Functions and The DOM

Putting the "fun" in functions

Functions

Functions allow us to reuse code

- Declaring it

```
function func(p1, p2) {
  /* Do things with p1 and p2 here. */
  return anythingYouWant;
}
```

- Calling it

```
x = func(5, "hello");
```

There are four ways to declare functions

1. Instantiate a Function
2. Function statement
3. Function expression
4. Arrow function

```
f = new Function('arg1, arg2','body here');
```

1. Instantiate a function

2. Function statement

```
function func(p1, p2) {
  /* Do things with p1 and p2 here. */
  return anythingYouWant;
}
```

3. Function expression

```
func = function (p1, p2) {
 /* Do things with p1 and p2 here. */
  return anythingYouWant;
}
```

4. Arrow operator

```
func = (p1, p2) => {
  /* Do things with p1 and p2 here. */
  return anythingYouWant;
}
```

- Parentheses can be omitted if # of parameters is one
- Curly braces can be omitted if # of lines is one
  - If you do, the function implicitly returns the value of your one line

Built-in Functions

There are many built-in functions

- String functions
- Number functions
- Window functions
- Date functions
- Math functions
- Many, many more

Here's an example built-in function

```
let theThing = elementFromPoint(x, y);
```

- Will return an object ... the thing visible at that location.

... or ...

```
let theThing = document.elementFromPoint(x, y);
```

- Same exact thing!

But wait, what is this 'document' thing?

It is an object which represents the entire document loaded.

It has an property called "children"

```
let kids = document.children;
```

- 'kids' is an object of type *HTMLCollection*

```
let greatGreatGrandChild =
  document.children[0].children[4].children[8].children[1];
```

- You can use this to get any element on the page
- It's a **model** of all of the **objects** in the **document**! It is the ...

MOD DOM!!


IMDb
Find Movies, TV shows, Celebrities and more...
All
Movies, TV & Showtimes
Celebs, Events & Photos
News & Community
Watchlist
Enjoy unlimited streaming on Prime Video
Includes thousands of titles. Monthly plans now available.
Start your 30-day free trial
FULL CAST AND CREW | TRIVIA | USER REVIEWS | IMDbPro | MORE | SHARE
Spider-Man (2002)
PG-13 | 2h 1min | Action, Adventure, Fantasy | 3 May 2002 (USA)
7.3/10
540,822
Rate This
1:06 | Trailer
5 VIDEOS | 114 IMAGES
For example ... Menus

To alter content dynamically, we will ...

1. Get existing elements
2. Change those elements
   - Remove them
   - Change them
   - Append content to them

Getting elements

```
let aDiv = document.getElementById("linksDiv");
let allDivs = document.getElementsByTagName("div");
let allAlerts = document.getElementsByClassName('alert');
```

- And the all-new, all-improved ...

```
let firstradio = document.querySelector('[type=radio]');
let allTxtBox = document.querySelectorAll('[type=text]');
```

Removing elements

```
let d = document.getElementById("theDiv");
d.remove();
```

To add elements, use createElement()

```
let p = document.createElement("p");
p.textContent = "Excelsior!"

aParent.insertBefore(p, null);

let i = document.createElement("img");
i.src = "/imgs/stanLee.jpg";

aParent.insertBefore(i, p);
```

Create a <p> in memory...

... and put it on the page

Create an <img>

Put it before the paragraph

Changing elements

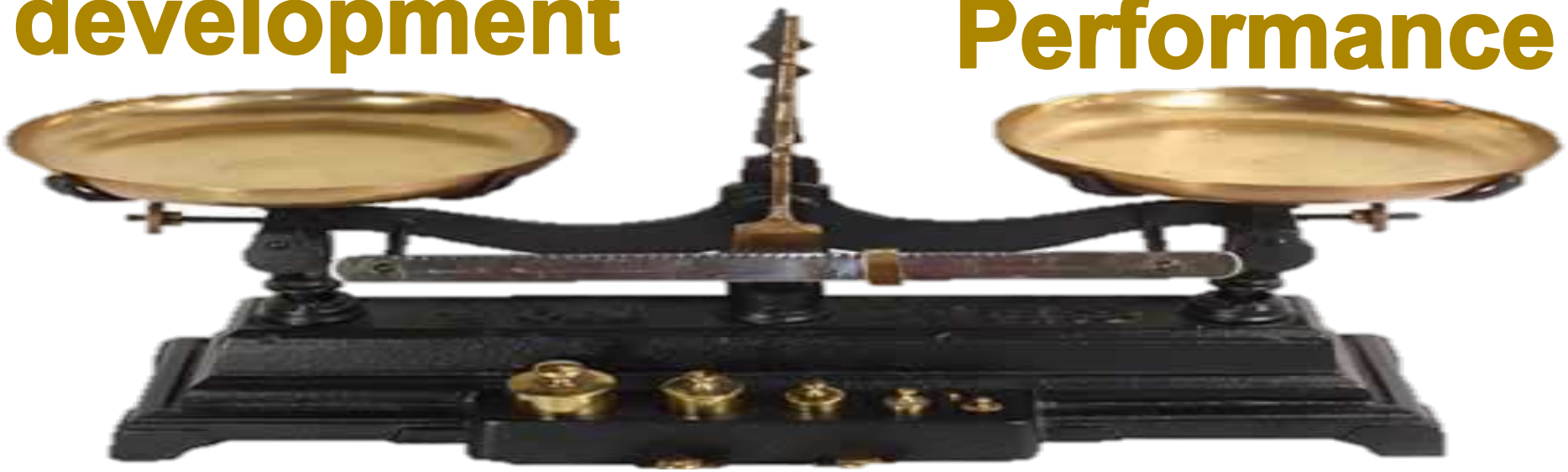
```
p.style.background = "yellow";
p.style.height = "50px";
p.setAttribute("title", "Stan says this");
```

