Prototype

Object.prototype and Prototype: JavaScript Power Tools

Amy Hoy, www.slash7.com

```
Object.extend = function(destination, source) {
  for (var property in source) {
    destination[property] = source[property];
  }
  return destination;
}
```

JavaScript is a real ~~boy~~ language.

everythings an object... everything.

really. everything.

creating functions

```
function noArgument() {
    // do stuff
    alert("I can't argue; I have no arguments")
}

function simpleFunction(arg1,arg2,arg3) {
    // do stuff here
    return arg1 + arg2 + arg3
}

noArgument()
// --> [alert] I can't argue; I have no arguments

simpleFunction(1,2,3)
// --> 6
```

look ma, no primitives!

```
var string = "Hello, I'm a string!"
// --> "Hello, I'm a string!"

string.length
// --> 20

"Hello, I'm a string!".length
// --> 20
```

creating objects & messing with properties

```
var newObject = {}
// --> {} (empty object!)

newObject.foo = "bar"
// --> "bar"

newObject.foo
// --> "bar"

var anotherObject = {foo:"bar", baz:"bat"}
// --> {foo:"bar", baz:"bat"}

anotherObject.baz
// --> "bat"

anotherObject.baz = "zort"
// --> "zort"
```

creating objects

fun with the object literal

```
var foo = {
    bar:"baz",
    zort:"narf",
    aNumber:5,
    anArray:['banana','plaintain','ugli fruit'],
    doStuff:function() {
        alert("I'm doing stuff!")
    }
}
// --> [Object object]
```

creating constructors

```
function anyFunction() {}

var object = new anyFunction()
// --> [Object object]
```

creating constructors

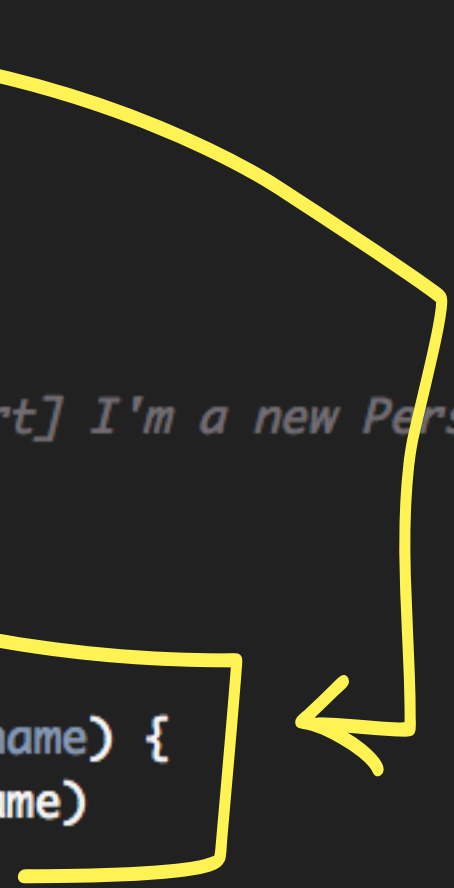
```
function Foo() {
    // use this keyword inside function constructors
    this.bar = "baz"
    // run arbitrary code on construction
    alert("I'm a new Foo!")
}

var foo = new Foo()
// --> [Object object] & [alert] I'm a new Foo!

foo.bar
// --> "baz"
```

extending objects using prototype

```
function Person() {
    this.greeting = "Howdy"
    this.name = ""
    alert("I'm a new Person!")
}

var amy = new Person() // --> [alert] I'm a new Person!
amy.greeting // --> "Homo sapiens"
amy.name = "Amy" // --> "Amy"

Person.prototype.name = "Bobby"
Person.prototype.greet = function(name) {
    alert(this.greeting + ", " + name)
}

amy.name // --> "Amy"
amy.greet("Will") // --> [alert] Howdy, Will
```

extending objects using prototype

```
function Person() {
    this.greeting = "Howdy"
    this.name = ""
    alert("I'm a new Person!")
}

var amy = new Person() // --> [alert] I'm a new Person!
amy.greeting // --> "Homo sapiens"
amy.name = "Amy" // --> "Amy"

Person.prototype.name = "Bobby"
Person.prototype.greet = function(name) {
    alert(this.greeting + ", " + name)
}

amy.name // --> "Amy"
amy.greet("Will") // --> [alert] Howdy, Will
```

