Javascript Arrays

Objects & Arrays

- The fundamental building blocks of java programs are
    - Object
    - Array
    - Functions

Creating & Using Objects

- The simplest way to create an object is either through:
    - the *object constructor*
    - the shorthand syntax known as *object literal.*
- Objects are *unordered key/value pairs*.
    - The *key* is formally known as a property and the value can be any valid JavaScript type, even another object.
- To create or access a property on an object, we use
    - *dot notation*
    - *bracket notation*

Creating Objects ...

Object Constructor

```
var person1 = new Object;

person1.firstName = "John";
person1.lastName = "Doe";
```

Object Literal

```
var person2 = {
  firstName: "Jane",
  lastName: "Doe"
};
```

Using Objects...

Dot Notation

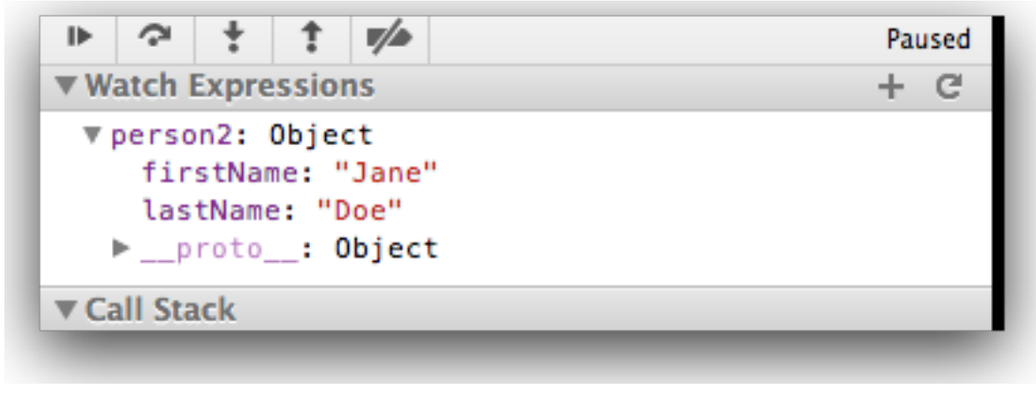
```
person1.firstName = "John";
person1.lastName = "Doe";
```

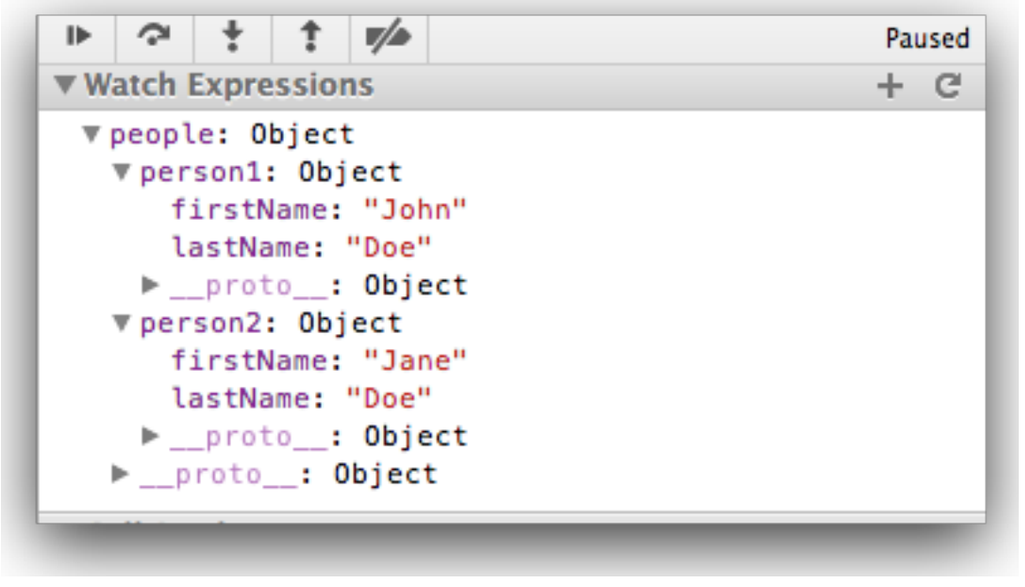
Bracket Notation

```
person['firstName'] = "Mary";
person['lastName'] = "Smith";
```

Tracing Objects

- Object structure and contents can be explored in detail in Chrome Developer Tools





Creating & Using Arrays

- Arrays are a type of object that are ordered by the index of each item it contains.

- The index starts at zero and extends to however many items have been added, which is a property of the array known as the "length" of the array.

- Similar to objects, an array can be created with the array constructor or the shorthand syntax known as array literal.

Creating Arrays...

Array Constructor

```
var foo = new Array;
```

Array Literal

```
var bar = [];
```

Creating Arrays with Dimensions

Array Constructor

```
var foo = new Array(5);
```

- In general, dont do this. Better to create empty array:

Array Literal

```
var bar = [];
```

- And the append items via as needed

```
bar.push(23);
bar.push(32);
```

Using Arrays...

- Insertion into arrays can be via:
  - [] notation (like Java)
  - Using ‘push’ which appends to the end of the array

```
var foo = [];

foo.push("a");
foo.push("b");

alert( foo[ 0 ] ); // => a
alert( foo[ 1 ] ); // => b

alert( foo.length ); // => 2

foo.pop();

alert( foo[ 0 ] ); // => a
alert( foo[ 1 ] ); // => undefined

alert( foo.length ); // => 1
```