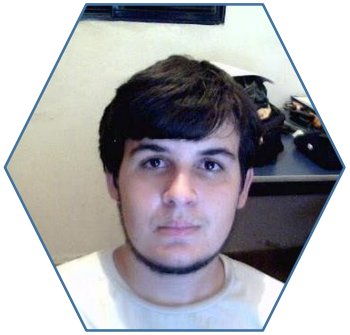
# Pós-graduação em Desenvolvimento Web e Aplicativos Móveis

**Fábio Rodrigues Jorge**
**fabinhojorgenet@gmail.com**

https://github.com/fabinhojorge/aula-javascript



# Fábio Rodrigues Jorge

novitas

www.novitasdi.com.br

## Assuntos:

- JavaScript
- jQuery
- Bibliotecas e Frameworks Atuais
- Angular

Computing and Information Systems Research Lab
COMPSI
2010
UNIVEM

USP

# Onde estamos?

| Disciplina | CH | Modalidade |
|---|---|---|
| Frameworks de Front-End com HTML e CSS<br>(HTML 5, CSS3 (SAAS e LESS), Bootstrap Gulp, Bower, Web Components) | 20 | Presencial |
| **Frameworks de Front-End Biblioteca Web JavaScript (JQuery, Meteor, React.js, Angular)** | **20** | **Presencial** |
| Conceitos de desenvolvimento de Web Apps | 10 | Presencial |
| Arquitetura da Informação e Design de Interação<br>(UX Design, Design Responsivo, Mobile First) | 10 | Presencial |

# Objetivo da Disciplina

Formar e atualizar os profissionais de TI com as principais metodologias envolvendo a linguagem de programação JavaScript e suas Bibliotecas e Frameworks mais renomados.

# Agenda

- Introdução ao Front-End
- Javascript
  - História e Características
  - DOM
  - Variáveis
  - Funções
  - Eventos
  - Projeto
- Ajax + Json

# Dúvidas? Podemos começar?

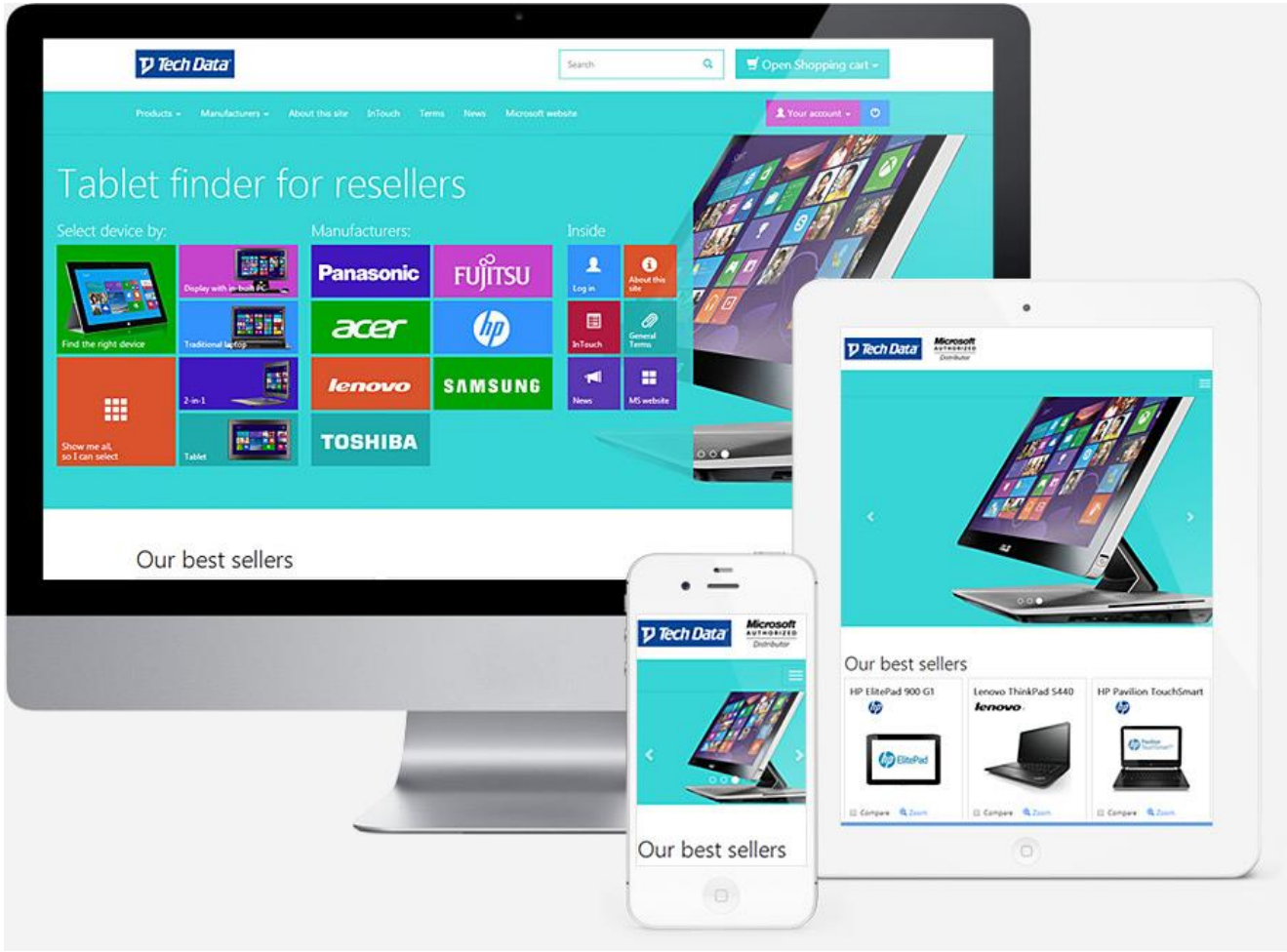
# Introdução ao Front-End ...

