

# JAVASCRIPT DE QUALIDADE

# HOJE, AMANHÃ E SEMPRE

GUILHERME CARREIRO

THIAGO OLIVEIRA

# dextra



**GUILHERME CARREIRO**



**THIAGO OLIVEIRA**



*Há muito tempo...*

*ECMAScript*

BASICS

*A linguagem (hoje)*

# prototype

```
a = ["Javascript", "Ruby", "Java", "Python", "Haskell"];
```

```
a.first();
```

```
// => TypeError: Object Javascript,Ruby,... has no method 'first'
```

```
Array.prototype.first = function() {  
  return this[0];  
}
```

```
a.first();
```

```
// => "Javascript"
```



# var

```
var js = 'JS';  
function teste() {  
  var ruby = 'Ruby';  
  console.log(ruby);  
  console.log(js);  
  var js = 'Javascript';  
}
```

```
teste();  
// => "Ruby"  
// => undefined
```

# var

```
var js = 'JS';  
function teste() {  
  var js, ruby = 'Ruby';  
  console.log(ruby);  
  console.log(js);  
  js = 'Javascript';  
}
```

```
teste();  
// => "Ruby"  
// => undefined
```



# var

```
function f() {  
  var i = 0;  
  for (; i < 10; i++) {  
    var js = 'JavaScript'  
  }  
  console.log(js);  
}  
f();  
// => JavaScript
```

# var

```
function f() {  
  var i = 0;  
  for (; i < 10; i++) {  
    var js = 'JavaScript'  
  }  
  console.log(js);  
}  
f();  
// => JavaScript
```

# let

```
function f() {  
  var i = 0;  
  for (; i < 10; i++) {  
    let js = 'JavaScript';  
  }  
  console.log(js);  
}  
f();  
// 'js' is not defined
```

# var

```
function f() {  
  var i = 0;  
  for (; i < 10; i++) {  
    var js = 'JavaScript'  
  }  
  console.log(js);  
}  
f();  
// => JavaScript
```

# let

```
function f() {  
  var i = 0;  
  for (; i < 10; i++) {  
    let js = 'JavaScript';  
  }  
  console.log(js);  
}  
f();  
// 'js' is not defined
```

# const

```
const js = 'JavaScript';  
  
js = 'Ruby';  
// const 'js' has already been  
// declared.
```



*Bad smells (front-end)*



# *Código Javascript misturado com código HTML*

```
<!DOCTYPE html>
<html>
<head></head>
<body>
  <input type="button" onclick="validateAndSubmit();" />
  <script type="text/javascript">
    doSomething();
  </script>
</body>
</html>
```

# *Código Javascript misturado com código HTML*

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <input type="button" id="btn" />
  <script src="tdc.js" type="text/javascript"></script>
</body>
</html>
```

```
// tdc.js
var btn = document.getElementById('btn');
btn.addEventListener('click', validateAndSubmit);

(function(){
  doSomething();
})();
```

## *Lógica de negócio no Javascript*

```
var botao = document.getElementById('botao'),  
    saldo = <%= @saldo %>;
```

```
botao.onclick = function(e) {  
    if(saldo > 0) {  
        comprar();  
    } else {  
        return false;  
    }  
}
```

## *Código HTML no Javascript*

```
var botao = document.getElementById('botao'),
    saldo = <%= @saldo %>;

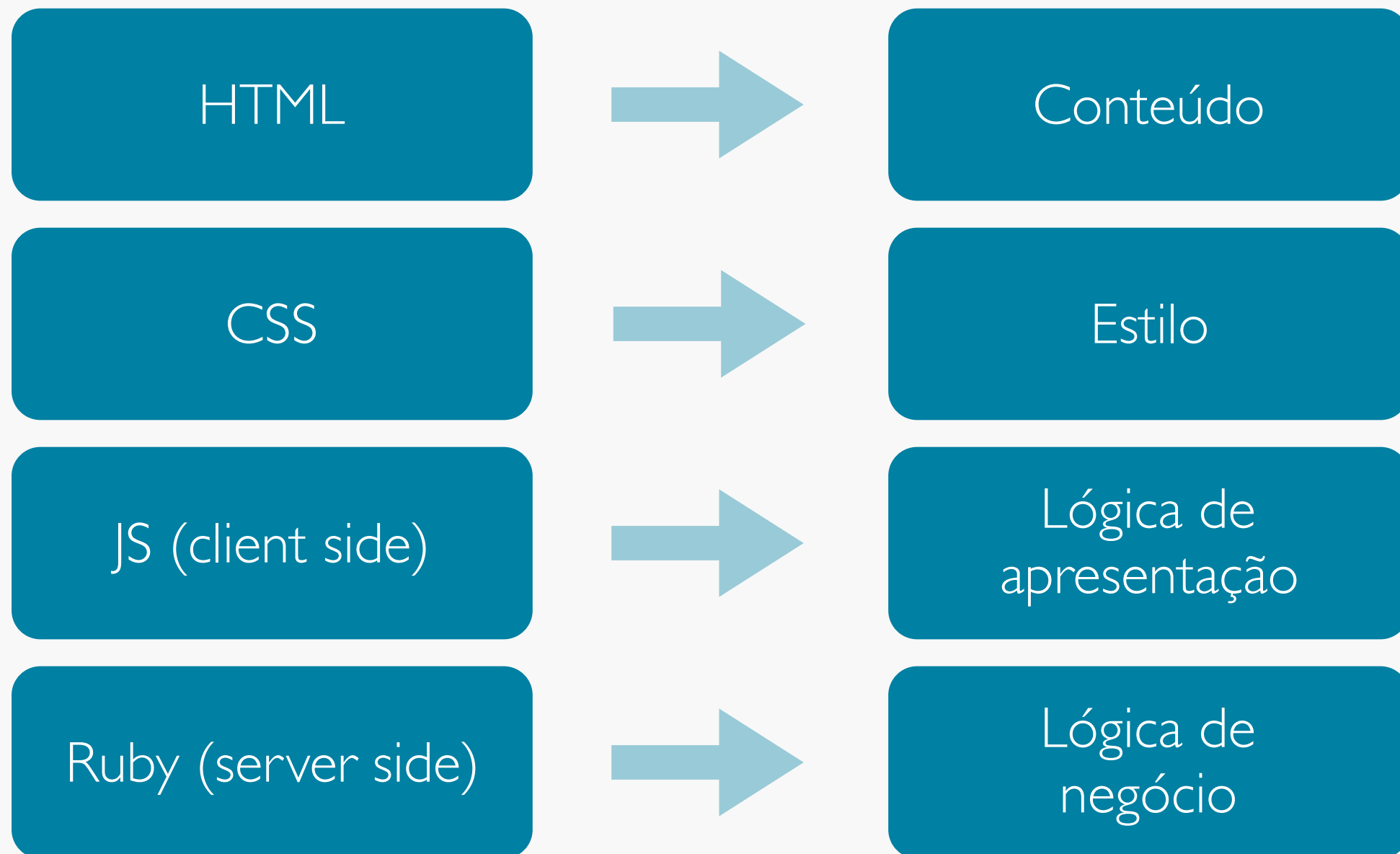
botao.onclick = function(e) {
    var status = document.getElementById('status'),
        html = '<div>',
        foto = getUserPicture();
    if(saldo > 0) {
        html += '';
        html += '<h1>Saldo: ' + saldo + ' =></h1>';
    }
    html += '</div>';
    status.innerHTML = html;
}
```

```
<!-- index.html -->
<script src="jquery.tmpl.js" type="text/javascript"></script>
<!-- ... -->
<div id="template">
  <div>
    
