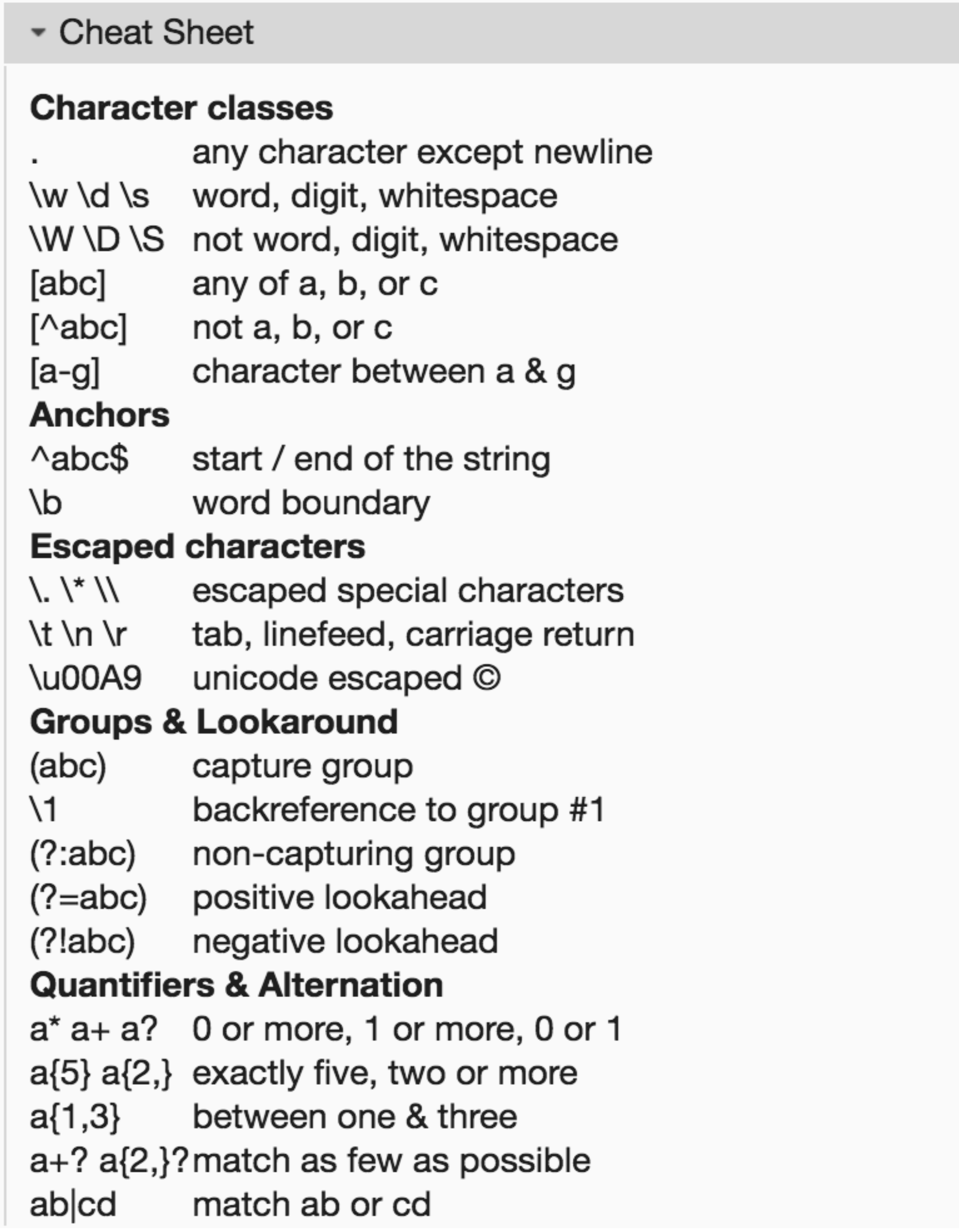
CSC 337

Client Side Validation and REGEX

Marty Stepp
Jessica Miller
Allison Obourne
Rick Mercer



Cheat Sheet

Character classes	
.	any character except newline
\w \d \s	word, digit, whitespace
\W \D \S	not word, digit, whitespace
[abc]	any of a, b, or c
[^abc]	not a, b, or c
[a-g]	character between a & g
Anchors	
^abc$	start / end of the string
\b	word boundary
Escaped characters	
\. * \\	escaped special characters
\t \n \r	tab, linefeed, carriage return
\u00A9	unicode escaped ©
Groups & Lookaround	
(abc)	capture group
\1	backreference to group #1
(?:abc)	non-capturing group
(?=abc)	positive lookahead
(?!abc)	negative lookahead
Quantifiers & Alternation	
a* a+ a?	0 or more, 1 or more, 0 or 1
a{5} a{2,}	exactly five, two or more
a{1,3}	between one & three
a+? a{2,}?	match as few as possible
ab|cd	match ab or cd