## ... mas agora com o JavaScript

# Introdução Front-End

*O Front-End é composto por diversas tecnologias que fazem a interação com o Usuário.*

*A **INTERFACE** que é responsável por transmitir informações e ditar o fluxo das aplicações*


Tech Data
Search
Open Shopping cart
Products
Manufacturers
About this site
InTouch
Terms
News
Microsoft website
Your account
Tablet finder for resellers
Select device by:
Find the right device
Display with in-built PC
Traditional laptop
2-in-1
Tablet
Show me all, so I can select
Manufacturers:
Panasonic
FUJITSU
acer
lenovo
SAMSUNG
TOSHIBA
Inside
Log in
About this site
InTouch
General Terms
News
MS website
Our best sellers
Tech Data
Microsoft Authorized Distributor
Our best sellers
HP ElitePad 900 G1
Lenovo ThinkPad S440
HP Pavilion TouchSmart
Tech Data
Microsoft Authorized Distributor
Our best sellers

# Qual a importância do Front-End

- **80% do tempo** de carregamento é gasto no Front-End;

- **Interfaces complexas e detalhadas** -> necessário um especialista

- **Diferentes tipos de dispositivos** com diferentes tamanhos e tipos de telas.

- O **VISUAL** é a primeira coisa a ser vista. Se ele estiver quebrado, o risco do usuário sair e nunca mais voltar é maior

# Introdução Front-End

O que é DHTML?


Dynamic HTML é a união do:

- HTML
- JavaScript
- Estilo(CSS)
- DOM (Modelo de objeto de Documento).

Permite a modificação da página HTML na própria máquina do cliente, sem a necessidade de processar no servidor.

# Agora sim o Javascript

# Hello World!

```
1
2  x = "Olá Mundo!"
3
4  alert(x)
5
```

E como e onde se executa o Javascript?

# Onde é executado?

- Nos próprios elementos (Não faça isso)

- Dentro da tag <script></script> do HTML

- Em arquivo externo carregado pela tag

<script src="arquivo.js"></script>

# História

Nome: **ECMAScript** (Ultima versão foi o ES6)

- Tinha o objetivo de validar formulários
- Criada pelo Netscape em setembro de 1995 com o nome de LiveScript
- **Javascript**, LiveScript(*), JScript e ActionScript não são nada mais que *dialetos* de ECMAScript
- Rumores: Linguagem Fraca? Bugada?

