CS602
Server-Side Web Development

Suresh Kalathur, PhD

kalathur@bu.edu

Overview

- *Topics*
  - *http://kalathur.com/courses/?course_id=cs602_18_summer*
- Weekly Discussions (10%)
- Weekly Assignments (30%)
- Closed Book/Closed Notes Final Exam (30%)
  - Essay type Code Snippet Questions
- Term Project (30%)

Project

- Concentration on Server-side Functionality
- Node.js option or PHP option

Server-Side Options

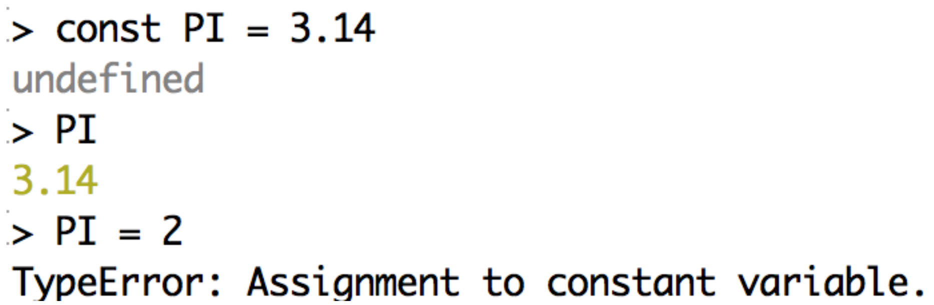
- Java
- .NET
- Ruby/Rails
- Python
- PHP
- Node.js

Advanced JavaScript Topics

- Objects
  - Defining properties
- Creating Objects
  - Factory pattern, Constructor pattern, Prototype pattern, Hybrid pattern
- Inheritance
  - Prototype chaining

ES6 (ECMAScript 6)

- Constants
  - Read-only reference to a value
  - Value itself may be mutable
  - Variable identifier cannot be reassigned



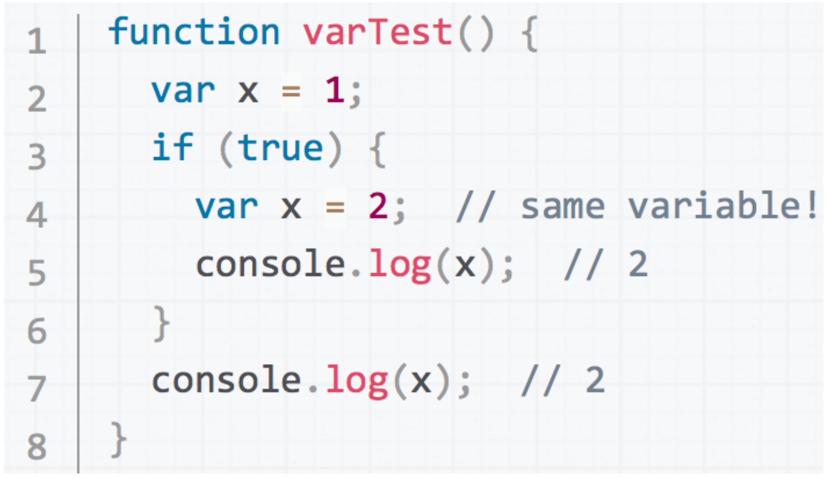
```
> const PI = 3.14
undefined
> PI
3.14
> PI = 2
TypeError: Assignment to constant variable.
```

ES6 (ECMAScript 6)

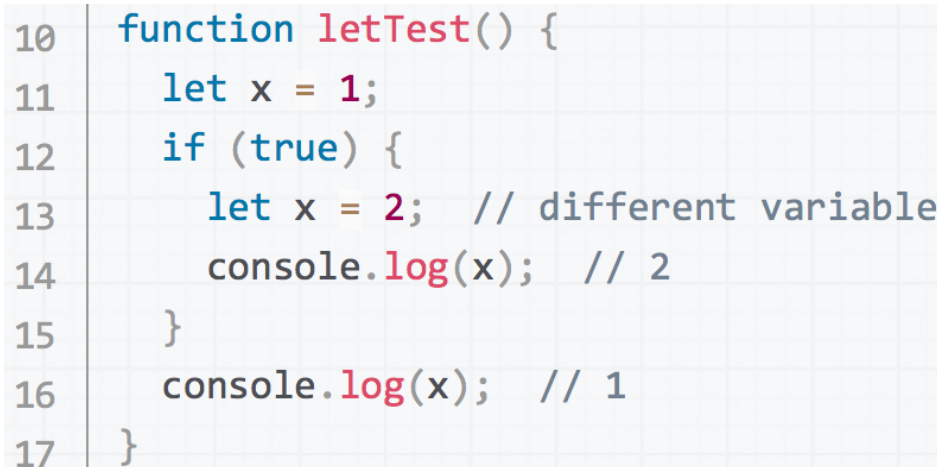
- Variables – **var** versus **let**
- **let**
  - Allows block scope variables
- **var**
  - No concept of a block scope
  - Defines global variables
  - Local to an entire function, if used in a function

ES6 (ECMAScript 6)

