

```
var x = 'hello world';
```

Variables store changeable values
in memory for later reference

What kinds of data can we store in variables?

Strings, Numbers, Functions, Arrays, Objects & more...

```
var x = 'hello world' ;
```

Strings - for textual data
(always enclosed in quotes)

```
var x = 1 ;
```

Numbers - for integers and floats (numbers with decimal points), used in math operations and calculations

```
var x = 1;  
var y = x + x;  
var z = x * 2;
```

(more operations with number variables)

```
var list = [1, 'two', 3.5];
```

Arrays - for sequences of data
(items can be any data type)

```
console.log(list[0]);
```

```
// 1
```

```
console.log(list[1]);
```

```
// 'two'
```

```
console.log(list[2]);
```

```
// 3.5
```

Accessing data from arrays (count starts at 0)

```
var person = {  
  name: 'Chris',  
  email: 'ccuellar2019@g.ucla.edu',  
  id: 123  
};
```

Objects - store data as pairs of keys with values


```
console.log(person.name);  
// 'Chris'  
console.log(person.email);  
// 'ccuellar2019@g.ucla.edu'  
console.log(person.id);  
// 123
```

Accessing data from Objects

```
var multiply = function (x) {  
    return x * 2;  
};  
  
multiply(2);
```

Functions - do stuff with any data type

```
var multiply = function (x)
```



Defining the function

```
    return x * 2;
```

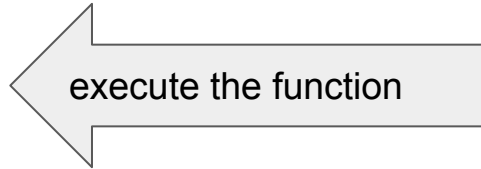
```
};
```

```
multiply(2);
```

Functions - do stuff with any data type

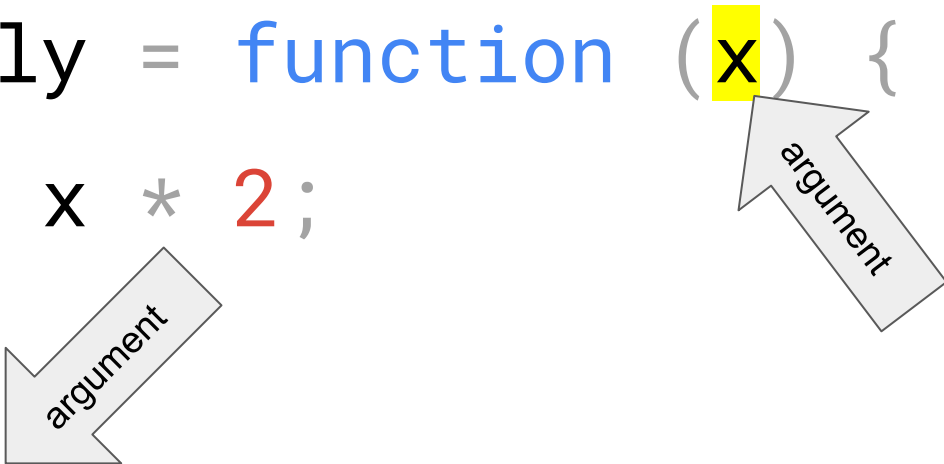
```
var multiply = function (x) {  
    return x * 2;  
};
```

```
multiply(2);
```



Functions - do stuff with any data type

```
var multiply = function (x) {  
    return x * 2;  
};  
multiply(2);
```



The diagram consists of two gray arrows with the word "argument" written inside them. One arrow points from the parameter 'x' in the function definition to the closing curly brace '}' of the function. The other arrow points from the value '2' in the function call 'multiply(2);' to the opening parenthesis '(' of the function call.

Arguments - values passed into a function, that are usually modified in some way or used to produce some other output

```
var hideParagraphs = function () {  
    $('p').hide();  
};  
$('button').click(hideParagraphs);
```

Any data type can be passed into functions,
including other functions (a.k.a *Callbacks*)

```
$('button').click(function () {  
    $('p').hide();  
});
```

Callback functions can also be defined inline

Logic


```
var x = 1;  
if (x > 100) {  
    console.log('Over a hundred');  
} else if (x > 50) {  
    console.log('Over fifty');  
} else {  
    console.log('Less than fifty');  
}
```

If...Else statements run things depending on the state of conditions that you define

Iterating

(& other array functions)

```
var names = ['Lisa', 'Mina', 'Jieun',  
'April', 'Jarrett'];
```

```
names.forEach(function(name) {  
    console.log(name);  
});
```

Arrays have built-in functions that let you iterate through and manipulate the stored values

```
var values = [1, 2, 3];  
var multiplied = values.map(function(x) {  
    return x * 2;  
})  
console.log(multiplied);  
// [2, 4, 6]
```

Map can transform an array using whatever algorithm you define