

# Functions

## Part II



# Let's review... basic function

```
function myFunction() {  
  alert('hello');  
}
```

```
myFunction()  
// calls the function once
```



# Let's review... function parameters

```
function greeting(name) {  
    alert('hello ' + name);  
}
```

```
greeting('John');  
// calls the function
```



# Callbacks



# Callback Function

A callback function is a function which is:

- passed as an argument to another function, and,
- is invoked after some kind of event.

Callback function is one of the most used pattern in JavaScript.



# Callback Function - example

```
function processUserInput(callback) {  
  var name = prompt('Please enter your name.');
```

```
  callback(name);  
}
```

```
function greeting(name) {  
  alert('Hello ' + name);  
}
```

```
processUserInput(greeting);
```



# Exercise

1. Create a function `getEven(array, callback)` which as a first argument accepts array, and as a second one callback. Function should return new array with only even numbers (tips: `even = number % 2 == 0`);



# Functions (methods) in Objects





# Let's review... objects

```
var obj = {  
  a: "hello world",  
  b: 42,  
  c: true  
};
```

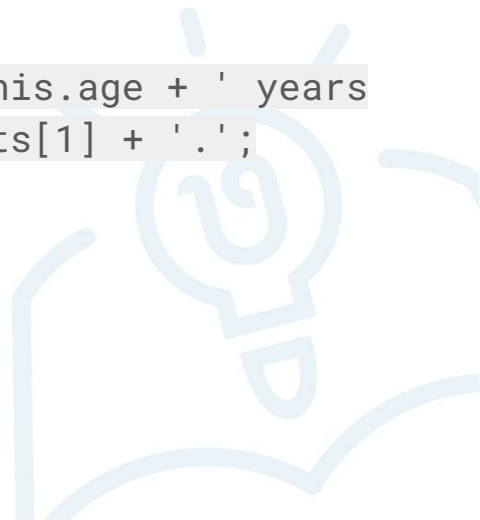
```
obj.a;      // "hello world"  
obj.b;      // 42  
obj.c;      // true
```

```
obj["a"];   // "hello world"  
obj["b"];   // 42  
obj["c"];   // true
```



# Create methods in object

```
var person = {  
  firstName: 'Bob',  
  lastName: 'Smith',  
  age: 32,  
  gender: 'male',  
  interests: ['music', 'skiing'],  
  bio: function() {  
    return this.firstName + ' ' + this.lastName + ' is ' + this.age + ' years  
old. He likes ' + this.interests[0] + ' and ' + this.interests[1] + '.';  
  },  
  greeting: function() {  
    return 'Hi! I\'m ' + this.firstName + '.';  
  }  
};
```



# Console.log() these...

```
person.firstName
```

```
person.age
```

```
person.interests[1]
```

```
person.bio()
```

```
person.greeting()
```



# Updating object properties

```
person.age = 45;
```

```
person.firstName = 'Daisy';
```

```
person['lastName'] = 'Duck';
```

```
person['gender'] = 'female';
```



# Creating new object properties

```
person['eyes'] = 'hazel';
```

```
person.farewell = function() { return "Bye everybody!"; }
```

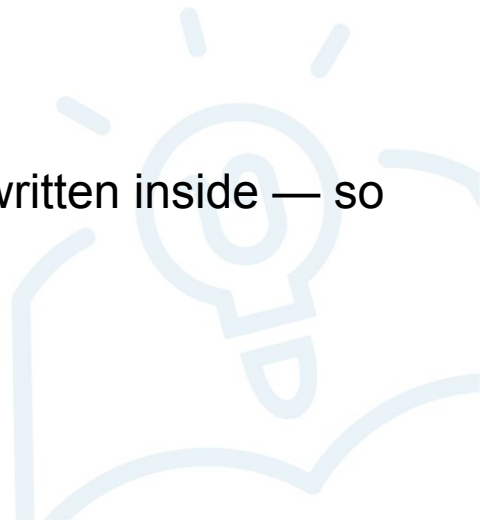


# What is "this"?

You may have noticed something slightly strange in our methods. Look at this one for example:

```
greeting: function() {  
  alert('Hi! I\'m ' + this.firstName + '.');  
}
```

The **this** keyword refers to the current object the code is being written inside — so in this case this is equivalent to **person**.



# The "this" example

```
var person1 = {  
  name: 'Chris',  
  greeting: function() {  
    alert('Hi! I\'m ' + this.name + '.');  
  }  
}
```

```
var person2 = {  
  name: 'Brian',  
  greeting: function() {  
    alert('Hi! I\'m ' + this.name + '.');  
  }  
}
```



# Exercise

1. Create a person object that represents you. Add these properties:

- firstName,
- lastName,
- yearOfBirth,
- school,
- occupation

Add these methods:

- fullName(combine first and last name properties),
- in2050IWillBe(use yearOfBirth to calculate)