Form Validation

- Validation: ensuring <form> input values are correct
- Types of validation
  - Preventing blank values
  - Ensuring the type of values
    - integer, real number, currency, phone number, Social Security number, postal address, email, date, ...
  - Ensuring the format and range of values
    - ZIP codes must be a 5-digit integer
  - Ensuring that values fit together
    - user types email twice, and the two must match

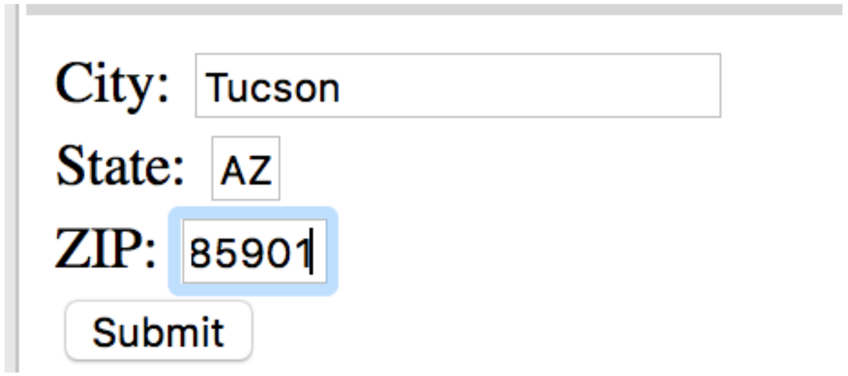
Can Validate on Client. in JS, and Server

- **Client-side** as part of an HTML form, before Submit
  - can lead to a better user experience
- **JS function or Server-Side** after form is submitted
  - needed for truly secure validation
- **Both**
  - best mix of convenience and security
- We only consider client side validation now

Example of client side validation

- Make input field smaller: `size` of `"2"` or `"5"` below
- Ensure the user can’t enter more than 2 letters for state or 5 letters for a `"zip"` code

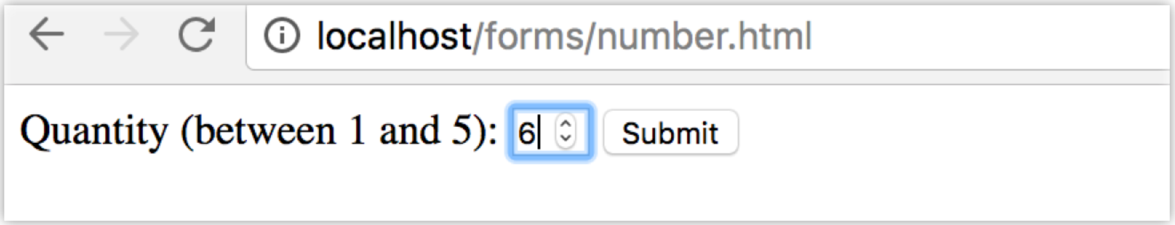
```
<form>
 City:  <input name="city"> <br>
 State: <input name="state" size="2" maxlength="2"> <br>
 ZIP:   <input name="zip" size="5" maxlength="5"> <br>
 <input type="submit">
</form>
```


City: Tucson
State: AZ
ZIP: 85901

One Validation

```
<form>
  Quantity (between 1 and 5):
  <input type="number" name="quantity" min="1" max="5">
  <input type="submit">
</form>
```

- Any input other than 1, 2, 3, 4, or 5 will show an error when submit is clicked
  - The form does not submit