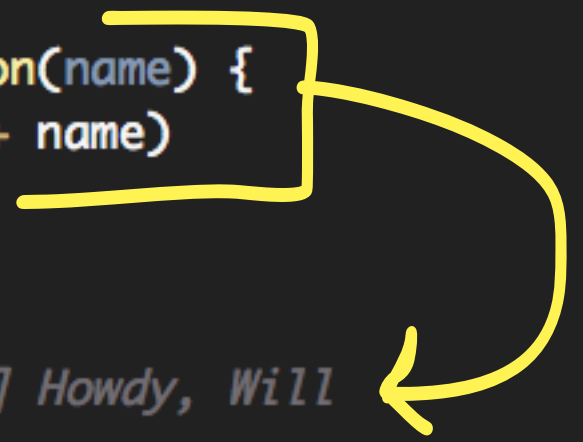


extending objects using prototype

```
function Person() {
    this.greeting = "Howdy"
    this.name = ""
    alert("I'm a new Person!")
}

var amy = new Person() // --> [alert] I'm a new Person!
amy.greeting // --> "Homo sapiens"
amy.name = "Amy" // --> "Amy"

Person.prototype.name = "Bobby"
Person.prototype.greet = function(name) {
    alert(this.greeting + ", " + name)
}

amy.name // --> "Amy"
amy.greet("Will") // --> [alert] Howdy, Will
```

extending objects using prototype

```
function Person() {
    this.greeting = "Howdy"
    this.name = ""
    alert("I'm a new Person!")
}

var amy = new Person() // --> [alert] I'm a new Person!
amy.greeting // --> "Homo sapiens"
amy.name = "Amy" // --> "Amy"

Person.prototype.name = "Bobby"
Person.prototype.greet = function(name) {
    alert(this.greeting + ", " + name)
}

amy.name // --> "Amy"
amy.greet("Will") // --> [alert] Howdy, Will
```



extending objects using prototype

```
var bobby = new Person() // --> [alert] I'm a new Person!
bobby.name // --> "Bobby"

var jules = new Person() // --> [alert] I'm a new Person!
jules.name = "Jules" // --> "Jules"
jules.greeting = "Hiya" // --> "Hiya"

jules.greet("Thomas") // --> [alert] Hiya, Thomas
bobby.greet("Thomas") // --> [alert] Howdy, Thomas
```

extending objects using prototype

```
var bobby = new Person() // --> [al
bobby.name // --> "Bobby"

var jules = new Person() // --> [alert] I'm a new Person!
jules.name = "Jules" // --> "Jules"
jules.greeting = "Hiya" // --> "Hiya"

jules.greet("Thomas") // --> [alert] Hiya, Thomas
bobby.greet("Thomas") // --> [alert] Howdy, Thomas
```