# Características

- Script: Interpretado em tempo de execução
- **Tipagem Dinâmica**
- **Fracamente Tipada**
- Case Sensitive
- **Linguagem Multi-paradigma**: Orientação a Objetos, Imperativa e Funcional
- Baseada em protótipo

# Tipagem Dinâmica e Fracamente Tipada

codigo1.js

```
1
2  var x = 1;
3  typeof x; //'number'
4
5  x = 'Hello World'
6  typeof x; //'string'
7
8  /*-----------------------*/
9
10 typeof variable === "string"  // String
11 typeof variable === "number"  // Number
12 typeof variable === "boolean" // Boolean
13 typeof variable === "object"  // Object
14 typeof variable === "function"// Function
15 Array.isArray(arrayObject)    // Array
```

# Características

“Outros” Javascripts:
TypeScript, CoffeeScript, IcedCoffeeScript, etc…

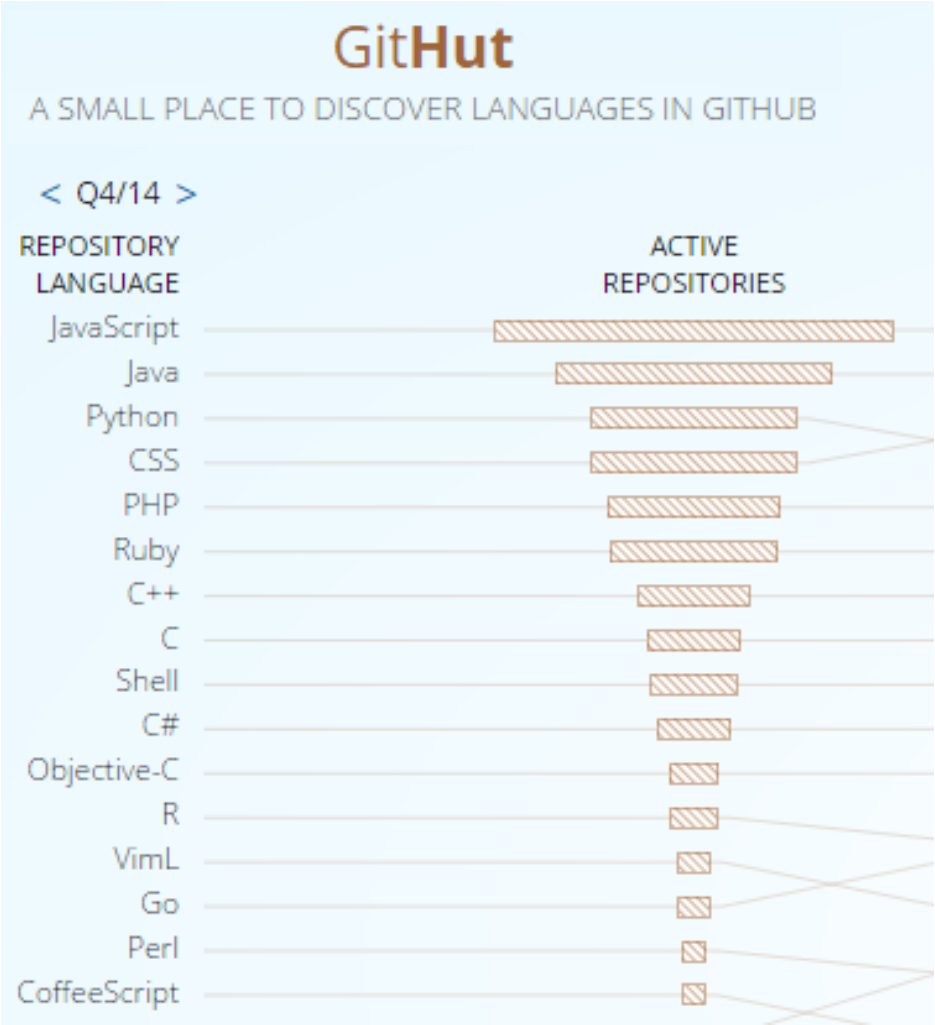
**Futuro**:
TypeScript (usado no Angular2): Deixa o javascript com tipagem e uma sintaxe mais parecida com java

