

JavaScript Functions

Our Goals

- Identify the need for functions
- Identify functions
- Create functions
- Call functions

Introduction to functions

- They are a reusable collection of statements
- The bread and butter of JS
- They are tools to structure large programs
- They reduce repetition
- They can associate names with subprograms

Introduction to functions

- The function of a bridge could be to provide access over water

Creating new words in human-language is, often fun, but probably bad practice. In programming, it is absolutely essential

How do they work?

```
var sayHello = function () {  
    console.log( "Hello!" );  
};  
  
var doSomethingFancy = function () {  
    console.log( "Ooooh, fancy!" );  
};  
  
sayHello();  
doSomethingFancy();
```

Defining a function

```
var makeSilentNoise = function () {  
    console.log( "Making 'noise'" );  
};  
// A Function Expression  
  
function makeSilentNoise () {  
    console.log( "Making 'noise'" );  
};  
// A Function Declaration
```

Parameters

```
var sayHello = function ( name ) {  
    var greeting = "Hello " + name;  
    console.log( greeting );  
};  
  
sayHello();  
sayHello( "Groucho" );
```

Parameters

```
var squareNumber = function (x) {  
    var square = x * x;  
    console.log( square );  
};
```

```
squareNumber( 12 );  
squareNumber( 45 );
```


Return Values

```
var squareNumber = function (x) {  
    var square = x * x;  
    return square;  
};  
  
var squareOfFour = squareNumber(4);  
var squareOfTwelve = squareNumber(12);  
  
squareNumber(4) + squareNumber(12);
```

Return Values

- Return means that a function has a result
- It will leave the function immediately!

```
var sayHello = function () {  
    return "No.";   
    console.log( "Hi!" );  
};  
  
sayHello();
```

Variable Scope

```
var someVariableOutside = "Outside";

var doSomethingFancy = function () {
    var someVariableInside = "Inside";
};

console.log( someVariableOutside );
// => "Outside"
console.log( someVariableInside );
// => undefined
```

Global vs. Local Scope

```
var globalResult;  
  
var addSomeNumbers = function (x, y) {  
    var localResult = x + y;  
    globalResult = x + y;  
};  
  
addSomeNumbers( 10, 2 );  
  
localResult; // => undefined  
globalResult; // => 12
```

Coding Conventions

```
var addTwoNumbers = function(x,y){return x+y};;
```

```
var addTwoNumbers = function (x, y) {  
  return x + y;  
};
```

```
var addTwoNumbers = function (x, y) {  
  return x + y;  
}; // The only good one
```

Passing in variables

```
var addTwoNumbers = function (x, y) {  
    return x + y;  
};
```

```
var firstNumber = 10;
```

```
addTwoNumbers( firstNumber, 4 );
```

```
addTwoNumbers( firstNumber, 6 );
```

Our Goals

- Identify the need for functions
- Identify functions
- Create functions
- Call functions

Have a crack at **these**
exercises