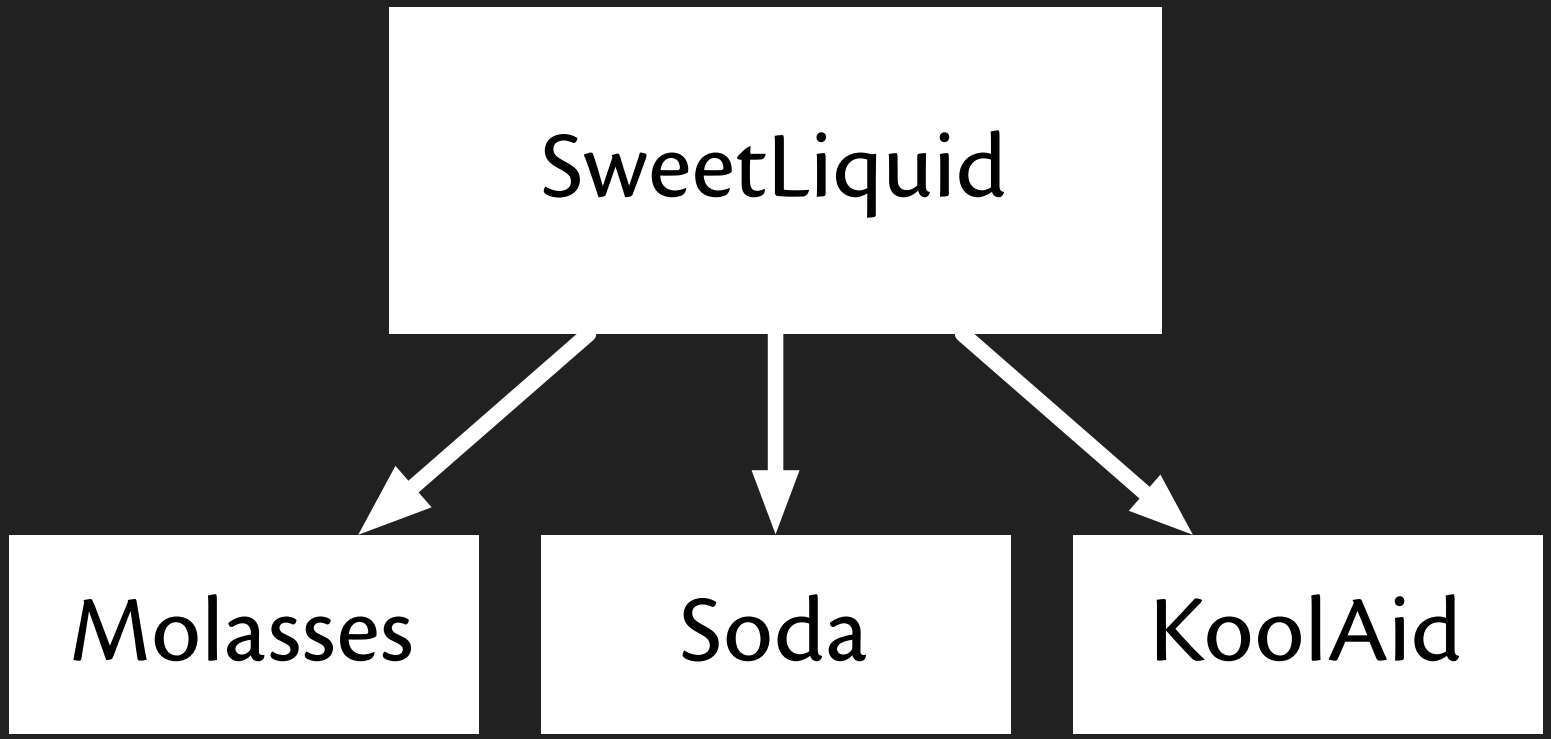
```
Person.prototype.name = "Bobby"
```

extending objects using prototype

```
var bobby = new Person() // --> [alert] I'm a new Person!
bobby.name // --> "Bobby"

var jules = new Person() // --> [alert] I'm a new Person!
jules.name = "Jules" // --> "Jules"
jules.greeting = "Hiya" // --> "Hiya"

jules.greet("Thomas") // --> [alert] Hiya, Thomas
bobby.greet("Thomas") // --> [alert] Howdy, Thomas
```

```
function Person() {
    this.greeting = "Howdy"
    //...
}
```

more on inheritance



creating a prototype

```
function SweetLiquid() {
    this.volume = 500
    this.unit = 'ml'
    this.drink = function() { this.volume-- }
    this.sweetnessRating = .5
}
```

creating objects...

```
function Soda() {}
function Molasses() {
    this.viscosity = "low"
}

function KoolAid() {
    alert("OHH YEAH!")
    this.mascot = "awesome"
    this.unit = 'quart'
    this.volume = '4'
}
```

creating objects...

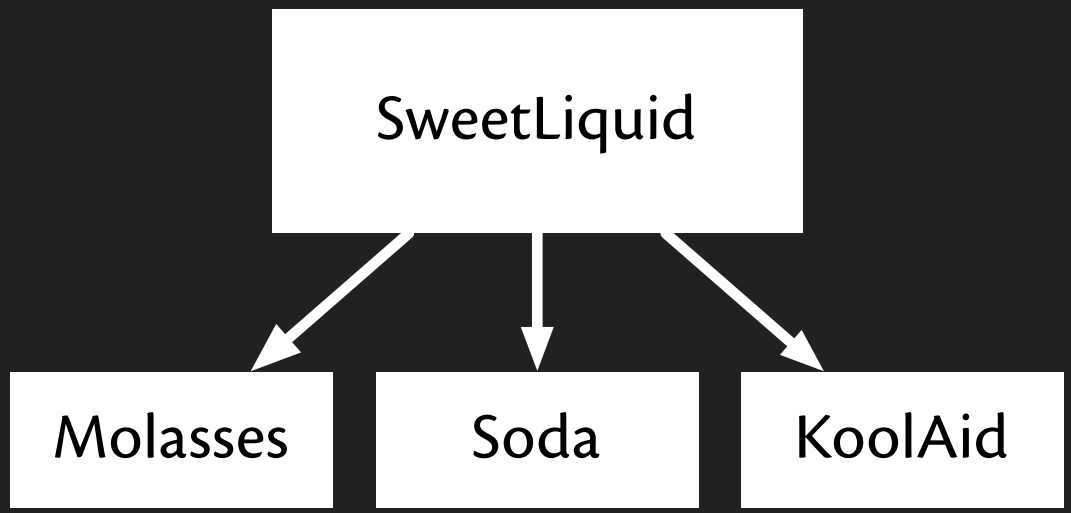
```
function Soda() {}
function Molasses() {
    this.viscosity = "low"
}

function KoolAid() {
    alert("OHH YEAH!")
    this.mascot = "awesome"
    this.unit = 'quart'
    this.volume = '4'
}
```

creating objects...

```
function Soda() {}
function Molasses() {
    this.viscosity = "low"
}

function KoolAid() {
    alert("OHH YEAH!")
    this.mascot = "awesome"
    this.unit = 'quart'
    this.volume = '4'
}
```

setting the prototype

```
Soda.prototype = new SweetLiquid
Molasses.prototype = new SweetLiquid
KoolAid.prototype = new SweetLiquid
```