**LISTA de Linguagens:**
https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js

# Mais vale a pena?

E a linguagem é conhecida?
http://githut.info/

- Web client side
- Web server side
- Mobile
- Desktop



# Uma vaga de emprego por ae ...

Requisitos

- Sólido conhecimento em AngularJS;
- Conhecimento em HTML5, CSS3, Web standards;
- Experiência como Full-Stack (PHP ou Python);
- Ter códigos no github;
- Dominar Javascript (JQuery não conta);

http://www.hunterco.com.br/?job_listing=socialbase-plataforma-de-comunicacao-interna-6-desenvolvedor-frontend-senior

# Ambiente e Ferramenta

Editores de **texto**:

- **Sublime**
- Vim
- Notepad++
- Bloco de Notas

Editores de **Online**:

JsFiddle: jsfiddle.net/
Punkler: plnkr.co/
**Cloud9**: c9.io/

Console:
Console do Navegador (F12)

Plugins para o Google Chrome:

- Wappalyzer
- Web Developer

# O que é DOM?

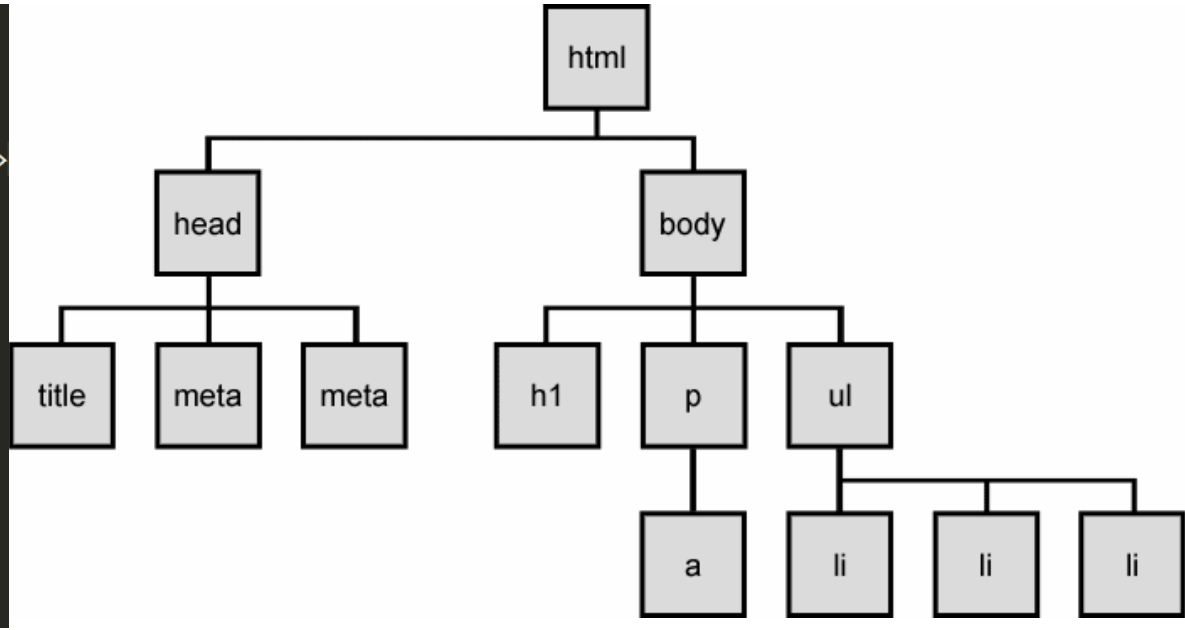
# Elementos DOM (Document Object Model)

- Criado pela W3C
- Representação de como o HTML, XHTML e XML são lidas pelo navegador
- Marcações são elementos de uma árvore
- Possível manipular usando API

# Elementos DOM (Document Object Model)

example_DOM.html

```
<html>
<head>
    <title>Web DOM</title>
    <meta charset="UTF-8">
    <meta name="description" content="DOM">
</head>
<body>
    <h1>Exemplo de Árvore DOM</h1>
    <p>
        <a href="#">LINK</a>
    </p>
    <ul>
        <li>Item 1</li>
        <li>Item 2</li>
        <li>Item 3</li>
    </ul>
</body>
</html>
```



# Elementos DOM (Document Object Model)

```
document.getElementById("total")

document.getElementsByName("telefone")

document.getElementsByClassName("centralizado");

document.getElementsByTagName("tr");
```