    <h1>Saldo: ${saldo} =></h1>
  </div>
</div>
```

```
// tdc.js
```

```
var botao = $('#botao'),
    template = $('#template'),
    saldo = <%= @saldo %>;
botao.click(function(e) {
  var html, status = $('#status'), foto = getUserPicture();
  if (saldo > 0) {
    html = $.tmpl(template.html(), {saldo: saldo, path: foto}).html();
  }
  status.html(html);
});
```

# *Separar responsabilidades*



# Code *Smells* (JavaScript)

# Code Smells (JavaScript)

```
var createUser = function (firstName, lastName, birthday, address, username, gender) {
    var age = (new Date().getTime() - birthday.getTime()) / 31556926000;
    var fullName = firstName + lastName;
    var type, g = gender == 'masculino' ? 'male' : 'female';

    if (age > 60) {
        type = 'old';
    } else if (age > 30) {
        type = 'adult';
    } else if (age > 16) {
        type = 'young';
    } else {
        type = 'kid';
    }
    return { name: fullName, age: age, address: address, gender: g, type: type };
};

var createUserRequest = function(firstName, lastName, birthday, address, username, gender) {
    $('.confirmation-modal').show();
    $('.confirmation-modal').onConfirm(function() {
        $.ajax({
            type: 'POST',
            url: '/api/users',
            data: createUser(firstName, lastName, birthday, address, username, gender)
        }).done(function() {
            $('.confirmation-modal').hide();
            $('.success-modal').show();
        });
    });
};
}
```



# Duplicated Code

```
var createUser = function (firstName, lastName, birthday, address, username, gender) {
    var age = (new Date().getTime() - birthday.getTime()) / 31556926000;
    var fullName = firstName + lastName;
    var type, g = gender == 'masculino' ? 'male' : 'female';

    if (age > 60) {
        type = 'old';
    } else if (age > 30) {
        type = 'adult';
    } else if (age > 16) {
        type = 'young';
    } else {
        type = 'kid';
    }
    return { name: fullName, age: age, address: address, gender: g, type: type };
};

var createUserRequest = function(firstName, lastName, birthday, address, username, gender) {
    $('#confirmation-modal').show();
    $('#confirmation-modal').onConfirm(function() {
        $.ajax({
            type: 'POST',
            url: '/api/users',
            data: createUser(firstName, lastName, birthday, address, username, gender)
        }).done(function() {
            $('#confirmation-modal').hide();
            $('#success-modal').show();
        });
    });
};
}
```

# Long Method

```
var createUser = function (firstName, lastName, birthday, address, username, gender) {  
    var age = (new Date().getTime() - birthday.getTime()) / 31556926000;  
    var fullName = firstName + lastName;  
    var type, g = gender == 'masculino' ? 'male' : 'female';  
  
    if (age > 60) {  
        type = 'old';  
    } else if (age > 30) {  
        type = 'adult';  
    } else if (age > 16) {  
        type = 'young';  
    } else {  
        type = 'kid';  
    }  
    return { name: fullName, age: age, address: address, gender: g, type: type };  
};
```

```
var createUserRequest = function(firstName, lastName, birthday, address, username, gender) {  
    $('.confirmation-modal').show();  
    $('.confirmation-modal').onConfirm(function() {  
        $.ajax({  
            type: 'POST',  
            url: '/api/users',  
            data: createUser(firstName, lastName, birthday, address, username, gender)  
        }).done(function() {  
            $('.confirmation-modal').hide();  
            $('.success-modal').show();  
        });  
    });  
};
```

# Long Parameter List

```
var createUser = function (firstName, lastName, birthday, address, username, gender) {  
    var age = (new Date().getTime() - birthday.getTime()) / 31556926000;  
    var fullName = firstName + lastName;  
    var type, g = gender == 'masculino' ? 'male' : 'female';  
  
    if (age > 60) {  
        type = 'old';  
    } else if (age > 30) {  
        type = 'adult';  
    } else if (age > 16) {  
        type = 'young';  
    } else {  
        type = 'kid';  
    }  
    return { name: fullName, age: age, address: address, gender: g, type: type };  
};
```

```
var createUserRequest = function(firstName, lastName, birthday, address, username, gender) {  
    $('.confirmation-modal').show();  
    $('.confirmation-modal').onConfirm(function() {  
        $.ajax({  
            type: 'POST',  
            url: '/api/users',  
            data: createUser(firstName, lastName, birthday, address, username, gender)  
        }).done(function() {  
            $('.confirmation-modal').hide();  
            $('.success-modal').show();  
        });  
    });  
};
```



# *Design Patterns*

“Each pattern **describes a problem** which occurs over and over again in our environment, and then **describes the core** of the solution to that problem, in such a way that you can use this solution a million times over, **without ever doing it the same way twice**”

- Christopher Alexander -

*Factory*

```
MyLib.modal({  
  width: 400,  
  height: 300,  
  theme: 'form-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```

```
MyLib.modal({  
  width: 100,  
  height: 70,  
  theme: 'alert-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```

```
MyLib.modal({  
  width: 400,  
  height: 300,  
  theme: 'form-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```



```
var _ = require('underscore');

var Modal = function (options) {
  var default = {
    buttons: true,
    overlay: true,
    onClose: function () {}
  };
  return MyLib.modal(_.extend(options, default));
};

var ModalFactory = function (type) {
  if (typeof this[type] !== 'function') {
    throw 'NotImplementedError';
  }
  return this[type]();
};

ModalFactory.prototype.alert = function () {
  return new Modal({
    width: 100,
    height: 70,
    theme: 'alert-modal'
  });
};

ModalFactory.prototype.form = function () {
  return new Modal({
    width: 400,
    height: 300,
    theme: 'form-modal'
  });
};
```

```
new ModalFactory('form');
new ModalFactory('alert');
new ModalFactory('form');
```

```
var _ = require('underscore');

var Modal = function (options) {
  var default = {
    buttons: true,
    overlay: true,
    onClose: function () {}
  };
  return MyLib.modal(_.extend(options, default));
};
```

```
var ModalFactory = function (type) {
  if (typeof this[type] !== 'function') {
    throw 'NotImplementedError';
  }
  return this[type]();
};
```

```
ModalFactory.prototype.alert = function () {
  return new Modal({
    width: 100,
    height: 70,
    theme: 'error-modal'
  });
};
```

```
ModalFactory.prototype.form = function () {
  return new Modal({
    width: 400,
    height: 300,
    theme: 'form-modal'
  });
};
```

```
new ModalFactory('form');
new ModalFactory('alert');
new ModalFactory('form');
```

```
var _ = require('underscore');

var Modal = function (options) {
  var default = {
    buttons: true,
    overlay: true,
    onClose: function () {}
  };
  return MyLib.modal(_.extend(options, default));
};
```

```
var ModalFactory = function (type) {
  if (typeof this[type] !== 'function') {
    throw 'NotImplementedError';
  }
  return this[type]();
};

ModalFactory.prototype.alert = function () {
  return new Modal({
    width: 100,
    height: 70,
    theme: 'error-modal'
  });
};

ModalFactory.prototype.form = function () {
  return new Modal({
    width: 400,
    height: 300,
    theme: 'form-modal'
  });
};
```

```
new ModalFactory('form');
new ModalFactory('alert');
new ModalFactory('form');
```

```
var _ = require('underscore');

var Modal = function (options) {
  var default = {
    buttons: true,
    overlay: true,
    onClose: function () {}
  };
  return MyLib.modal(_.extend(options, default));
};

var ModalFactory = function (type) {
  if (typeof this[type] !== 'function') {
    throw 'NotImplementedError';
  }
  return this[type]();
};

ModalFactory.prototype.alert = function () {
  return new Modal({
    width: 100,
    height: 70,
    theme: 'error-modal'
  });
};

ModalFactory.prototype.form = function () {
  return new Modal({
    width: 400,
    height: 300,
    theme: 'form-modal'
  });
};
```

```
new ModalFactory('form');
new ModalFactory('alert');
new ModalFactory('form');
```

```
MyLib.modal({  
  width: 400,  
  height: 300,  
  theme: 'form-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```