let's try it out!

```
var cola = new Soda // --> [Object object]
cola.volume // --> 500
cola.unit // --> "ml"

var strawberryKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"
strawberryKoolAid.mascot // --> "awesome"
strawberryKoolAid.unit // --> "quart"
strawberryKoolAid.volume // --> 4
```

let's try it out!

```
var cola = new Soda // --> [Object object]
cola.volume // --> 500
cola.unit // --> "ml"

var strawberryKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH
strawberryKoolAid.mascot // --> "awesom
strawberryKoolAid.unit // --> "quart"
strawberryKoolAid.volume // --> 4
```

```
function SweetLiquid() {
    this.volume = 500
    this.unit = 'ml'
    //...
}
```

let's try it out!

```
var cola = new Soda // --> [Object object]
cola.volume // --> 500
cola.unit // --> "ml"

var strawberryKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"
strawberryKoolAid.mascot // --> "awesome"
strawberryKoolAid.unit // --> "quart"
strawberryKoolAid.volume // --> 4
```

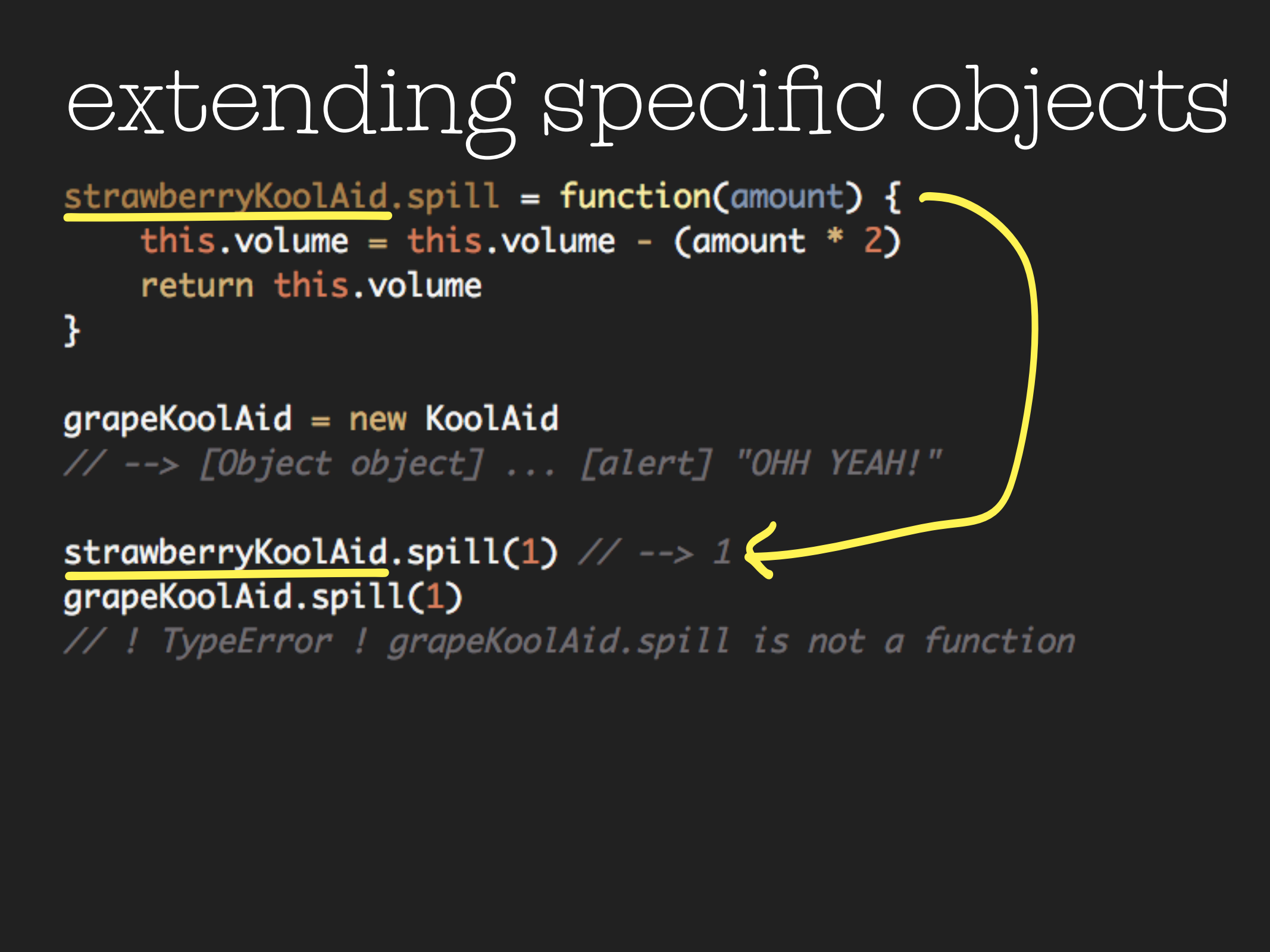
```
function KoolAid() {
    alert("OHH YEAH!")
    this.mascot = "awesome"
    this.unit = 'quart'
    this.volume = 4
}
```

