# JavaScript Array, Loop, etc.

**Harry J. Wang, Ph.D.**

University of Delaware

# References

- Part of the slides are reviews for this course: https://www.codecademy.com/learn/introduction-to-javascript
- The old version of this course is logically better: https://www.codecademy.com/courses/learn-javascript/

# Link External JS Files

- To run the same JavaScript on several web pages, an external JavaScript file should be created with a .js extension and linked in the HTML file.
- The external script file cannot contain the <script> tag.

```
<script src="my-scripts.js"></script>
```

[See Additional Array Methods]

# Arrays

- Arrays store **ordered list** of data
- JavaScript is zero-indexed.

```
let bucket_list = ['Build a house', 'Learn to paint', 'buy a motocycle'];
console.log(bucket_list);
console.log(bucket_list[2]);
```

- length property can be used to find how many items are stored inside of an array.
- .push()/.pop() allows us to add/remove items to **the end** of an array

```
console.log(bucket_list.length);  // 3
bucket_list[2] = "Visit Japan";

bucket_list.push('Climb a mountain');
console.log(bucket_list.length);  // 4

bucket_list.pop();
console.log(bucket_list.length);  // 3
console.log(bucket_list.indexOf("Learn to paint"));  // 1
```

# For Loop (number of loops known)

- Simple for loop

```
let bucket_list = ['Build a house', 'Learn to paint', 'buy a motocycle'];
console.log("My bucket list has the following items:");
for (let i = 0; i < bucket_list.length; i++) {
  console.log( (i + 1) + ": " +bucket_list[i]);
}
```

- Nested loop

```
let imdb_movie_list = [
  "The Shawshank Redemption",
  "The Godfather",
  "The Godfather: Part II",
  "The Dark Knight",
  "12 Angry Men"
];

let ranker_movie_list = [
  "The Godfather",
  "The Shawshank Redemption",
  "Pulp Fiction",
  "Star Wars",
  "Forrest Gump",
  "The Dark Knight"
];
```

```
console.log("matched movies:");
for (let i = 0; i < imdb_movie_list.length; i++) {
  for (let j=0; j < ranker_movie_list.length; j++) {
    if (imdb_movie_list[i] == ranker_movie_list[j]) {
      console.log(imdb_movie_list[i]);
    }
  }
}
```

```
matched movies:
The Shawshank Redemption
The Godfather
The Dark Knight
```

# While Loop (number of loops unknown)

```
while (condition) {
  // code block that loops until
condition is false
}
```

```
let balance = 100;
let years = 0;
while (balance < 1000000) {
  balance += balance * 0.05;
  years += 1;
}
console.log("You will be a millinaire in " + years + "!");
```

```
You will be a millinaire in 189 years !
```

# Iterators

- Iterators are array methods that loop over an array and select elements that meet certain criteria.
- .forEach() loop over a array without changing its elements.
- Try to use arrow function

```
// iterator function
imdb_movie_list.forEach(function(movie){
  console.log(" - " + movie);
});

// iterator arrow function
imdb_movie_list.forEach(movie => console.log(" - " + movie));
```

# .map() and .filter()

- .map() returns a **new array** with elements that have been modified by the code in its block
- .filter() returns a **new array** with certain elements from the original array that evaluate to truth based on conditions written in the block of the method.

```
// map
let my_movie_list = imdb_movie_list.map(movie => "The Godfather");
console.log(my_movie_list);

// return movie with name starts with "The"
// .slice(a, b): a begin, b up to not including
let the_movie_list = imdb_movie_list.filter(movie => movie.slice(0,3) == "The");
console.log(the_movie_list);
```

- See other iterators

# Objects

- JavaScript *objects* are containers that can store **data** and **functions**.
- The data in an object is **not ordered**, which can only be accessed by using its associated *key*.
- Create an object with **key-value** pairs:

```
let player1 = {
  name: "Stephen Curry",
  dob: "March 14, 1988",
  height: 1.91,
  weight: 86,
  shoot(){
    return "3-pointer";
  }
};
```

```
let player2 = {
  name: "LeBron James",
  dob: "December 30, 1984",
  height: 2.03,
  weight: 118,
  shoot(){
    return "dunk";
  }
};
```

```
console.log(player1.name);  // dot notation
console.log(player2["name"]); // bracket notation
console.log(player2.weight - player1.weight);
console.log(player1.shoot());
console.log(player2.shoot());
```

```
Stephen Curry
LeBron James
32
3-pointer
dunk
```

# this keyword

- In Javascript, *this* refers to the object we call it inside.
- Add object property value on the fly

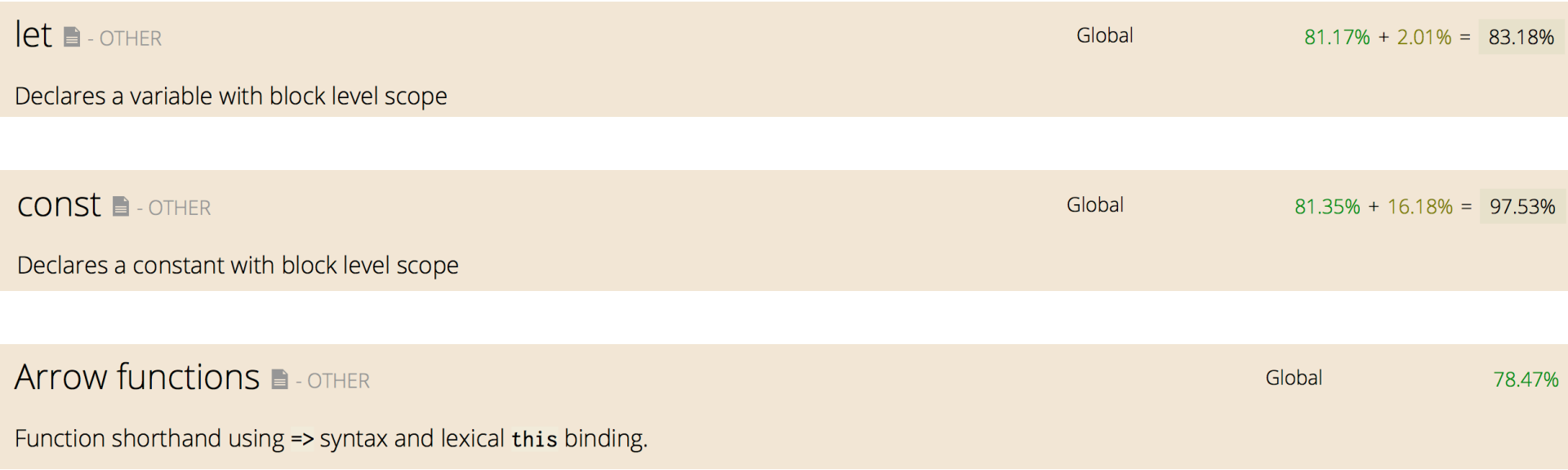
```
let player2 = {
  name: "LeBron James",
  dob: "December 30, 1984",
  height: 2.03,
  weight: 118,
  shoot(){
    return "dunk";
  },
  info(){
    return `${this.name} is ${this.height}m and ${this.weight}kg.`;
  }
};
```

```
console.log(player2.info());
player2.draft_year = 2003;
console.log(player2.draft_year);
```

```
LeBron James is 2.03m and 118kg.
2003
```

# Browser Compatibility (caniuse.com)

- let, const vs. var
- Arrow function
- Spring concatenation
- **Sections on NPM is optional**



# Modules (Optional)

- Useful when you write Modules of your own

# jQuery and AJAX

- Next lecture

# In-class Exercise

- Implement some code examples from this lecture