# A Linguagem

# Variáveis

Os tipos mais comuns são:

- **String**
- **Number**
- **Boolean**
- Object
- Array
- Function

Obs: Com o EcmaScript 6 apareceram novas: Date, Set, Map

# Variáveis

| | |
|---|---|
| **String** | “Hello” ‘Hello’ |
| **Number** | 1 2.5 1e2 1E2 5E2 |
| **Boolean** | true false |
| **Object** | {<br>“id” : 1,<br>“nome” : “Paulo Garcia”,<br>“item” : [“garrafa”, “livro”, “calculadora”]<br>} |
| **Array** | [ ] [1,2,3] [ 1, 2, 3.745, “teste”, { }, [5] ] |
| **Function** | function soma (args) { } |

# Operadores Aritméticos

| | | | |
|---|---|---|---|
| **+** | Soma/Concatenação | 1 + 1 | // 2 |
| **-** | Subtração | 1 – 1 | // 0 |
| ***** | Multiplicação | 2 * 3 | // 6 |
| **/** | Divisão | 10/4 | // 2.5 |
| **%** | Modulo | 10%2 | // 0 |

# Operadores de Atribuição

| | |
|---|---|
| **=** | x = 2 |
| **+=** | x += y |
| **-=** | x -= y |
| ***=** | x *= y |
| **/=** | x /= y |
| **%=** | x %= y |

# Operadores de Comparação

| | | |
|---|---|---|
| **==** | Igualdade | 2 == 2 //true 2==“2” //true |
| **===** | Equivalência | 2 ===2 //true 2===“2” //false |
| **!=** | Não Igual | |
| **!==** | Não Equivalente | |
| **>** | Maior que | |
| **<** | Menor que | |
| **>=** | Maior ou igual | |
| **<=** | Menor ou igual | |
| **?** | Operador ternário-> | condição ? expre1 : expre2; |

# Valores “FALSOS”

- false
- null
- undefined
- “”
- ‘’
- 0
- NaN
- [ ] -> cuidado if([]) //true, mas [] == 0 //false

Obs: O operador de negação é **!**

# Exercício 1

- 2.5 + 2.5
- 3 + 7.2
- “5”+”5”
- 2 + “12”
- “12”+2
- 5 * “10”
- 5* “Hello World”

# Resposta do Exercício 1

- 2.5 + 2.5 // 5
- 3 + 7.2 // 10.2
- “5”+”5” // 55
- 2 + “12” // ”212”
- “12”+2 // ”122”
- 5 * “10” // 50
- 5* “Hello World” // NaN

# Exercício 2

- 1 == ‘1’
- 1 === ‘1’
- 0 == []
- 1 == [“oi”]
- false == false
- false == ‘false’
- 3 == 2
- Boolean(“Uma String Qualquer”)

# Resposta do Exercício 2

- 1 == ‘1’ // true
- 1 === ‘1’ // false
- 0 == [] // true
- 1 == [“oi”] // false
- false == false // true
- false == ‘false’ // false
- 3 == 2 // false
- Boolean(“Uma String Qualquer”) // true

# Operadores Lógicos - OR

Operador **OR** é representado por **||**

O OR retorna o primeiro valor VERDADEIRO da expressão (Não necessariamente um Boolean)

Ou retorna o último valor se todos são FALSOS

# Operadores Lógicos - OR

```
"" || false || 0 || "casa"          // "casa"
0 || true || 20                     // true
10 || true                          // 10
undefined || {}                     // {}
null || undefined || 0              // 0
```

# Operadores Lógicos - AND

Operador **AND** representa usando **&&**

O AND retorna o primeiro valor FALSO da expressão (Não necessariamente um Boolean)

Ou retorna o último valor se todos são VERDADEIROS

# Operadores Lógicos - AND

```
"" && false && 0 && "casa"          // “”
0 && true && 20                     // 0
1 && true && 20                     // 20
10 && true                          // true
undefined && {}                     // undefined
null && undefined && 0              // null
```

# Quando usar o OR e AND?

codigo2.js

```
var pessoa = {   "id" : 1,
                 "nome": "Paulo Garcia",
                 "item" : ["garrafa", "livro", "calculadora"]
             };

if (pessoa && pessoa.nome){
    console.log(pessoa.nome);
}
```