extending specific objects

```
strawberryKoolAid.spill = function(amount) {
    this.volume = this.volume - (amount * 2)
    return this.volume
}

grapeKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"

strawberryKoolAid.spill(1) // --> 1
grapeKoolAid.spill(1)
// ! TypeError ! grapeKoolAid.spill is not a function
```

extending specific objects

```
strawberryKoolAid.spill = function(amount) {
    this.volume = this.volume - (amount * 2)
    return this.volume
}

grapeKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"

strawberryKoolAid.spill(1) // --> 1
grapeKoolAid.spill(1)
// ! TypeError ! grapeKoolAid.spill is not a function
```

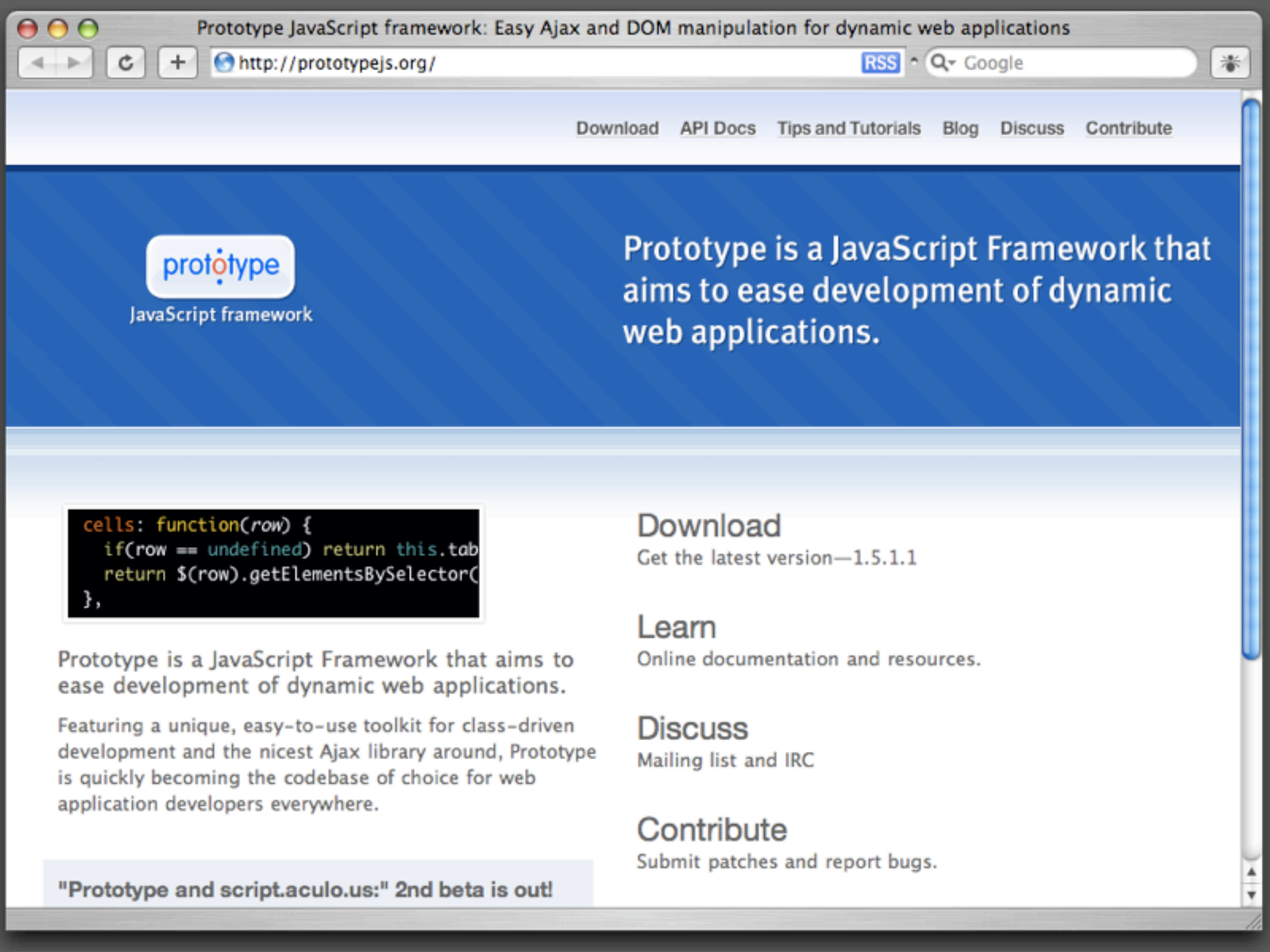


extending specific objects

```
strawberryKoolAid.spill = function(amount) {
    this.volume = this.volume - (amount * 2)
    return this.volume
}

grapeKoolAid = new KoolAid
// --> [Object object] ... [alert] "OHH YEAH!"

strawberryKoolAid.spill(1) // --> 4
grapeKoolAid.spill(1)
// ! TypeError ! grapeKoolAid.spill is not a function
```

