

JavaScript

Lecture 4a (Arrays, Inheritance)

Waterford Institute of Technology

June 20, 2016

John Fitzgerald

JavaScript Introduction

Topics discussed this presentation

- Arrays
- Prototypal inheritance

Arrays

Create

- Not necessary to declare size when constructing
 - Create easily using array literal
 - Grow automatically
 - Locate values by key
- Access using [] operator

```
// literal method
const cars = ['Ford', 'Honda', 'Nissan', 'Peugot', 'Toyota'];
console.log(cars[0]); //Output: Ford
```

```
//using new
const sameCars = new Array('Ford', 'Honda', 'Nissan', 'Peugot', 'Toyota');
console.log(cars[0]); //Output: Ford
```

Arrays

Iterate

- **for** loop easy method of iterating
- Size array: use **length** property

```
const cars = ['Ford', 'Honda', 'Nissan', 'Peugot', 'Toyota'];  
for(let i = 0; i < cars.length; i += 1)  
{  
  console.log(cars[i]);  
}
```

```
//Output  
Ford  
Honda  
Nissan  
Peugot  
Toyota
```

Arrays

Iterate - forEach

```
function logArrayElements(element, index, array)
{
  console.log('a[' + index + '] = ' + element);
}

var cars = [ 'Ford', 'Honda', 'Nissan', 'Peugot'];

cars.forEach(logArrayElements);
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>JavaScript</title>
  </head>
  <body>
    <script src="array.js"></script>
  </body>
</html>
```

a[0] = Ford
a[1] = Honda
a[2] = Nissan
a[3] = Peugeot

Arrays

Iterate - forEach

```
var cars = [ 'Ford', 'Honda', 'Nissan', 'Peugot'];

cars.forEach(function(element, index, array) {
    console.log('a[' + index + '] = ' + element);
});
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>JavaScript</title>
  </head>
  <body>
    <script src="array.js"></script>
  </body>
```

```
a[0] = Ford
a[1] = Honda
a[2] = Nissan
a[3] = Peugeot
```



Arrays

Methods

Selection of array methods:

- **length** : provides number elements in array
- **join** :converts elements to string & concatenates
- **reverse**: reverses order of elements
- **push**: adds element(s) end array
- **pop** : removes element(s) end array
- **unshift**:adds element beginning array
- **shift**: removes element beginning array
- **sort**: sorts array

Arrays

Methods: *length*, *join*

```
const greet = ['hello', 'ictskills'];  
  
const length = greet.length; // => 2
```

```
const str1 = greet.join(); // => hello,ictskills  
const str2 = greet.join(separator=' '); // => hello ictskillls
```


Arrays

Methods: *reverse*, *push*, *pop*

```
const greet = ['hello', 'ictskills'];  
  
console.log(greet.reverse()); // => ['ictskills', 'hello']
```

```
greet.push('2016');  
console.log(greet); // => ['hello', 'ictskills', '2016']
```

```
greet.pop();  
console.log(greet); // => ['hello', 'ictskills']
```

Arrays

Methods: *unshift*, *shift*

```
const greet = ['hello', 'ictskills'];  
greet.unshift('2016');  
console.log(greet); // => ['2016', 'hello', 'ictskills']
```

```
greet.shift();  
console.log(greet); // => ['hello', 'ictskills']
```

Arrays

Methods: *sort*

```
// The default sort order is according to string Unicode code points.  
const numbers = [6, 11, 22, 43, 19, 10];  
numbers.sort();  
console.log(numbers); // => [10, 11, 19, 22, 43, 6]
```

```
// Provide customized comparator function to sort numbers in ascending order.  
function compare(a,b) {  
    return a - b;  
}  
  
numbers.sort(compare);  
console.log(numbers); // => [6, 10, 11, 19, 22, 43]
```

Arrays

Element types

- Array elements may be different types

```
const cars = ['Ford', 'Honda', 'Nissan', 'Peugot'];
const manual = {
  title: 'Fix Me',
  author: 'H. Wrench',
};
cars.push(manual);
cars.push('Lexus');
cars.shift();
for (let i = 0; i < cars.length; i += 1)
{
  console.log(cars[i]);
}
```

```
Honda
Nissan
Peugot
Object {title: "Fix Me", author: "H. Wrench"}
Lexus
```

JavaScript

Object v Array

```
// Objects: comprise key:value pairs
```

```
const book = {};  
book.title = 'Java';  
book.author = 'Chapman';  
console.log(book);
```

```
// Retrieval:
```

```
console.log(book.title); // => Java
```

```
// Arrays: Use for numerically indexed data
```

```
const cars = [];  
cars[4] = 'Toyota';
```

```
// Retrieval:
```

```
console.log(cars.length); // => 5  
console.log(cars[0]); // => undefined  
console.log(cars[4]); // => Toyota: length increases automatically  
console.log(cars[6]); // => undefined  
console.log(cars.length); // => 5: No array bounds error
```

JavaScript Inheritance

ES5 inheritance example

```
const shape = {  
  xPosition: 0.0,  
  yPosition: 0.0,  
};  
  
const circle = Object.create(shape);  
  
circle.area = function () {  
  return Math.round(Math.PI * Math.pow(this.radius, 2));  
};  
  
circle.xPosition = 100;  
circle.radius = 50;  
  
console.log('area ' + circle.area()); // 7854  
console.log('xPosition ' + circle.xPosition); // 100  
console.log('yPosition ' + circle.yPosition); // 0 (default)
```

JavaScript Inheritance

ES6 simulates classical inheritance

```
class Shape {  
  constructor(xPosition, yPosition) {  
    this.xPosition = xPosition;  
    this.yPosition = yPosition;  
  }  
}  
  
class Circle extends Shape {  
  constructor(xPosition, yPosition, radius) {  
    super(xPosition, yPosition);  
    this.radius = radius;  
  }  
  
  area() {  
    return Math.round(Math.PI * Math.pow(this.radius, 2));  
  }  
}  
  
const circle = new Circle(100.0, 100.0, 50.0);  
console.log('area ' + circle.area()); // 7854
```

JavaScript

Presentation summary

- Arrays
 - Store multiple elements in single variable.
 - Elements may be different types.
 - Rich set Array methods available.
- Inheritance
 - Prototypal
 - Syntactic sugar - ES6



Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚD TEICNEOLAÍOCHTA PHORT LÁIRGE