- Variables – **var** versus **let**



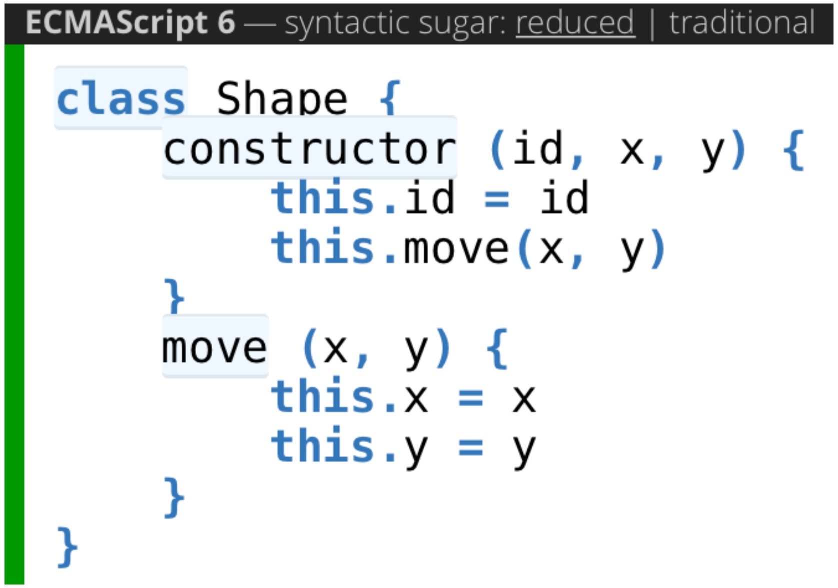
```
function varTest() {
  var x = 1;
  if (true) {
    var x = 2;  // same variable!
    console.log(x);  // 2
  }
  console.log(x);  // 2
}
```



```
function letTest() {
  let x = 1;
  if (true) {
    let x = 2;  // different variable
    console.log(x);  // 2
  }
  console.log(x);  // 1
}
```

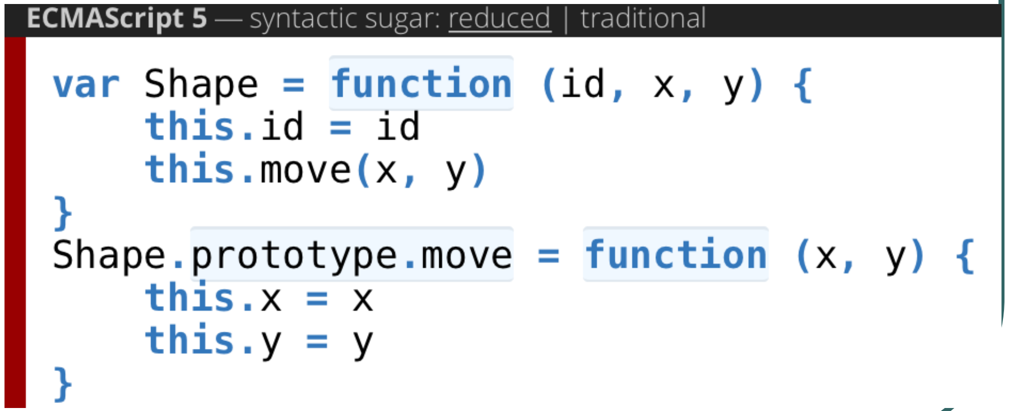
ES6 (ECMAScript 6)

- Class Inheritance   http://es6-features.org/



ECMAScript 6 — syntactic sugar: reduced | traditional

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
}
```



ECMAScript 5 — syntactic sugar: reduced | traditional

```
var Shape = function (id, x, y) {
    this.id = id
    this.move(x, y)
}
Shape.prototype.move = function (x, y) {
    this.x = x
    this.y = y
}
```

ES6 (ECMAScript 6)

- Functions => arrow notation

```
(param1, param2, …, paramN) => { statements }
(param1, param2, …, paramN) => expression
         // equivalent to:  => { return expression; }

// Parentheses are optional when there's only one parameter:
(singleParam) => { statements }
singleParam => { statements }

// A function with no parameters requires parentheses:
() => { statements }
```

ES6 (ECMAScript 6)

```javascript
var data = [10,20,30];

var m1 = data.map(function (value) {
            return 2 * value;
        })

console.log("m1:", m1);

var m2 = data.map(value => 2 * value);

console.log("m2:", m2);
```

```
m1: [ 20, 40, 60 ]
m2: [ 20, 40, 60 ]
```

ES6 (ECMAScript 6)

```
var total1 = data.reduce(function (a, b) {
  return a + b;
}, 0);

console.log("total1:", total1);


var total2 = data.reduce((a, b) => a + b, 0);

console.log("total2:", total2);
```

```
total1: 60
total2: 60
```

Node.js

- A JavaScript runtime
  - Built on Chrome’s V8 JavaScript engine
- Event-driven
  - Event emitters
  - Event listeners
- Non-blocking I/O model
- Lightweight and efficient
- https://nodejs.org

Node.js Modules

- Core of Node.js
- Each JavaScript file exports a module
- Applications import the required modules
- Export
  - Objects
  - Object factory

...Modules

- Core modules
  - path, fs, os, util (covered in lecture)
  - Other modules as we go along
- NPM (Node Package Manager, https://www.npmjs.com)
  - Manages module dependencies
  - Installs modules from Node repository
  - Examples
    - underscore (http://underscorejs.org)
    - Colors (https://www.npmjs.com/package/colors)

Node.js – Events

- Node.js core
  - Event driven architecture
  - Emitters – emit named events
  - Listeners – act on these events
- Emitters
  - Instances of **EventEmitter** class
  - **on()** method – to register listeners
  - **emit()** method – to trigger the event
    - Listeners invoked synchronously, by default

Node.js - Streams

- For working with streaming data
  - Readable
  - Writable
  - Duplex
  - Transform

HW1

- Review HW1
  - Modules
  - Events

…Module2

- Web Applications
  - **net** module – clients and servers
  - **http** module – HTTP protocol clients and servers
  - Express Framework