```
new ModalFactory('form');
```

```
MyLib.modal({  
  width: 100,  
  height: 70,  
  theme: 'alert-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```



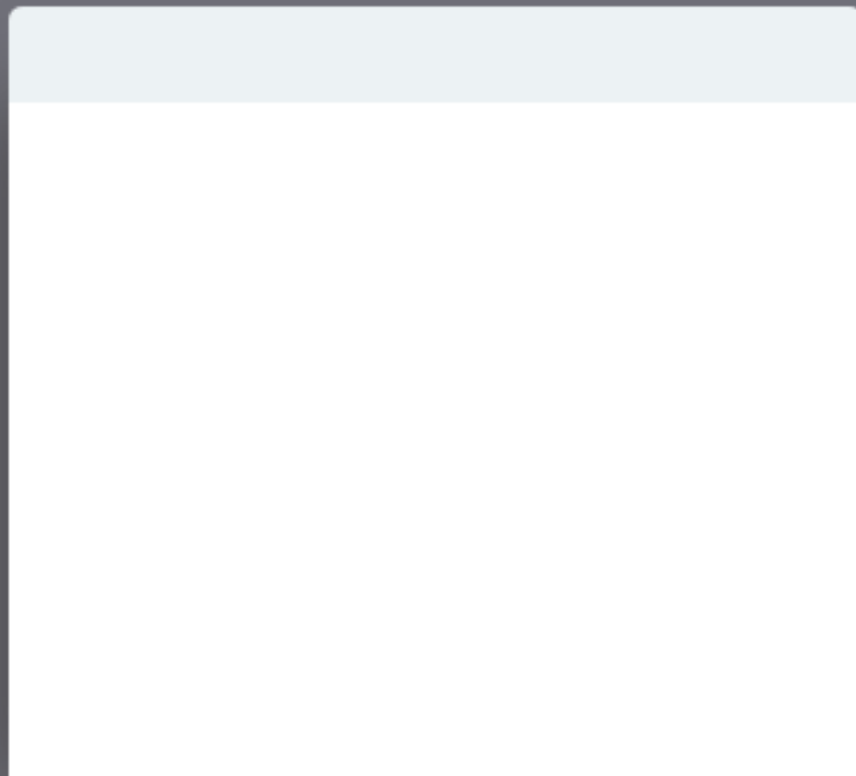
```
new ModalFactory('alert');
```

```
MyLib.modal({  
  width: 400,  
  height: 300,  
  theme: 'form-modal',  
  buttons: true,  
  overlay: true,  
  onClose: function () {}  
});
```



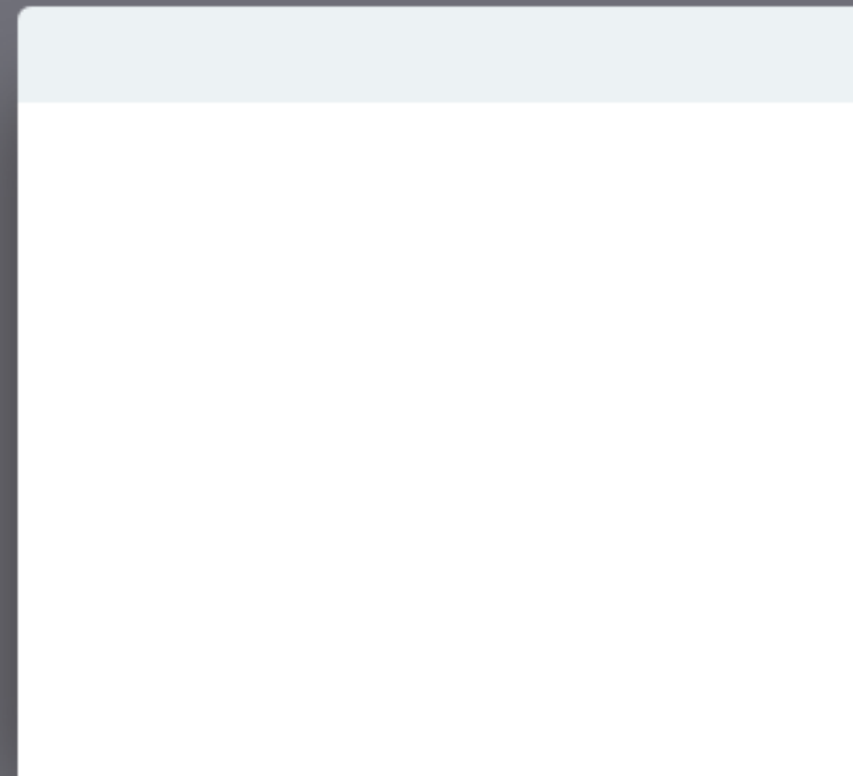
```
new ModalFactory('form');
```

*Decorator*



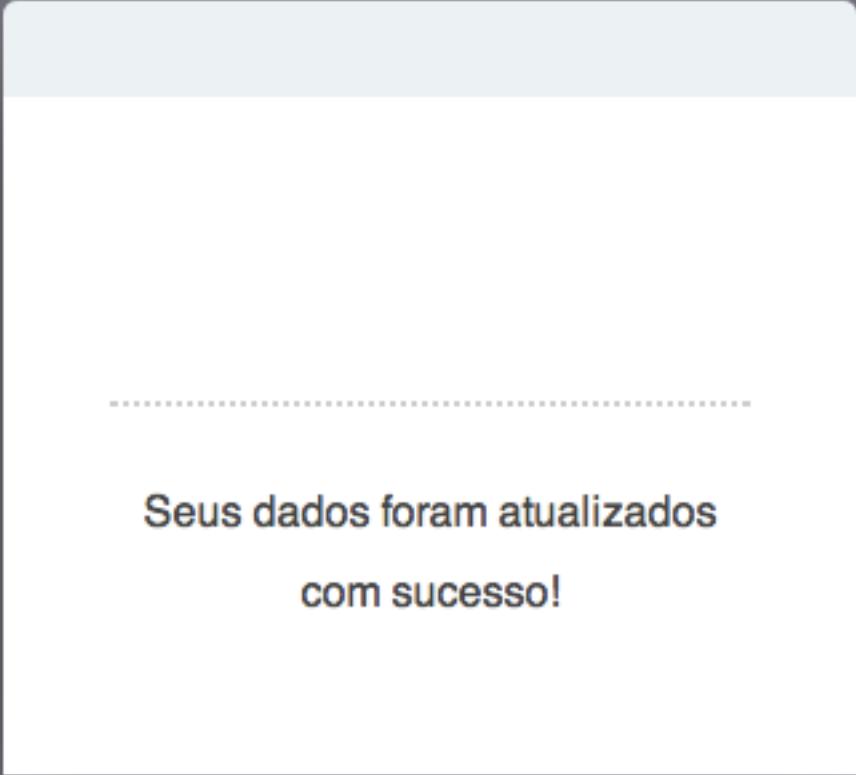
```
var modal = new Modal();
```

```
modal.show();
```




```
var modal = new Modal();
```

```
modal.show();
```



Seus dados foram atualizados  
com sucesso!