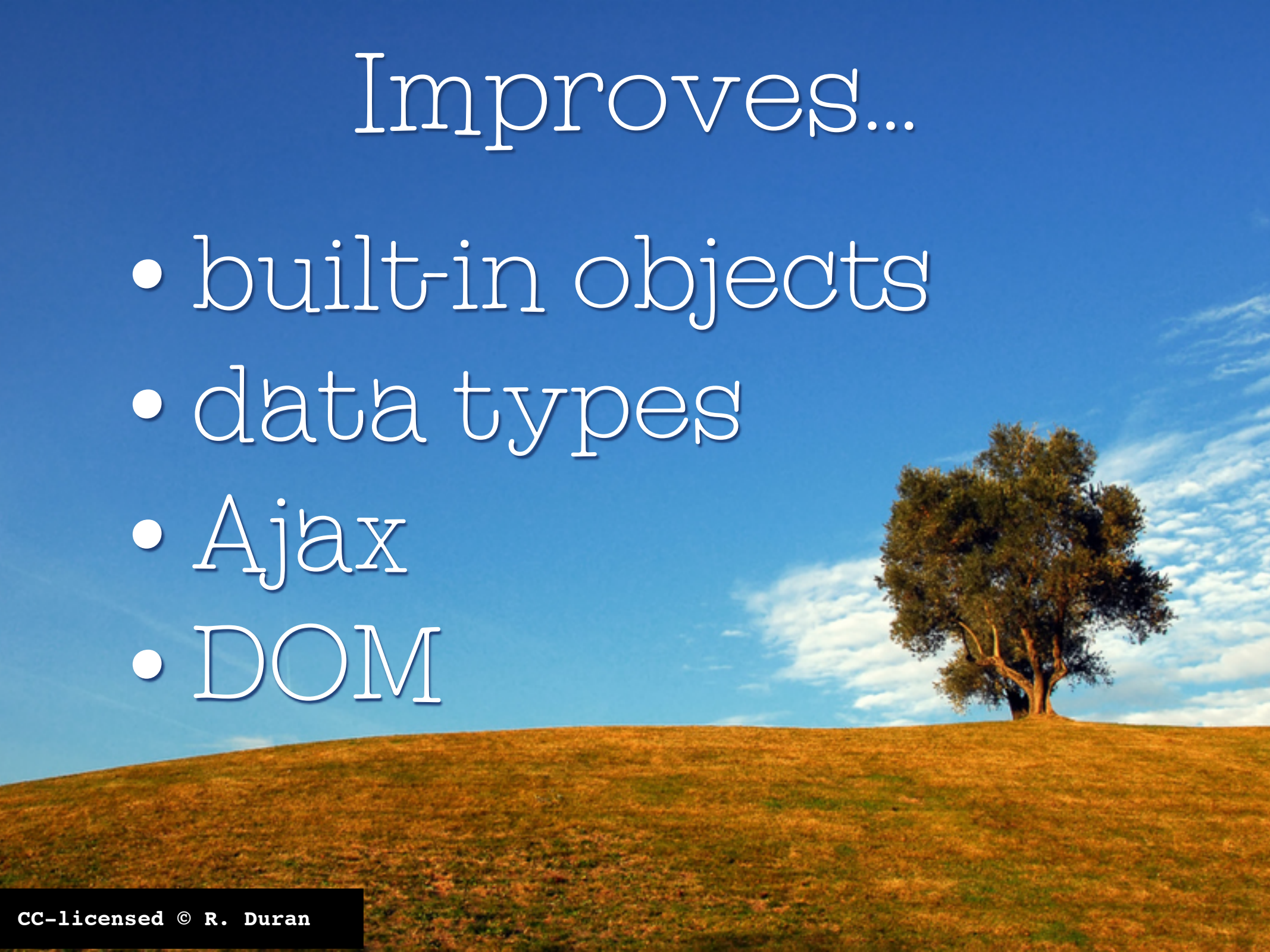
!!! FAIL


little p, big P
KAUFMANN BOLD 36
CC-licensed © mattdesmond


Prototype JavaScript framework: Easy Ajax and DOM manipulation for dynamic web applications
http://prototypejs.org/
RSS
Google
Download
API Docs
Tips and Tutorials
Blog
Discuss
Contribute
prototype
JavaScript framework
Prototype is a JavaScript Framework that aims to ease development of dynamic web applications.
cells: function(row) {
if(row == undefined) return this.tab
return $(row).getElementsBySelector(
},
Prototype is a JavaScript Framework that aims to ease development of dynamic web applications.
Featuring a unique, easy-to-use toolkit for class-driven development and the nicest Ajax library around, Prototype is quickly becoming the codebase of choice for web application developers everywhere.
"Prototype and script.aculo.us:" 2nd beta is out!
Download
Get the latest version—1.5.1.1
Learn
Online documentation and resources.
Discuss
Mailing list and IRC
Contribute
Submit patches and report bugs.

