# Javascript

## First Javascript Lesson

Anonymous 01/31/15(Sat)09:32:46 No.46348970 ▶ >>46349004

>>46346548

```
> '5' - 3
2         // weak typing + implicit conversions * headaches
> '5' + 3
'53'      // Because we all love consistency
> '5' - '4'
1         // string - string * integer. What?
> '5' + + '5'
'55'
> 'foo' + + 'foo'
'fooNaN' // Marvelous.
> '5' + - '2'
'5-2'
> '5' + - + - - + - - + + - + - + - + - - - '-2'
'52'      // Apparently it's ok

> var x * 3;
> '5' + x - x
50
> '5' - x + x
5       // Because
```

# Including javascript to your HTML file

- Similar to how you add CSS to an html file
  - JS can be a part of the document using the <script> tag, which works identically to <style>
  - Or you can link an external script with <script src="filename.js"></script>, with nothing between the tags, which works the same as <link> for CSS
- Problem with JS is pages load and run the JS before it can display the page, which takes time
  - To minimize this, use keywords to let the page download the JS without stopping the page rendering process
  - ```
    <script type="text/javascript" src="path/to/script1.js" async></script>
    <script type="text/javascript" src="path/to/script2.js" async></script>
    ```
    - This will let the browser continue to parse html while this is downloading. This means that it is possible that script2 can be downloaded and run before script1
  - ```
    <script type="text/javascript" src="path/to/script1.js" defer></script>
    <script type="text/javascript" src="path/to/script2.js" defer></script>
    ```
    - This will download them while the HTML is being loaded, but not run these until the page

# Javascript Types

- Primitive types (integer, float, String) are all one type
- Everything else is all one type
- They’re both the same type -- var
- Numbers, Strings, arrays, objects, even functions are all the same type
- Although you can’t specify the type of it, you can use the typeof operator to see what type is being stored in a variable at a given time
  - var x = 3;
  - typeof x; //returns number
- Technically they are different and have different properties, but you can’t force a variable to stay as a different type

# Basics of JS

- Many similarities to C++/Java/other languages in that family
  - Semicolons at the end of each line, brackets around block statements
  - Use // for single-line comments, /* comment */ for multi-line
  - I’m just trying to make this slide look like it has more information

# Hello World! in Javascript

- Finally we learn a language where we can write this in code (I don’t think HTML really counts)
  - Assuming the file the javascript is saved in is included in the HTML, you can make the page display “Hello World!” with the following JS:
  - ```
    var myHeading = document.querySelector(“h1”);

    myHeading.textContent = “Hello World!”;
    ```
- This will find an h1 tag in the HTML page, and if it exists, set it to myHeading, then make the text displayed within the tag equal to “Hello World!”
    ```

# Loops!

- For loops -- basically identical to Java/C++
  - `for(i=0;i<10;i++){console.log(i);}`
  - Don’t have to specify var i in the first statement like in Java/C++
  - Can also put multiple statements in the first line, i.e. for(i=0, j=10;etc.;etc){};
- Can also do for loop in a way similar to python syntax
  - Given an object (dictionary/hash array), you can loop through its properties:
  - for(x in object){etc. Etc. etc.}
- While loops -- literally identical (as far as I can tell)
  - `while(boolean condition){do stuff}`
- Do-while, same as while loop, code block is ALWAYS run at least once
  - do{code block}while(boolean condition)

# Javascript Functions

- `function name(parameter1, parameter2){}`
  - Function keyword part of definition, required
  - Can just do stuff (like void in java), or return a type (but you don’t have to specify the return type)

# Javascript Objects

- `var myObject = new Object();`
  - `myObject.text = “hi”;`
  - `myObject.asldfkjasldfk; //undefined`
  - `myObject[text]; //returns ‘hi’`
  - `myObject.obj = new Object();`
- `myObject = {“text”: “hi”, “obj”, “Object”};`
- `for(x in myObject){console.log(myObject[x]);}`

# Equality!

- == vs ===
  - == - abstract equality operator
  - === - strict equality operator
  - 3=='3' //return true
  - 3==='3' //returns false
  - true=='1' //true
  - true==='1' //false
  - You can also do !=, !==
  - "This is a string."==new String("This is a string.") //true
  - "This is a string."===new String("This is a string.")

# Constructor Function

- Syntax is identical to a normal javascript function
- ```
  function car(make, model year){
      This.make = make;
      This.model = model;
      This.year = year;
  }

  Var car = function(make, model, year){etc.};
  ```
- ```
  Var mycar = new car(“honda”, “civic”, 2008);
  ```

# Custom Objects

- Var class = function() {};
  - myClass = new class()

```
Var Person = function(name){
    This.name = name;
    console.log(“Object created”);
}
Function Person(name){etc.};

Person.prototype.sayHello(){console.log(this.name + “ says hi”);}
Person1 = new Person(“Bob Smith”);
Person1.sayHello();
```