Ocorreu um erro durante a  
atualização! :O

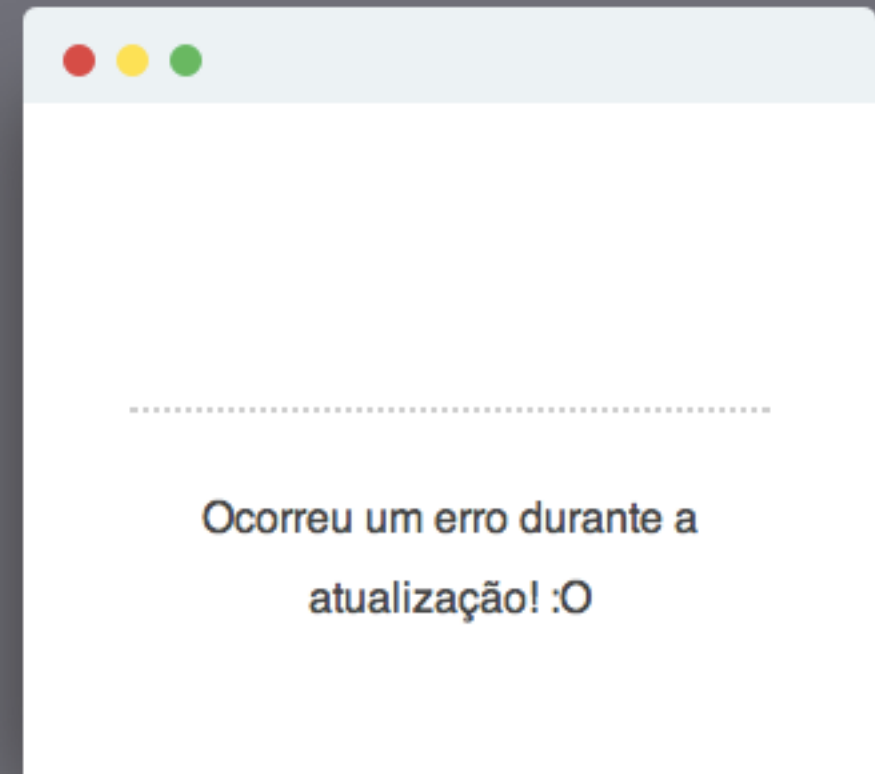
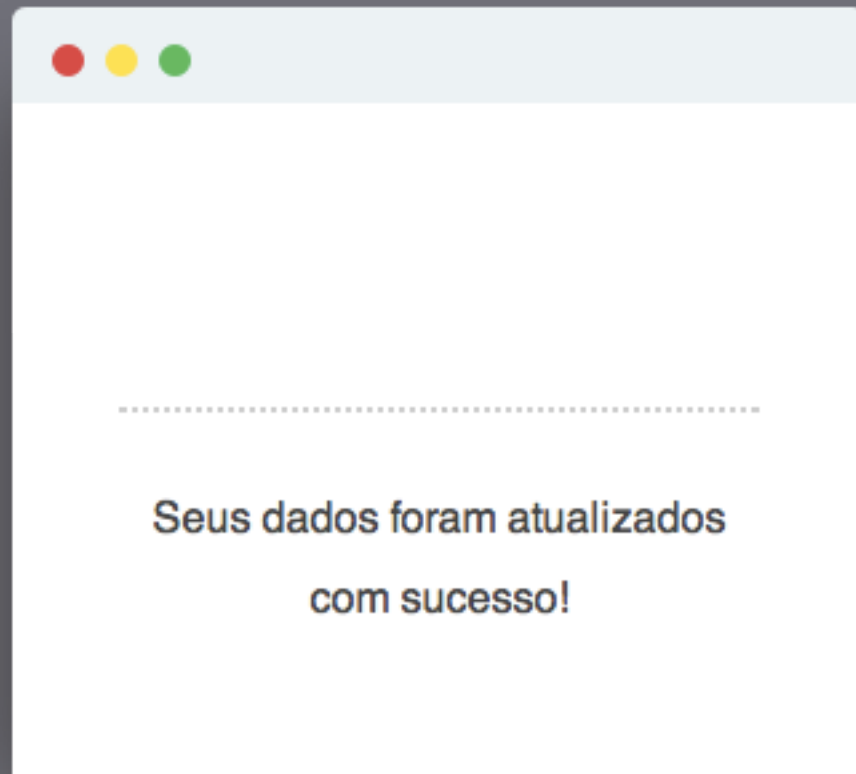
```
var modal = new Modal();  
  
modal.message = 'Seus dados foram  
atualizados com sucesso!';
```

```
modal.show();
```

```
var modal = new Modal();  
  
modal.message = 'Ocorreu um erro  
durante a atualização! :O';
```

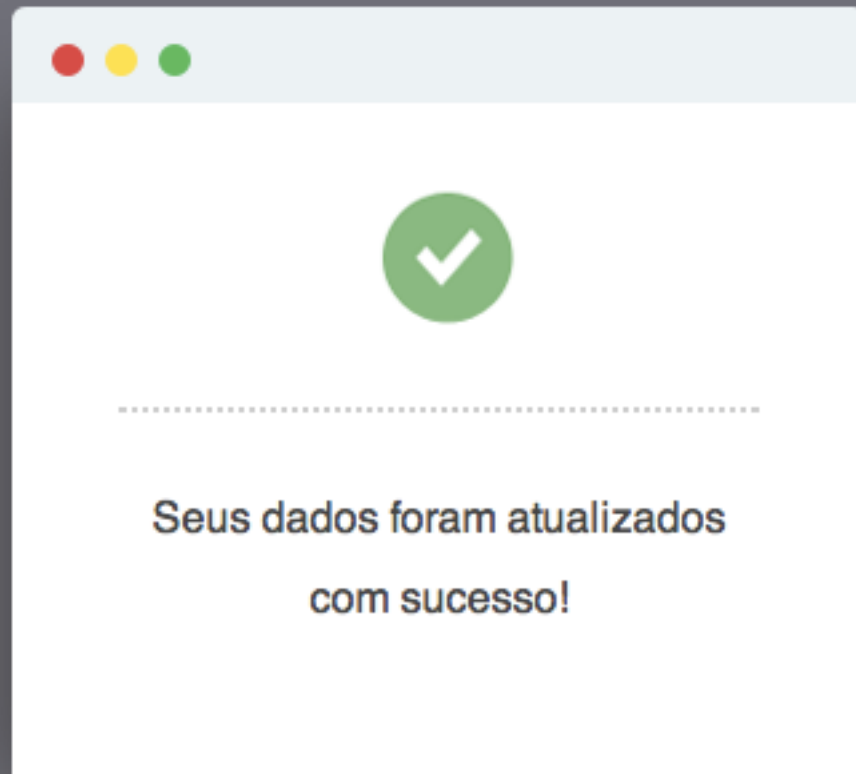
```
modal.show();
```





```
var modal = new Modal();  
  
modal.message = 'Seus dados foram  
atualizados com sucesso!';  
  
TopButtons(modal);  
  
modal.show();
```

```
var modal = new Modal();  
  
modal.message = 'Ocorreu um erro  
durante a atualização! :O';  
  
TopButtons(modal);  
  
modal.show();
```



```
var modal = new Modal();

modal.message = 'Seus dados foram
    atualizados com sucesso!';

TopButtons(modal);
SuccessModal(modal);

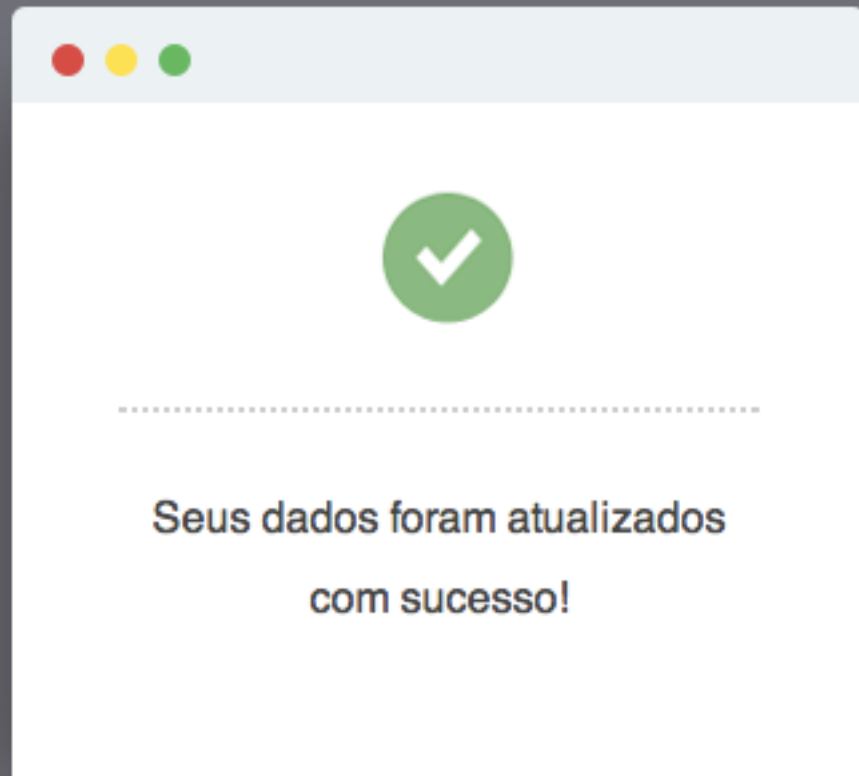
modal.show();
```

```
var modal = new Modal();

modal.message = 'Ocorreu um erro
    durante a atualização! :O';

TopButtons(modal);
ErrorModal(modal);

modal.show();
```



```
var TopButtons = function (modal) {  
    modal.showTopButtons = true;  
  
    modal.close = function () {  
        // ...  
    };  
    modal.maximize = function () {  
        // ...  
    };  
    modal.minimize = function () {  
        // ...  
    };  
};
```

```
var modal = new Modal();  
  
modal.message = 'Seus dados foram  
    atualizados com sucesso!';  
  
TopButtons(modal);  
SuccessModal(modal);  
  
modal.show();
```

```
var modal = new Modal();  
  
modal.message = 'Ocorreu um erro  
    durante a atualização! :0';  
  
TopButtons(modal);  
ErrorModal(modal);  
  
modal.show();
```