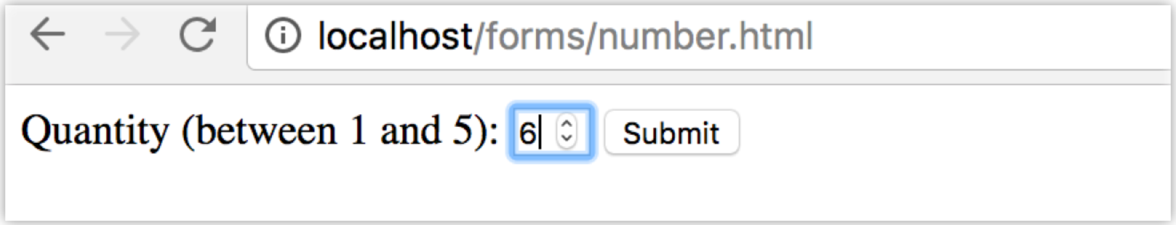


Quantity (between 1 and 5): 6

Better Client Side Validation: ranges

```
<form>
  Quantity (between 1 and 5):
  <input type="number" name="quantity" min="1" max="5">
  <input type="submit">
</form>
```

- Any input other than 1, 2, 3, 4, or 5 will show an error when submit is clicked
  - The form does not submit

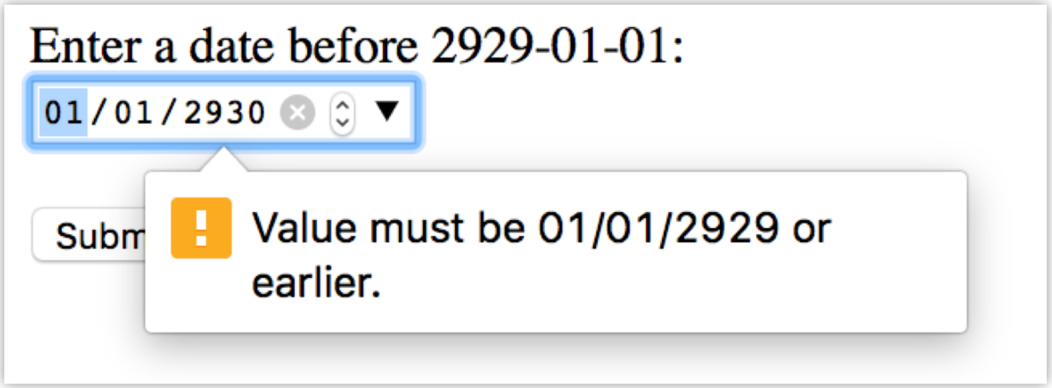

localhost/forms/number.html

Quantity (between 1 and 5): 6

Client Side Validation: ranges

```
<form>
  Enter a date >= 1980-01-01:<br>
  <input type="date" min="1980-01-01"><br><br>
  Enter a date before 2929-01-01:<br>
  <input type="date" max="2929-01-01"><br><br>
  <input type="submit">
</form>
```

- Out of range dates causes an error
  - This form does not submit


Enter a date before 2929-01-01:

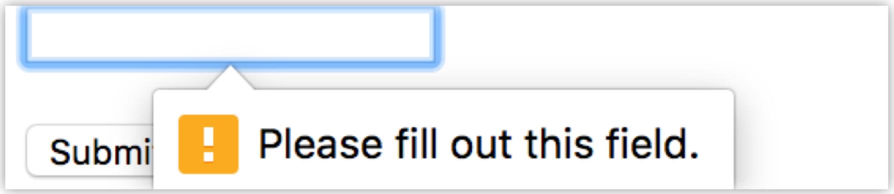
01/01/2930

Value must be 01/01/2929 or earlier.

The required attribute

```
<form>
  <input type="text" name="firstName" required>
  <br> <br>
  <input type="submit">
</form>
```

- If the input part of a form is empty, the browser will write something like this when the submit button is clicked


Please fill out this field.

Regular expressions

- Regular expression (regex)
  - text string for describing a search pattern
  - tests whether a string matches the expression's pattern
  - can use a regex to search/replace characters in a string
- Regular expressions are extremely powerful but difficult to read and write correctly
- This regular expression matches email addresses

```
"^[a-zA-Z_\-]+@(([a-zA-Z_\-])+\.)+[a-zA-Z]{2,4}$"
```

- Regular expressions are wildcards on steroids

  `ls *.txt` show all files of type `txt`

Where are regular expression used?

