CSE 154

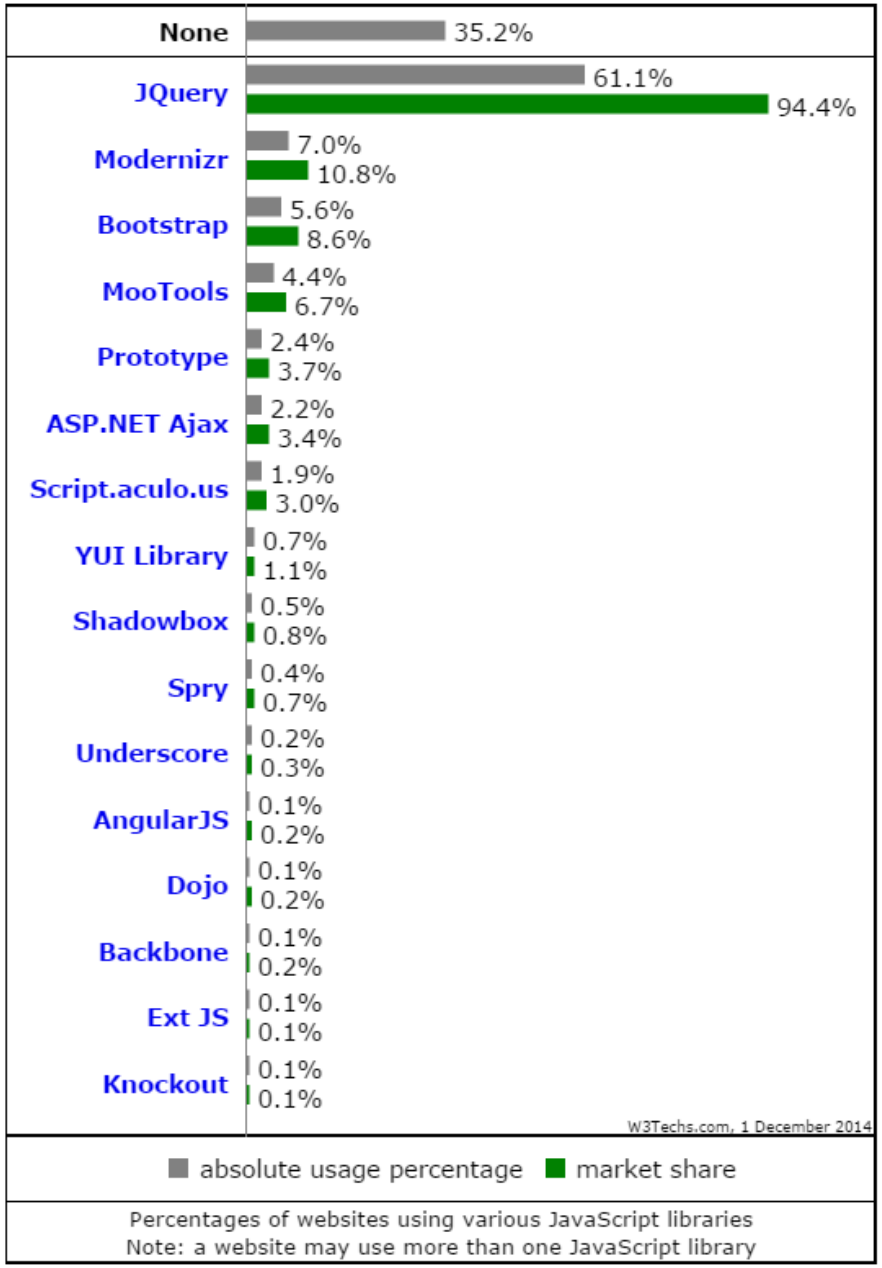
LECTURE 26: JAVASCRIPT FRAMEWORKS

Why Frameworks?

JavaScript is a powerful language, but it has many flaws:

- the DOM can be clunky to use
- the same code doesn't always work the same way in every browser
  - code that works great in Chrome, Firefox, Safari, ... will fail in IE and vice versa
  - many developers work around these problems with hacks (checking if browser is IE, etc.)

Which framework should you learn?