*Observer*

```
var Context = function() {
    this.modals = [];

    this.updateAll = function(msg){
        this.notify(msg);
    };

    this.notify = function(msg) {
        var i = 0;
        for (; i < this.modals.length; i++) {
            this.modals[i].update(msg);
        }
    };
};

var Modal = function() {

    this.update = function(msg) {
        if (msg == 'close') {
            this.close();
        }
    };

    this.close = function(){
        console.log('Popup fechada!');
    };
};
```

```
var context = new Context();

var modal1 = new Modal();
var modal2 = new Modal();
var modal3 = new Modal();

context.modals.push(modal1);
context.modals.push(modal2);
context.modals.push(modal3);

context.updateAll('close');
// Popup fechada!
// Popup fechada!
// Popup fechada!
```

*Strategy*

```

var Validator = function (options) {
    var field;
    var fields = options.fields;
    var validations = options.validations;

    this.errors = [];

    this.hasErrors = function() {
        return this.errors.length !== 0;
    };

    this.validate = function (fieldName) {
        var type = validations[fieldName];
        var method = this.types[type];
        var value = fields[fieldName];

        if (!method(value)) {
            this.errors.push('Invalid value for '
                + fieldName);
        };
    };

    for (field in fields) {
        this.validate(field);
    }
};

Validator.prototype.types = {
    isEmpty: function(value) {
        return value !== "";
    }
};

```

```

var validator = new Validator({
    fields: {
        firstName: 'Thiago'
    },
    validations: {
        firstName: 'isEmpty'
    }
});

console.log(validator.hasErrors());
// => false

console.log(validator.errors);
// => []

```

```

var Validator = function (options) {
    var field;
    var fields = options.fields;
    var validations = options.validations;

    this.errors = [];

    this.hasErrors = function() {
        return this.errors.length !== 0;
    };

    this.validate = function (fieldName) {
        var type = validations[fieldName];
        var method = this.types[type];
        var value = fields[fieldName];

        if (!method(value)) {
            this.errors.push('Invalid value for '
                + fieldName);
        };
    };

    for (field in fields) {
        this.validate(field);
    }
};

Validator.prototype.types = {
    isEmpty: function(value) {
        return value !== "";
    }
};

```

```

var validator = new Validator({
    fields: {
        firstName: ''
    },
    validations: {
        firstName: 'isEmpty'
    }
});

console.log(validator.hasErrors());
// => true

console.log(validator.errors);
// => ['Invalid value for firstName']

```



```

var Validator = function (options) {
    var field;
    var fields = options.fields;
    var validations = options.validations;

    this.errors = [];

    this.hasErrors = function() {
        return this.errors.length !== 0;
    };

    this.validate = function (fieldName) {
        var type = validations[fieldName];
        var method = this.types[type];
        var value = fields[fieldName];

        if (!method(value)) {
            this.errors.push('Invalid value for '
                + fieldName);
        };
    };

    for (field in fields) {
        this.validate(field);
    }
};

```

```

Validator.prototype.types = {
    isEmpty: function(value) {
        return value !== "";
    }
};

```

```

var validator = new Validator({
    fields: {
        firstName: ''
    },
    validations: {
        firstName: 'isEmpty'
    }
});

console.log(validator.hasErrors());
// => true

console.log(validator.errors);
// => ['Invalid value for firstName']

```

```

var Validator = function (options) {
  var field;
  var fields = options.fields;
  var validations = options.validations;

  this.errors = [];

  this.hasErrors = function() {
    return this.errors.length !== 0;
  };

  this.validate = function (fieldName) {
    var type = validations[fieldName];
    var method = this.types[type];
    var value = fields[fieldName];

    if (!method(value)) {
      this.errors.push('Invalid value for '
        + fieldName);
    };
  };

  for (field in fields) {
    this.validate(field);
  }
};

```

```

Validator.prototype.types = {
  isEmpty: function(value) {
    return value !== "";
  }
};

```

```

var validator = new Validator({
  fields: {
    firstName: ''
  },
  validations: {
    firstName: 'isEmpty'
  }
});

console.log(validator.hasErrors());
// => true

console.log(validator.errors);
// => ['Invalid value for firstName']

```

```

var Validator = function (options) {
    var field;
    var fields = options.fields;
    var validations = options.validations;

    this.errors = [];

    this.hasErrors = function() {
        return this.errors.length !== 0;
    };

    this.validate = function (fieldName) {
        var type = validations[fieldName];
        var method = this.types[type];
        var value = fields[fieldName];

        if (!method(value)) {
            this.errors.push('Invalid value for '
                + fieldName);
        };
    };

    for (field in fields) {
        this.validate(field);
    }
};

Validator.prototype.types = {
    isEmpty: function(value) {
        return value !== "";
    }
};

```

```

var validator = new Validator({
    fields: {
        firstName: ''
    },
    validations: {
        firstName: 'isEmpty'
    }
});

console.log(validator.hasErrors());
// => true

console.log(validator.errors);
// => ['Invalid value for firstName']

```

HERANCA

*Herança*

```
function Parent() {  
  this.name = 'Joey';  
}
```

```
Parent.prototype.say = function() {  
  console.log('I\'m ' + this.name);  
}
```

```
function Child() {  
  this.name = 'Dee Dee';  
}
```

```
function inherits(Child, Parent) {  
  Child.prototype = Object.create(Parent.prototype);  
}
```

```
inherits(Child, Parent);
```

```
var a = new Child();
```

```
a.say(); // => I'm Dee Dee
```

