Check-ins**: Schedule meetings to assess progress, discuss challenges, and recalibrate roles or tasks as needed.
- **Peer Reviews**: Incorporate a peer review process to evaluate each other’s contributions and ensure high-quality output.
- **Feedback Loops**: Create mechanisms for internal feedback before the final submission to improve the quality of the work.

Note: The above edited points were originally generated by ChatGPT

FAQs

- **What technology should my presentation be in?**
  - You can use whatever you like…
    - PPT, Google Slides, Canva, or other web-based presentation tools could be used.
    - Create a web page and present that.
    - Create demo pages and show your code…
  - The goal of the presentation is to communicate what, why, where, and how of the technology to the class.
- **Can we show a pre-existing YouTube video not made by our group on the screen and call it a day?**
  - No, however, if the video you are showing is less than 30% of the total duration of your presentation, then that is fine.
  - You are allowed to include links to videos or embed videos made by others on your GitHub Pages website.

FAQs

- **Can we use Bootstrap for our GitHub pages site?**
  - Yes
- **Can we use other web technologies like Sass and jQuery?**
  - Yes, if it is compatible with GitHub Pages
- **Can we upload our presentation to our GitHub pages site?**
  - Yes, if you think it adds to your webpage.
    - Just make sure to link to it in your document. Don’t just upload it to your GitHub repo.
- **Do we have to attend the other group presentations?**
  - Yes, if you do not attend the other groups’ presentations without a valid reason, you will receive a 10% penalty.