Improves...

- built-in objects
- data types
- Ajax
- DOM



the code heard round the world... wide web

```
Object.extend = function(destination, source) {
  for (var property in source) {
    destination[property] = source[property];
  }
  return destination;
}
```

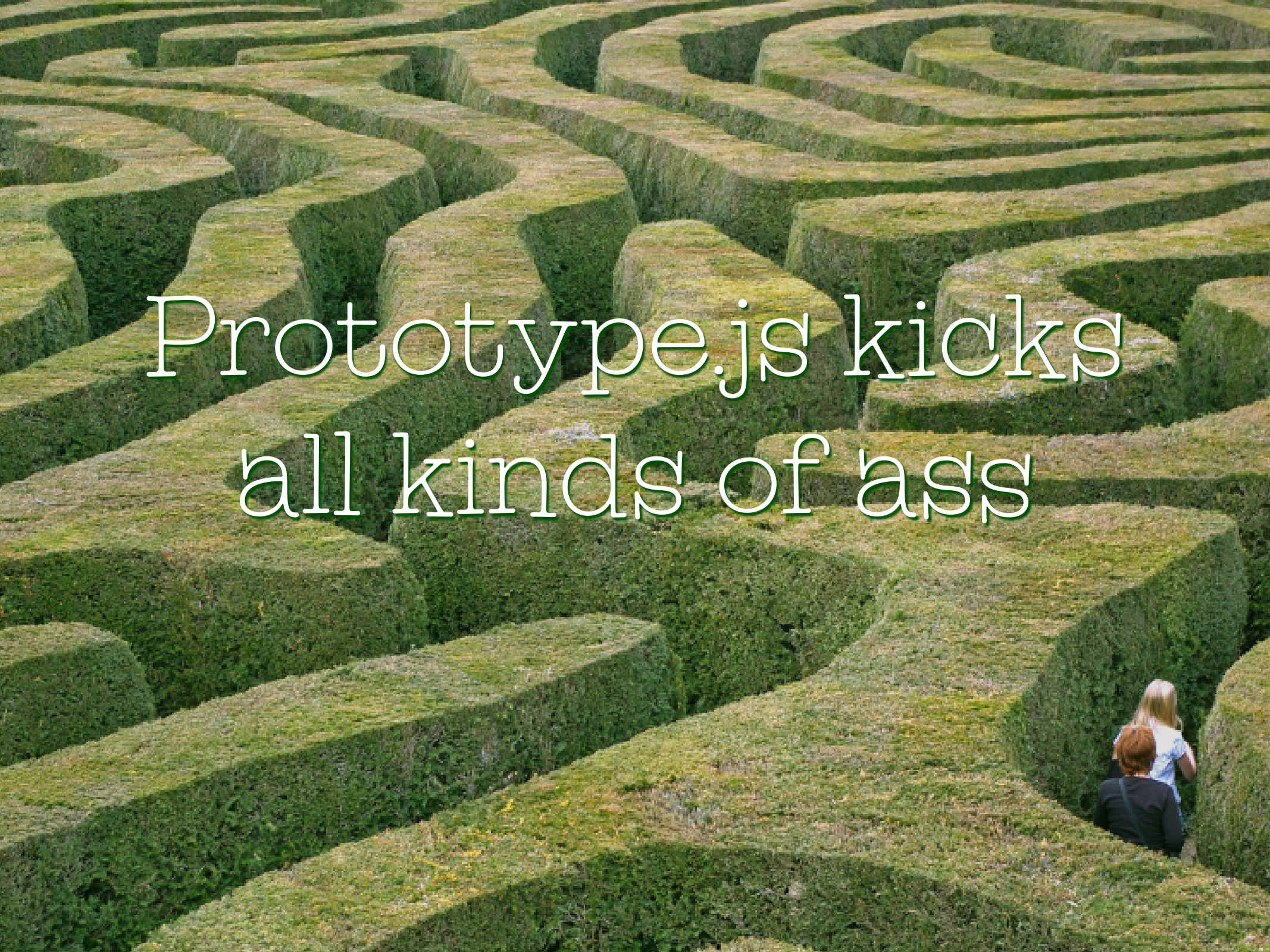
por ejemplo....

```
Object.extend(Number.prototype, {
  toColorPart: function() {
    var digits = this.toString(16);
    if (this < 16) return '0' + digits;
    return digits;
  },

  succ: function() {
    return this + 1;
  },

  times: function(iterator) {
    $R(0, this, true).each(iterator);
    return this;
  }
});
```

por ejemplo....

```
Object.extend(Number.prototype, {
  toColorPart: function() {
    var digits = this.toString(16);
    if (this < 16) return '0' + digits;
    return digits;
  },

  succ: function() {
    return this + 1;
  },

  times: function(iterator) {
    $R(0, this, true).each(iterator);
    return this;
  }
});
```

```
5.times ( alert("Hi!") )
// --> [alert] Hi!, [alert] Hi!, etc.
6.succ()
// --> 7
```