*Padrão Klass*

```
var klass = require('klass');

var Person = klass(function (name) {
  this.name = name;
}).methods({
  walk: function () {
    console.log('Walking...');
  },
  say: function () {
    console.log('Hey, my name is ' + this.name);
  }
});

var Thiaguinho = Person.extend(function () {
  this.name = 'Thiaguinho';
}).methods({
  sing: function () {
    console.log('Caraca, moleque! Que dia! Que isso?');
  }
});

var person = new Person('John Doe');
person.say();
// => Hey, my name is John Doe

var thi = new Thiaguinho();
thi.sing();
// => Caraca, moleque! Que dia! Que isso?
thi.say();
// => Hey, my name is Thiaguinho
```



```
var klass = require('klass');

var Person = klass(function (name) {
  this.name = name;
}).methods({
  walk: function () {
    console.log('Walking...');
  },
  say: function () {
    console.log('Hey, my name is ' + this.name);
  }
});
```

```
var Thiaguinho = Person.extend(function () {
  this.name = 'Thiaguinho';
}).methods({
  sing: function () {
    console.log('Caraca, moleque! Que dia! Que isso?');
  }
});
```

```
var person = new Person('John Doe');
person.say();
// => Hey, my name is John Doe
```

```
var thi = new Thiaguinho();
thi.sing();
// => Caraca, moleque! Que dia! Que isso?
thi.say();
// => Hey, my name is Thiaguinho
```

```
var klass = require('klass');

var Person = klass(function (name) {
  this.name = name;
}).methods({
  walk: function () {
    console.log('Walking...');
  },
  say: function () {
    console.log('Hey, my name is ' + this.name);
  }
});

var Thiaguinho = Person.extend(function () {
  this.name = 'Thiaguinho';
}).methods({
  sing: function () {
    console.log('Caraca, moleque! Que dia! Que isso?');
  }
});

var person = new Person('John Doe');
person.say();
// => Hey, my name is John Doe

var thi = new Thiaguinho();
thi.sing();
// => Caraca, moleque! Que dia! Que isso?
thi.say();
// => Hey, my name is Thiaguinho
```



*Classes com o ECMAScript 6*

# Classes

```
class Man {  
  constructor (name) {  
    this.name = name;  
  }  
  say (message) {  
    return this.name + ': ' + message;  
  }  
}
```

```
let john = new Man('John Doe');
```

```
john.say('Hi!');  
// => John Doe: Hi!
```

```
class Man {  
  constructor (name) {  
    this.name = name;  
  }  
  say (message) {  
    return this.name + ': ' + message;  
  }  
}
```

```
class Superman extends Man {  
  constructor () {  
    super('Clark Kent');  
  }  
  fly () {  
    return 'Flying...';  
  }  
}
```

```
let superMan = new Superman();  
superMan.say('Yeah!');  
// => Clark Kent: Yeah!  
superMan.fly();  
// => Flying...
```

NOVIDADES

## *Arrow functions*

```
var plus = function (a, b) {  
  return a + b;  
};
```

```
var plus = (a, b) => {  
  return a + b;  
};
```

```
var plus = (a, b) => a + b;
```

```
var square = a => a * a;
```



## Arrow functions

```
[1, 2, 3].map(function (i) {  
  return i * i;  
});  
// => [1, 4, 9]
```

```
[1, 2, 3].map(x => x * x);  
// => [1, 4, 9]
```

# Modules

```
// plugins/math.js
export function square (a) {
  return a * a;
}
```

```
// index.js
import {square} from 'plugins/math.js';
square(1);
```

# Modules

```
// plugins/math.js
export function square (a) {
  return a * a;
}
```

```
// index.js
import 'plugins/math.js' as Math;
Math.square(1);
```

# Default arguments

```
var g = function (a, b) {  
  a = a || 1;  
  b = b || 1;  
  return a + b;  
}
```

```
var f = function (a = 1, b = 1) {  
  return a + b;  
}
```

```
f();  
// => 2
```

```
f(2, 2);  
// => 4
```

```
f(undefined, 7);  
// => 8
```

## *Rest parameters*

```
var f = function (a = 1, ...b) {  
  console.log(a, b);  
}
```

```
f(1);  
// => 1 []
```

```
f(1, 2);  
// => 1 [2]
```

```
f(1, 2, 3);  
// => 1 [2, 3]
```

# Interpolation

```
let a = 4;  
let b = 3;  
let code = `${a} + ${b} = ${a + b}`;  
// => 4 + 3 = 7
```

```
let code = `  
  def plus(a, b)  
    a + b  
  end  
`;  
;
```

*Quando?*

## ECMAScript 6 compatibility table

Also see compatibility tables for [ES5](#) or [non-standard](#) features

 Flattr

40

by kangax

Please note that *some of these tests* represent **existence**, not functionality or full conformance.

Sort by number of features? ☐

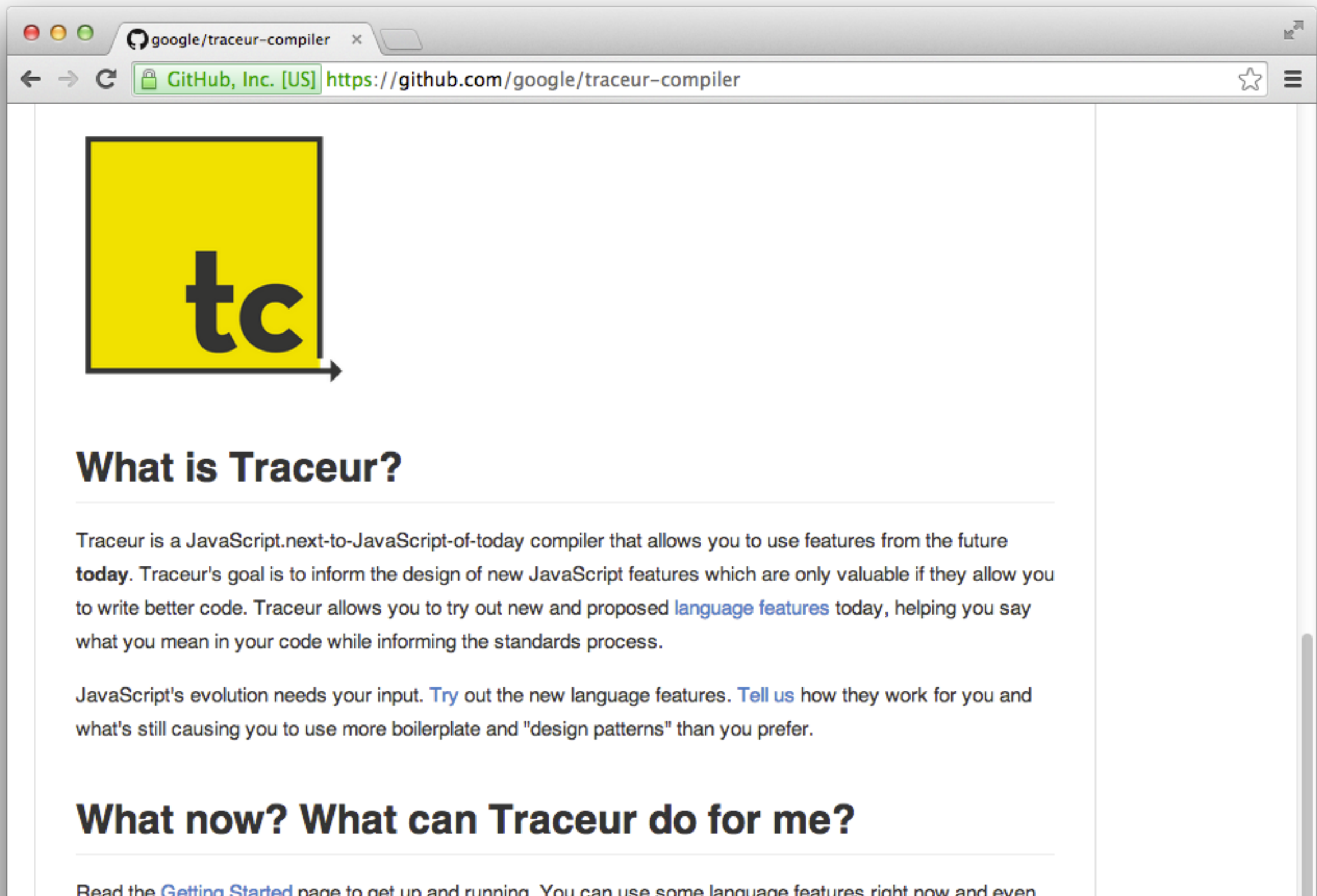
Show obsolete browsers?

