Learning JavaScript (in CS312)

JavaScript is an object-oriented, prototype-based, dynamic, “brackets” language

- A pragmatic language that “evolved” (instead of being “designed”)
- Gotchas abound
- Recent versions (ES6) have smoothed some rough edges (e.g. introduced ”classes”)

The tools (and the notes) will teach us the gotchas, our goal in-class is the main ideas

Declaring variables

- ~~no declaration~~
  - implicitly create a new global variable
- ~~`var`~~
  - create new variable with function (or global) scope
  - variables are *hoisted* to the top of their context
- `let`
  - create new variable with block-level scope
- `const`
  - create a new constant with block-level scope

Higher-order functions

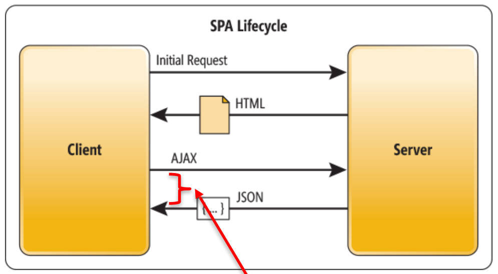
```
const m = [4,6,2,7];
for (let i=0; i<m.length; i++) {
  console.log(m[i]);
}
```

```
m.forEach(function(i) {
  console.log(i);
});

// or…

m.forEach((i) => {
  console.log(i)
});
```

Abstract over “actions” not just values by passing functions as arguments

Common operations of this kind are `map`, `filter`, `reduce` and `sort`

Callbacks



What is happening during this time?

Wasson, Microsoft

Making callbacks work in JS

Functions as 1st class objects

```
const wrapValue = (n) => { // function(n) {
  const local = n;
  return () => local; // function () { return local; }
}

const wrap1 = wrapValue(1); // () => 1
const wrap2 = wrapValue(2); // () => 2
console.log(wrap1()); // What will print here?
console.log(wrap2()); // What will print here?
```

Function “closes” over `local`

What does the following code print?

```
let current = Date.now(); // Time in ms since epoch

// setTimeout(callback, delay[,param1[,param2…]]) delay in ms
setTimeout(() => {
  console.log("Time elapsed (ms): " + (Date.now() - current))
}, 100);

console.log("First?")
```

A	B	C

`First?`	`First?`
	`Time elapsed (ms): 100`
		`Time elapsed (ms): 100`
		`First?`