Form Validation with CSS

Validation

- Validating input is essential for any computer program.
- With web forms, input validation comes in two forms:
  - Server side
  - Client side
- Validate things like whether a field has been filled in, whether the field contains the right input, or whatever you want to validate.

Client Side Validation

- Client side validation occurs on the user's device, while the input is still under their control.
- Client side validation occurs through two primary means:
  - HTML
  - JavaScript
- The HTML validation is relatively new; Not all browsers support it yet, so JavaScript is needed.
- This lecture examines the HTML-based validation; JavaScript will be covered later in the course.

Server Side Validation

- Server side validation takes the input from the user after the user has relinquished control.
- This point of control is important.
- Because the input is no longer under user control, the validation can proceed with the assumption that it will not change.
- **Never assume that client side validation has worked.**

Never Assume

- Client side validation can and will break.
  - Sometimes JavaScript has been disabled.
  - Sometimes there are malicious users.
- The server side program should always assume that the data has been tampered with and is invalid.
- This assumption should extend to all input types, including hidden fields, radio buttons, selects, and checkboxes.
- **Never assume input data is valid; Always assume input data is bad.**

Required and Optional

- HTML validation occurs through several means.
- Requiring a field has been filled in with {something} is as simple as using the 'required' keyword:
- <input type="text" name="email" required>
- You can add 'required' to a single radio button or checkbox input and it will require one of that group to be selected.

Reacting with CSS

- If a user fails to fill in a required field, the browser will put a message in.
- You should also add some styling. There are pseudo-classes for that
  - valid
  - invalid
  - in-range
  - out-of-range
- There are also pseudo classes that are used when the "required" attribute is present or missing:
- :required
- :optional

Practice: Required Field

- Take the form that you've built (in practice or assignment)
- Add a * to the HTML to indicate that one or more fields are required.
- Add the 'required' attribute to those inputs.
- Add a pseudoclass for the field when invalid:

```
input:required:invalid { background-color: lightyellow; }
```

But Required is Just Required

- You may really want to validate what they're filling in.
- For that, the pattern attribute is used.
- Pattern attributes bring us into the wonderful world of Regular Expressions.

Regular Expressions

- Regular expressions are patterns that are used to match characters within text.
- Regular expressions can be used to indicate that a given bit of text, say a form field, should contain only digits or should fit a certain pattern, like a phone number.
- The use of regular expressions is prevalent in web and other programming but understanding them is typically a lifelong endeavor (or seems that way).

Simple Sample Regexes

- Here's a regex pattern for just digits (more than one):
  - [\d+]
- A pattern for letters:
  - [A-Za-z]
- A pattern for words:
  - [A-Za-z0-9]
    - (Note the blank "space" within that character class)
    - Note that regular expressions are case sensitive.

But It's Simpler

- The previous slide showed the use of character classes and metacharacters.
- Character classes appear within brackets and indicate a level of abstraction.
- Character class shortcuts such as \d match special types (like digits for \d) – Also \w for words (and others)
- You don't need character classes, just make a pattern and match it:
  - pattern="me@example.com" will only allow me@example.com within the field.

Determine What's "True"

- The first step in developing a regular expression is determining what's true.
- Will everyone always have "me@example.com" as their e-mail address?
- Is a phone number always NNN-NNN-NNNN ?
- When you determine truth, you can write the regex.

Metacharacters

- Metacharacters have special meaning.
- The . is a metacharacter in regex; It matches any single character except a newline.
- That . in the e-mail example actually matches any character, not just a literal .
- If you need to include a dot or other special character you need to escape it with a backslash: \.
- There are numerous metacharacters or special characters, including some covered on the forthcoming slides.

Quantifiers

- Quantifiers specify a quantity of the preceding character.
- Four primary quantifiers:
  - * - Match any character; zero or more of them.
  - + - Match the previous character one or more times.
  - ? – Match once or none, but not more than one.
  - { } – Repetitive quantifier

Repetitive Quantifiers

- Repetitive quantifiers specify how many of a certain pattern is acceptable.
- Example:
  - pattern="\d{5}"
  - Match any digit exactly 5 times.
- Ranges:
  - {x,y} – Match at least x but not exceeding y
  - {x,} – Match at least x to any quantity

Validation Practice

- Update your form or create a new one.
- Add an text input to capture a phone number.
- Write a validation pattern for it.
- Truth:
  - U.S. phone numbers are NNN-NNN-NNNN
  - The field is required
- Suggestions for how to program for the truth?

E-mail Validation

- E-mail validation is difficult.
- It's much more than just me@example.com
- Newer browsers include support for the email input type:
- <input type="email" ...

More Inputs

- IE lags in support for the email input type; Need v10 or higher.
- Other input types:
  - url
  - search
  - tel
- As IE ~~catches up~~ evolves this support will be better.

Even More Inputs and Even More Attributes

- Sliders, progress bars, date inputs, and more are all coming.
- There are other attributes like tabindex and autofocus which weren't covered here.
- See https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input for a comprehensive examination of inputs.

Validation Summary

- Validation is now possible with HTML5
- Browser support is getting better (mainly as older IEs go away)
- Use the HTML5 validation sparingly for now but the ideas for how to validate are used in JavaScript.
- See http://html5pattern.com for pattern ideas