[illegible]



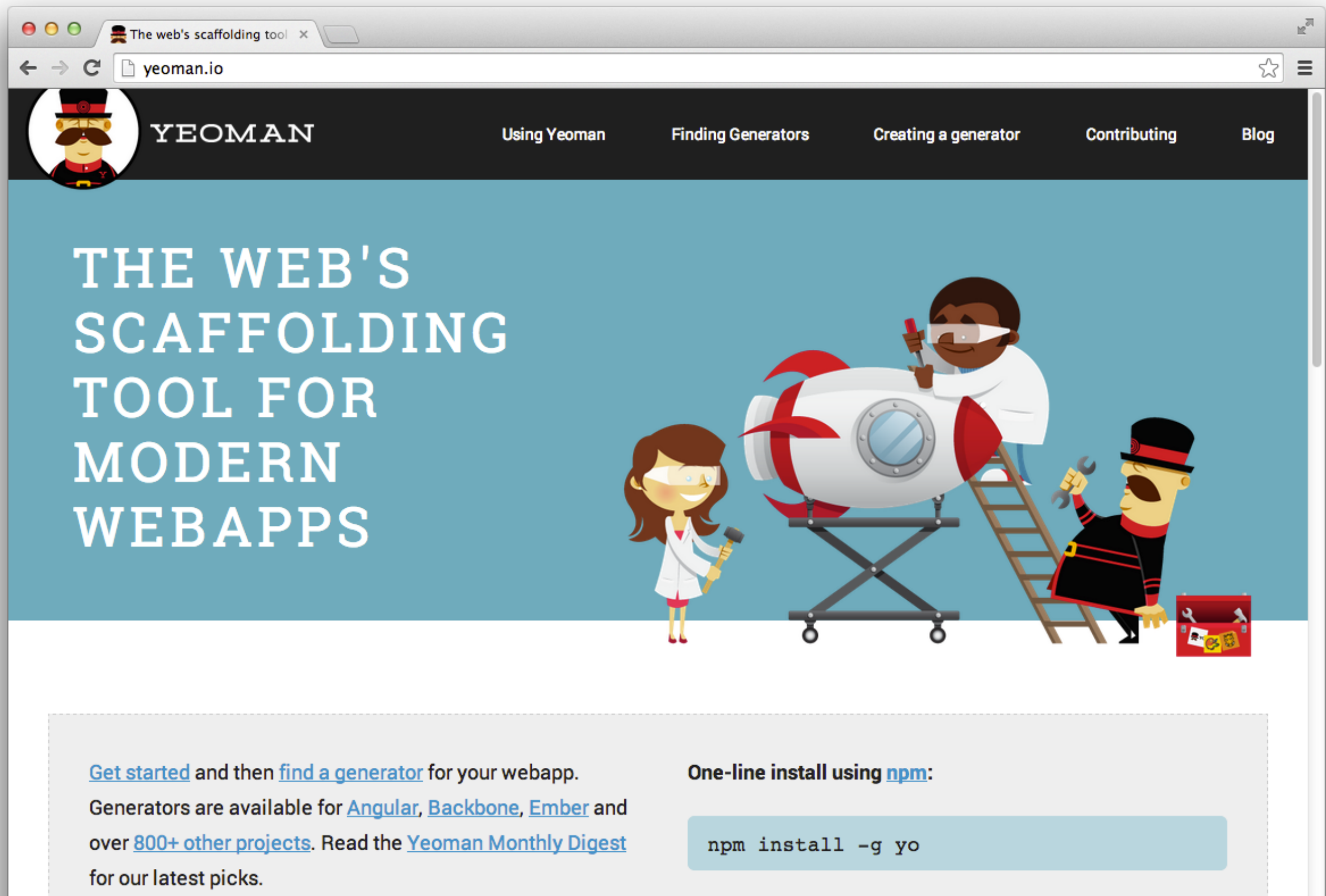
*Como começar?*

# Traceur



*Como melhorar hoje?*

# Yeoman



The screenshot shows the Yeoman website homepage in a browser window. The browser's address bar displays 'yeoman.io'. The website has a dark navigation bar with the Yeoman logo (a character with a top hat and mustache) and the word 'YEOMAN'. To the right of the logo are links: 'Using Yeoman', 'Finding Generators', 'Creating a generator', 'Contributing', and 'Blog'. The main content area has a light blue background. On the left, the text 'THE WEB'S SCAFFOLDING TOOL FOR MODERN WEBAPPS' is written in large, white, sans-serif capital letters. On the right, there is a colorful illustration of three characters working on a white rocket with red fins. One character is on a ladder, another is on the ground with a wrench, and a third is on the ground with a hammer. A red toolbox is also visible. Below this illustration, there is a light gray box containing text and a code snippet.

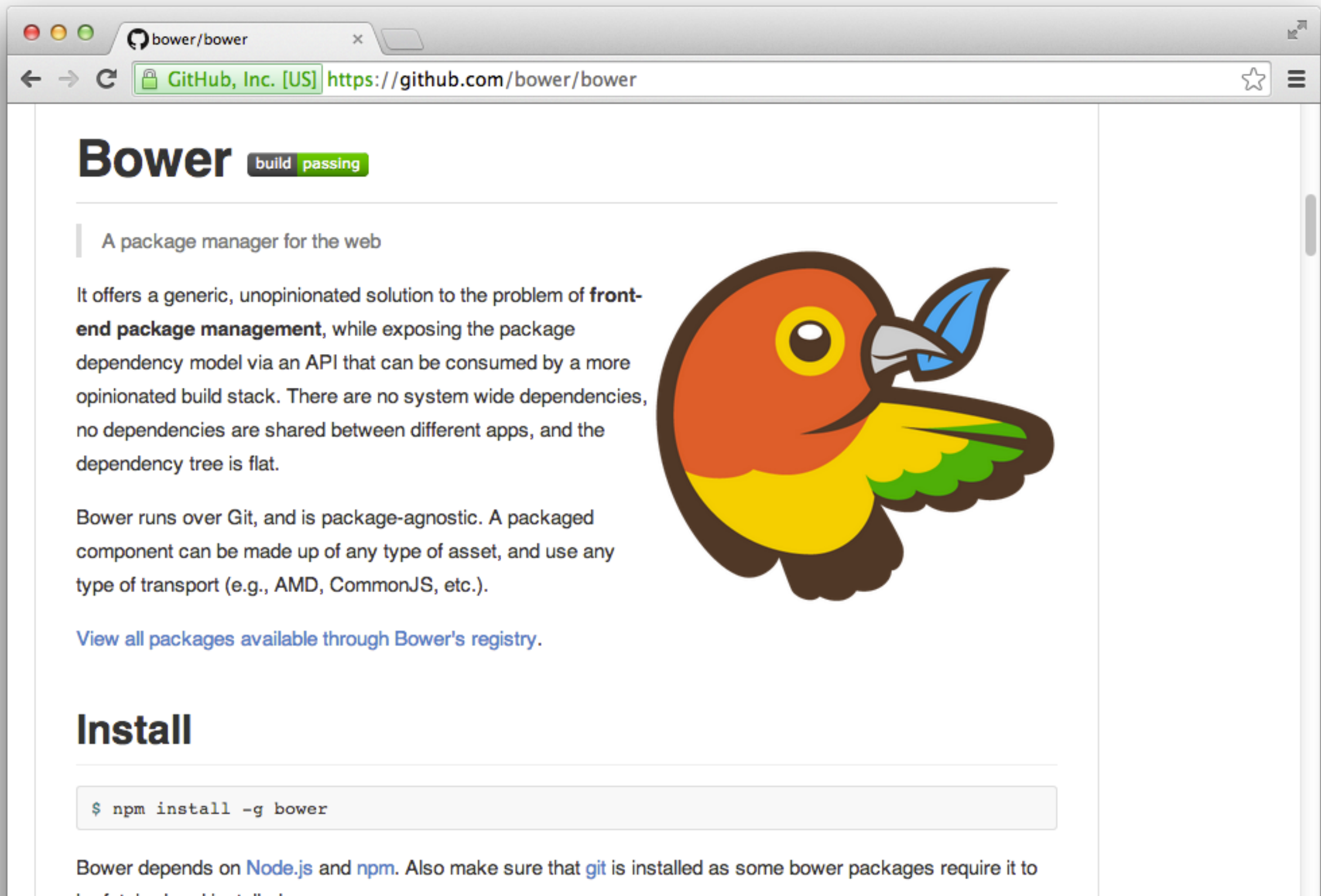
THE WEB'S  
SCAFFOLDING  
TOOL FOR  
MODERN  
WEBAPPS

[Get started](#) and then [find a generator](#) for your webapp.  
Generators are available for [Angular](#), [Backbone](#), [Ember](#) and  
over [800+ other projects](#). Read the [Yeoman Monthly Digest](#)  
for our latest picks.