Prototype.js kicks
all kinds of ass

Prototype's classy

```
var Ninja = Class.create();
Ninja.prototype = {
  initialize: function(abilities) {
    this.abilities = [
     'Kick you in the face',
     'Rip out your spleen'
    ];

    this.abilities.each(function(ability) {
      this.executeAbility(ability)
    });
  },

  executeAbility: function(ability) {
    alert(ability);
  }
}
```

Prototype's classy

```
var Ninja = Class.create();
Ninja.prototype = {
  initialize: function(abilities) {
    this.abilities = [
     'Kick you in the face',
     'Rip out your spleen'
    ];

    this.abilities.each(function(ability) {
      this.executeAbility(ability)
    });
  },

  executeAbility: function(ability) {
    alert(ability);
  }
}
```

Prototype's classy

```
var Ninja = new Class({
  initialize: function(abilities) {
    this.abilities = [
     'Kick you in the face',
     'Rip out your spleen'
    ];

    this.abilities.each(function(ability) {
      this.executeAbility(ability)
    });
  },

  executeAbility: function(ability) {
    alert(ability);
  }
});
```

fun with Strings

```
"<b>I'm a tag!</b>".stripTags()
// I'm a tag!

"<b>I'm a tag!</b>".escapeHTML()
// &lt;b&gt;I'm a tag!&lt;/b&gt;

[5,6,7,'peach',9].without('peach',7)
// [5,6,9]
```

fun with Enumerables

```
['one','two','three'].each (
    function(num,index) {
        msg = num+" in position "+index
    }
)
```

convenience methods

```
<div class="box" id="unique_dom_id">Stuff</div>
<div class="box" id="dom_id_1">More stuff</div>
<div class="box" id="dom_id_2">YA stuff</div>
```

```
var element_single = $('unique_dom_id')
var elements_collection = $(['dom_id_1', 'dom_id_2'])
var elements_collection_css = $$('box')
```

convenience methods

```
<div class="box" id="unique_dom_id">Stuff</div>
<div class="box" id="dom_id_1">More stuff</div>
<div class="box" id="dom_id_2">YA stuff</div>
```

```
var element_single = $('unique_dom_id')
var elements_collection = $(['dom_id_1', 'dom_id_2'])
var elements_collection_css = $$('box')
```

convenience methods

```
<div class="box" id="unique_dom_id">Stuff</div>
<div class="box" id="dom_id_1">More stuff</div>
<div class="box" id="dom_id_2">YA stuff</div>
```

```
var element_single = $('unique_dom_id')
var elements_collection = $(['dom_id_1', 'dom_id_2'])
var elements_collection_css = $$('box')
```

convenience methods

```
<div class="box" id="unique_dom_id">Stuff</div>
<div class="box" id="dom_id_1">More stuff</div>
<div class="box" id="dom_id_2">YA stuff</div>
```

```
var element_single = $('unique_dom_id')
var elements_collection = $(['dom_id_1', 'dom_id_2'])
var elements_collection_css = $$('box')
```

fun with forms

```
Form.focusFirstElement("formtastic");
$("formtastic").focusFirstElement();

$('text1').activate();
Form.Element.activate('text1');

$('text1').focus();
$('select1').focus();
Form.Element.activate('text1');
```

simplest ajax possible

```
new Ajax.Request("page.php");
```

the options hash

```
new Ajax.Updater('results', ‘backend.php’, {
    method:'get',
    evalScripts:true
  })
```

```
<div id="results"> <!-- insert here --></div>
```

insertion & options

```
new Ajax.Updater('results', ‘backend.php’, {
    method:'get',
    evalScripts:true
  })
```

```
<div id="results"> <!-- insert here --></div>
```

insertion & options

```
new Ajax.Request('results', '/some_url', {
    method:'get',
    onSuccess: function(transport){
      var response = transport.responseText || "no response text"
      alert("Success! \n\n" + response)
    },
    onFailure: function(){ alert('Something went wrong...') }
  })
```

```
<div id="results"> <!-- insert here --></div>
```

callbacks

- onUnitialized
- onCreate
- onLoading
- onLoaded
- onInteractive
- onComplete
- onException
- onFailure
- onSuccess
- on200
- on404

www.prototypejs.org

questions?

www.slash7.com

amy@slash7.com