Source:
http://w3techs.com/technologies/overview/javascript_library/all

Prototype

```
<script src="https://ajax.googleapis.com/ajax/libs/prototype/1.7.1.0/prototype.js"
        type="text/javascript"></script>
```

- Started in 2005
- Original goal: make Ajax better and simplify things
- Not intrusive
  - Style similar to plain Javascript

JQuery

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"
        type="text/javascript"></script>
```

- Started in 2006
- Currently the most popular
- Original goal: enhance and normalize browsers

Selecting Elements

JQuery

```
$ == document.querySelectorAll()
```

Prototype

```
$ == document.getElementById()
$$ == document.querySelectorAll()
```

Avoid looping!

No Framework

```
var elems = document.querySelectorAll("li");
for (var i = 0; i < elems.length; i++) {
  var e = elems[i];
  // do stuff with e
}
```

Prototype

```
$$(“li").each(function(e) {
    // do stuff with e
});
```

JQuery

```
$("li").each(function(idx, e) {
  // do stuff with this
});

$('.findMe').css({ "color" : "red" });
```

Accessing Styles

Prototype

```
function biggerFont() {
  // turn text yellow and make it bigger
  var size = $("clickme").getStyle("font-size");   // "22px"
  $("clickme").style.fontSize = parseInt(size) + 4 + "pt";
}
```

JQuery

```
function biggerFont() {
  // turn text yellow and make it bigger
  var size = parseInt($("#clickme").css("font-size"));
  $("#clickme").css("font-size", size + 4 + "pt");
}
```

Ajax

Prototype

```
new Ajax.Request("foo/bar/mydata.txt", {
    method: "get",
    onSuccess: myAjaxSuccessFunction
});
...

function myAjaxSuccessFunction(ajax) {
  do something with ajax.responseText;
}
```

JQuery

```
$.get({
    url : "service.php",
    data : {
        param1 : data1,
        param2 : data2
    }
    ...
})
```

JQuery: The jQuery object

- The $ function always (even for ID selectors) returns an array-like object called a jQuery object.
- The jQuery object wraps the originally selected DOM objects.
- You can access the actual DOM object by accessing the elements of the jQuery object.

```
// false
document.getElementById("id") == $("#myid");
document.querySelectorAll("p") == $("p");

// true
document.getElementById("id") == $("#myid")[0];
document.getElementById("id") == $("#myid").get(0);

document.querySelectorAll("p")[0] == $("p")[0];
```

JQuery: using $ as a wrapper

- $ adds extra functionality to DOM elements
- passing an existing DOM object to $ will give it the jQuery upgrade

```
// convert regular DOM objects to a jQuery object
var elem = document.getElementById("myelem");
elem = $(elem);

var elems = document.querySelectorAll(".special");
elems = $(elems);
```

jQuery: method returns

When there is no other return to make, jQuery methods return the same jQuery object back to you

method	**return type**
$("#myid");	jQuery object
$("#myid").children();	jQuery object
$("#myid").css("margin-left");	String
$("#myid").css("margin-left", "10px");	**jQuery object**
$("#myid").addClass("special");	**jQuery object**

jQuery: chaining

```
$("#main").css("color", "red");
$("#main").attr("id", "themainarea");
$("#main").addClass("special");
```

The implicitly returned jQuery object allows for chaining of method calls.

```
$("img")
  .css("color", "red")
  .addClass("special")
  .src = "foo.png";
```

Expression return value at each line:

```
// [<img />, ...]
// [<img style="color: red" />, ...]
// [<img class="special" style="color: red" />, ...]
// cannot chain further because this is an assignment :(
```

jQuery: attr() function

- jQuery has a wrapper function for getting/setting various attributes of selected elements.
- Allows us to chain our method calls.

```
$("img")                    // poor jQuery style
  .css("color", "red")
  .addClass("special")
  .src = "foo.png";
```

```
$("img")                   // good jQuery style
  .css("color", "red")
  .addClass("special")
  .attr("src", "foo.png");
  // we could chain further right here
```

JQuery: creating nodes

```
$("#ex3 span.special").prepend($("<img>", {
  "src": "images/laughing_man.jpg",
  "alt": "laughing man",
  "css": {
    "vertical-align": "middle",
    "border": "2px solid black"
  },
  "click": function() {
    alert("clicked");
  }
}));
```

Scriptaculous overview

- **Scriptaculous** : a JavaScript library, built on top of Prototype, that adds:
- visual effects (animation, fade in/out, highlighting)
- drag and drop
- Ajax features:
  - Auto-completing text fields (drop-down list of matching choices)
  - In-place editors (clickable text that you can edit and send to server)
- some DOM enhancements
- other stuff (unit testing, etc.)