One-line install using [npm](#):

```
npm install -g yo
```

# Bower



The image is a screenshot of a web browser displaying the GitHub repository page for Bower. The browser's address bar shows the URL `https://github.com/bower/bower`. The page title is "Bower" with a status badge indicating "build passing". Below the title, a subtitle reads "A package manager for the web". The main text describes Bower as a generic, unopinionated solution for front-end package management, highlighting its flat dependency tree and package-agnostic nature. To the right of the text is a cartoon illustration of a bird with an orange head, yellow body, and green wings. Below the text, there is a link to "View all packages available through Bower's registry." and a section titled "Install" which contains a code block for installing Bower via npm. At the bottom, a note states that Bower depends on Node.js and npm, and that git is also required for some packages.

**Bower** build passing

A package manager for the web

It offers a generic, unopinionated solution to the problem of **front-end package management**, while exposing the package dependency model via an API that can be consumed by a more opinionated build stack. There are no system wide dependencies, no dependencies are shared between different apps, and the dependency tree is flat.

Bower runs over Git, and is package-agnostic. A packaged component can be made up of any type of asset, and use any type of transport (e.g., AMD, CommonJS, etc.).

[View all packages available through Bower's registry.](#)

## Install

```
$ npm install -g bower
```

Bower depends on [Node.js](#) and [npm](#). Also make sure that [git](#) is installed as some bower packages require it to be fetched and installed.

# Grunt.js



The screenshot shows the Grunt.js website in a web browser. The browser's address bar displays 'gruntjs.com'. The website has an orange navigation bar with links for 'Getting Started', 'Plugins', and 'Documentation'. The main content area features a large, stylized yellow cartoon animal head (a Grunt) on the left. To its right, the word 'GRUNT' is written in large, bold, dark brown letters, followed by 'The JavaScript Task Runner' in a slightly smaller, bold, dark brown font. Below this, there are two columns of text. The left column is titled 'Why use a task runner?' and the right column is titled 'Why use Grunt?'. At the bottom left, there is a section for 'Latest Version' showing 'Stable: v0.4.5' and a small Chrome DevTools interface. Below that, there is an advertisement for 'Discover Dev Tools' by Bocoup.

Grunt: The JavaScript Task Runner

Getting Started Plugins Documentation

# GRUNT

## The JavaScript Task Runner

### Latest Version

- Stable: [v0.4.5](#)

Discover Dev Tools, a free screencast to help you master Chrome Dev Tools.

Ads by Bocoup.

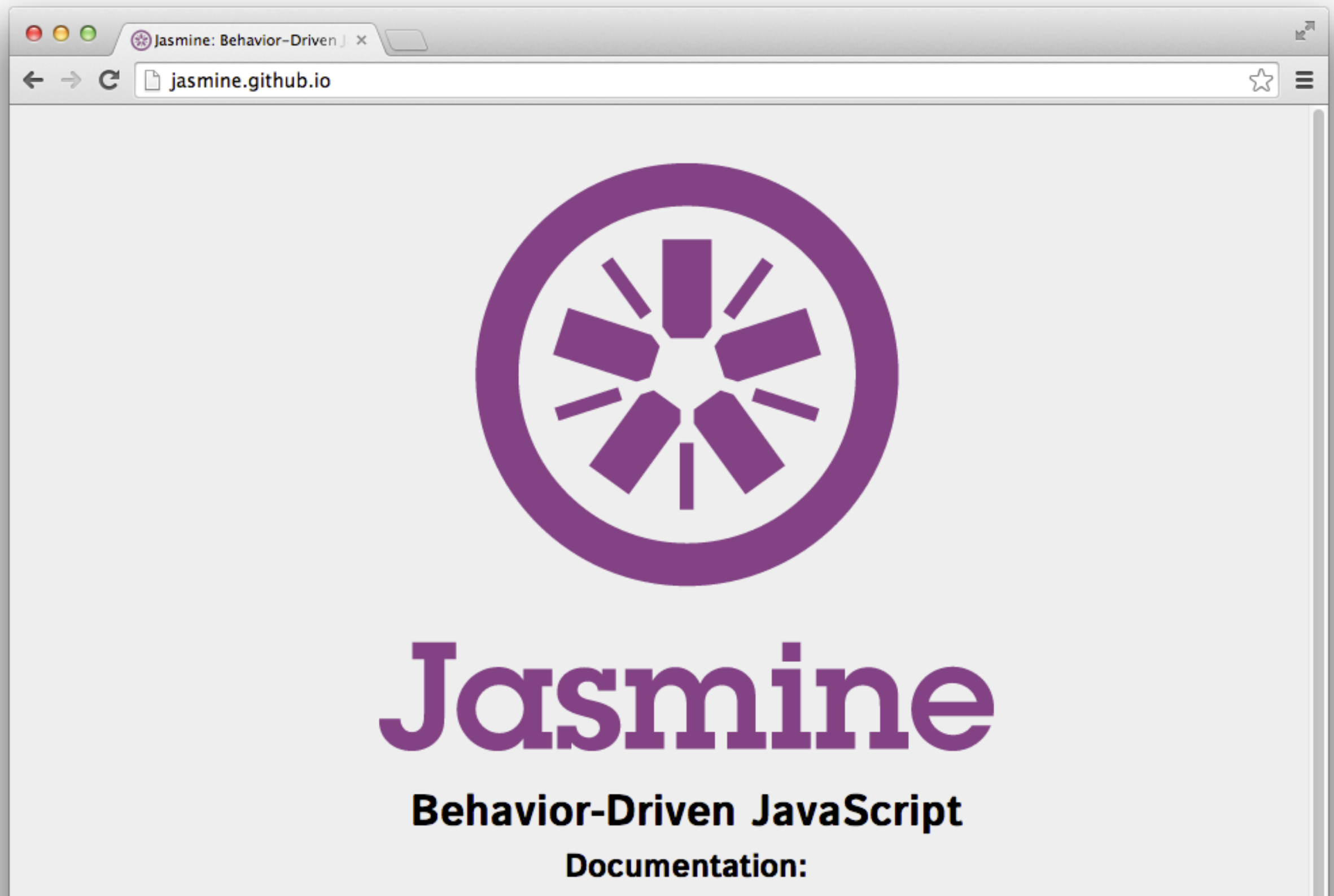
### Why use a task runner?

In one word: automation. The less work you have to do when performing repetitive tasks like minification, compilation, unit testing, linting, etc, the easier your job becomes. After you've configured it, a task runner can do most of that mundane work for you — and your team — with basically zero effort.

### Why use Grunt?

The Grunt ecosystem is huge and it's growing every day. With literally hundreds of plugins to choose from, you can use Grunt to automate just about anything with a minimum of effort. If someone hasn't already built what you need, authoring and publishing your own Grunt plugin to npm is a breeze.

# *Jasmine*



**OBRIGADO! :)**

**PERGUNTAS?**