- Regular expressions occur with
  - Supported by Java, PHP, JavaScript, and other languages
  - Java Scanner, String's split method, Python’s split
  - Many text editors allow regexes in search/replace
  - HTML forms use regex expressions to ensure correct formatting of user input

Quantifiers {min,max}

- `{min,max}` means between min and max occurrences (inclusive)
    - `a(bc){2,4}` matches "abcbc" or "abcbcbcbc"
- min or max may be omitted to specify any number
    - `{2,}` means 2 or more
    - `{,6}` means up to 6
    - `{3}` means exactly 3

Anchors

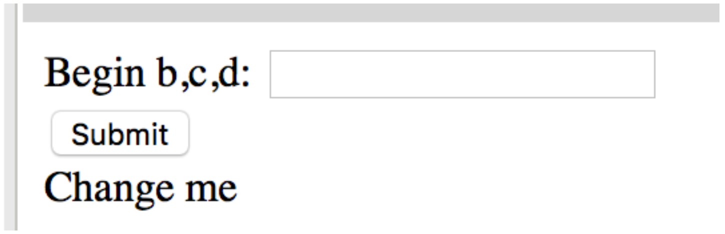
- `^` represents the beginning of the string or line;
- `$` represents the end
  - `Jess` matches all strings that contain Jess;
  - `^Jess` matches all strings that start with Jess;
  - `Jess$` matches all strings that end with Jess;
  - `^Jess$` matches the exact string "Jess" only

Character sets [ ]

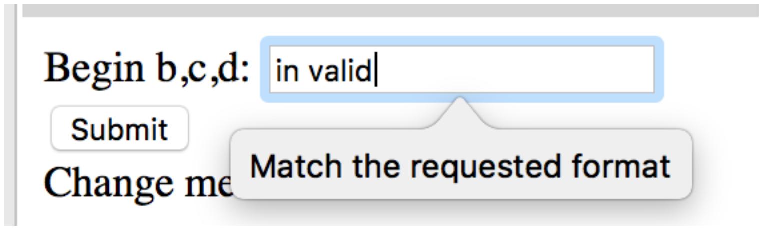
- `[]` group characters into a character set; will match any single character from the set
  - `[bcd]art` matches strings containing "bart", "cart", "dart"
  - equivalent to `(b|c|d)art` but shorter

Full Example

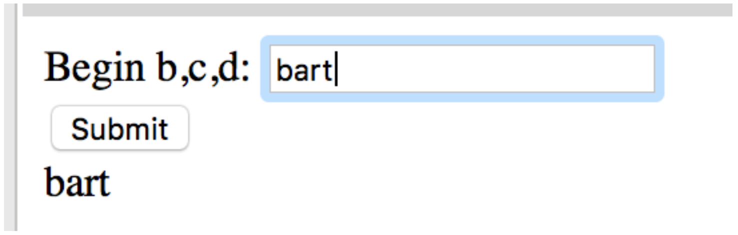
```
<form onsubmit="f(event); return false;">
  Begin b,c,d:
  <input id="in"
         type="text"
         pattern="[bcd]art"
         required> <br>
<input type="submit">

</form>
<div id="here">Change me</div>
<script>
function f(event) {
  event.preventDefault();
  document.getElementById("here").innerHTML =
       document.getElementById("in").value;

}
</script>
```


Begin b,c,d:

Change me


Begin b,c,d: in valid

Change me
Match the requested format


Begin b,c,d: bart

bart

Character ranges [start-end]

- Specify a range of characters with -
  - `[a-z]` matches any lowercase letter
  - `[a-zA-Z0-9]` matches any letter or digit
- An initial ^ inside a character set negates it
  - `[^abcd]` matches any character other than a, b, c, or d
- Inside a character set, - must be escaped to match
  `[+\-]?[0-9]+` matches an optional `+` or `–`, followed by at least one digit
- What regular expression matches letter grades A, B+, or D- ?

Regex Required on Project

Regex Expression	Description
`[A-Z a-z]*`	Any number of spaces, upper case and lower case letters
`^(\+0?1\s)?\(?\d{3}\)?[\s.-]\d{3}[\s.-]\d{4}$`	Many phone number formats (520) 123-4567
`[A-Z a-z]*`	Any number of spaces, upper case and lower case letters
`[0-9]{5}`	Exactly 5 digits
`1-5`	Digits 1, 2, 3, 4,or 5 only

In class Activity a.k.a Learning

- Build a form that has two input fields of type password that accepts only upper- and lower-case letters and digit 0..9
- When the submit button is clicked
  - write "Match" below the 2nd password field if the same exact password were in the password fields
  - If the don't match write "Passwords do not match"