JavaScript Versioning

JavaScript Versioning

- Behind JavaScript, is EMCAScript
    - The standard is called ECMA-262
- Up to now, JavaScript was always based on a number
    - Now it is a year-based schema

Dealing with new Versions

Working with future JS, today

- Ignore it
- Polyfills and Shims
- Transpilation

Variables in JavaScript

Variables in JavaScript

What's wrong with `var`?

- Function Scoping vs. Block Scoping
- No **Temporal Dead Zone**

`let`

- Block Scoped
- Can be re-assigned

```
let something = true;
```

`const`

- Block Scoped
- Can't be re-assigned
    - It has an **immutable binding**

```
const favNumber = 42;
```

Functions in JavaScript

Enter: Arrow Functions

- More concise
- The option of implicit returns
- They are always anonymous though

Arrow Functions

```javascript
const sayHi = () => {
  console.log("Hello");
};

sayHi();
```

Arrow Functions

```
const add = (x, y) => {
  return x + y;
};

add(4, 5);
```

Arrow Functions

```javascript
const add = (x, y) => x + y;

add(4, 5);
```

Default Function Arguments

```
function sayHello(name = "World") {
  return "Hello " + name;
}
```