Adding with innerHTML

When you set innerHTML, the browser ...

1. forces a parsing of a sub-DOM
2. which is then injected into the page DOM
3. which is then re-parsed and
4. which is then re-rendered

Ease of development



Performance

Forms and the DOM

Use *value* to get to form fields

- Like <input type="whatever" />
- To read:

```
let foo = field.value;
```

- To write:

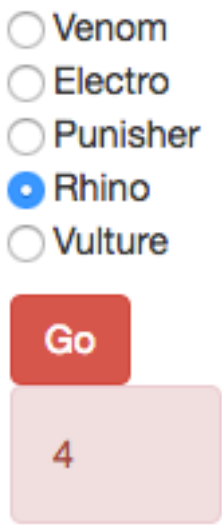
```
field.value = "foo";
```

Note: It is not innerHTML nor innerText.

Radio buttons

- Most say you have to loop through radio buttons manually.
- But you don't ...

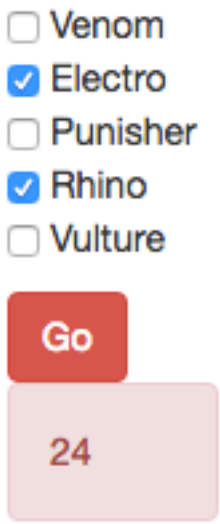
```
<input type='radio' value="1" name="foe" />Venom
<input type='radio' value="2" name="foe" />Electro
<input type='radio' value="3" name="foe" />Punisher
<input type='radio' value="4" name="foe" />Rhino
<input type='radio' value="5" name="foe" />Vulture
<button>Go</button>
<div id="output"></div>
<script>
 o = document.getElementById("output");
 b = document.querySelector("button");
 b.addEventListener("click",() => {
  foe = document.querySelector(":checked");
  o.innerText = foe.value;
 });
</script>
```



Checkboxes are largely the same

- Since the user can choose >1 thing ...

```
<input type='checkbox' value="1" name="foe" />Venom
<input type='checkbox' value="2" name="foe" />Electro
<input type='checkbox' value="3" name="foe" />Punisher
<input type='checkbox' value="4" name="foe" />Rhino
<input type='checkbox' value="5" name="foe" />Vulture
<button>Go</button>
<div id="output"></div>
<script>
 o = document.getElementById("output");
 b = document.querySelector("button");
 b.addEventListener("click",() => {
  foes = document.querySelectorAll(":checked");
  for (var i=0;i<foes.length;i++)
    o.innerText += foes[i].value;
 });
</script>
```



Select Lists

```
<select multiple>
 <option value="1">Venom</option>
 <option value="2">Electro</option>
 <option value="3">Punisher</option>
 <option value="4">Rhino</option>
 <option value="5">Vulture</option>
</select>
<button>Go</button>
<div id="output"></div>
<script>
 o = document.getElementById("output");
 b = document.querySelector("button");
 b.addEventListener("click",() => {
  foes = document.querySelectorAll(":checked");
  console.log(foes);
  for (var i=0 ; i<foes.length; i++) {
    o.innerText += foes[i].value;
  }
```

React

DOM manipulation is so much easier with certain JavaScript libraries

tl;dr

- Functions create reusable code
- There are built-in functions in addition to the ones you create
- Some of those are on the document object
- Speaking of which ...
- The document object give you JavaScript access to anything on the page
- You can add elements, remove elements, or change elements