# Objetos

# Objetos

codigo6.js

```
var pessoa = {
            "id" : 1,
            "nome": "Paulo Garcia",
            "item" : ["garrafa", "livro", "calculadora"]
        }

console.log(pessoa.id);          //1
console.log(pessoa['id']);       //1
pessoa.nome = "José Soarez";
console.log(pessoa.item[2]);     //calculadora
```

# Objetos

codigo6.js

```
delete pessoa.nome;      //deleta o atributo nome
pessoa.sexo = "masculino";  // cria novo atributo
pessoa.hasOwnProperty("nome");  //false
pessoa.hasOwnProperty("item");  //true
```

# Estrutura de Repetição FOR

# FOR

codigo5.js

```
var produtos = ["peixe", "biscoito/bolacha", "manteiga"];

for (var i=0; i< produtos.length; i++){
    console.log(produtos[i]);
}

for (var i in produtos){
    console.log(produtos[i]);
}
```

# Funções

# Funções

Funções são objetos de primeira ordem:

- Podem ter métodos e propriedades
- Atribuídas a variáveis
- Criadas em tempo de execução
- Podem ser passadas como parâmetros
- Podem retornar outras funções

```
function soma (n1, n2){
    return n1 + n2;
}

//Se não tiver return o valor retornado
//é undefined

//Função Anônima
var multiplica = function(n1, n2) {
    return n1 * n2;
}

soma(2, 8);       // 10
multiplica(2,3); // 6
```

codigo7.js

# Funções que retornam Funções

```
function criarFunc(func, n){
    return function(x){
        return func(n, x);
    }
}

var soma1 = criarFunc(somar, 1);
var multiplicaPor2 = criarFunc(multiplica, 2);

soma1(2);    //3
multiplicaPor2(6)    //12
```

codigo7.js

# Argumentos das Funções

Argumento Default no ES6/ES2015:

```
function somar(n1=0, n2=0){
    return n1+n2;
}
```

Argumento Default no ES5 e anteriores:

```
function somar(n1, n2){
    n1 = typeof n1 !== 'undefined' ? n1 : 0;
    n2 = typeof n2 !== 'undefined' ? n2 : 0;

    return n1+n2;
}
```

# Argumentos das Funções

Toda função recebe uma variável ***arguments***.

- **Similar (**mas não igual**)** a um Array

```
function somar(){
    var resultado = 0
    var i;
    for(i=0; i<arguments.length; i++){
        resultado += arguments[i];
    }
    return resultado;
}
```

# Funções e Escopo

Em Javascript só existem 2 tipos de escopo:

- Global
- De Função

Diferentes de outras linguagens, blocos como IF e FOR não criam escopo de variáveis. Tome cuidado!

Alternativa: IIFE (Immediately-Invoked Function Expression) OU também conhecido por “Função anônima auto-executada”

# Funções e Escopo

Uma função anônima que é auto-executada (IIFE) cria um escopo

```
var valor = 10;
(function(){
    var valor = 33;
    console.log(valor);
})();
```

```
//----- Função soma IIFE -----
(function(n1, n2){
    return n1+n2
})(2,3);
```

# Eventos!

# Eventos

Eventos disparam uma **função** em determinadas situações. Essas funções recebem o nome de **callback**.

Exemplos de eventos:

- onload: Quando o elemento terminar de carregar
- onresize: Quando o elemento DOM muda de tamanho
- onblur: Quando o elemento perde o foco
- onsubmit: Ao enviar um form

# Eventos

```
<button id="btn">Click!</button>

<script>

function startAlert(){
    alert("[Mensagem]");
}

var btn = document.getElementById("btn");
btn.addEventListener("click", startAlert);
//    OR
//btn.onclick = startAlert;
```

```
<button id="btn" onclick="startAlert();">Click!</button>
```

# Fazendo um TO DO List [projeto]

# TO DO List

- <input> para escrever as Tarefas
- Botão para Add as tarefas
- Lista <ul></ul>
- Cada item da lista terá esse formato:

<li>

[checkbox] [texto]

</li>

# Ajax e JSON

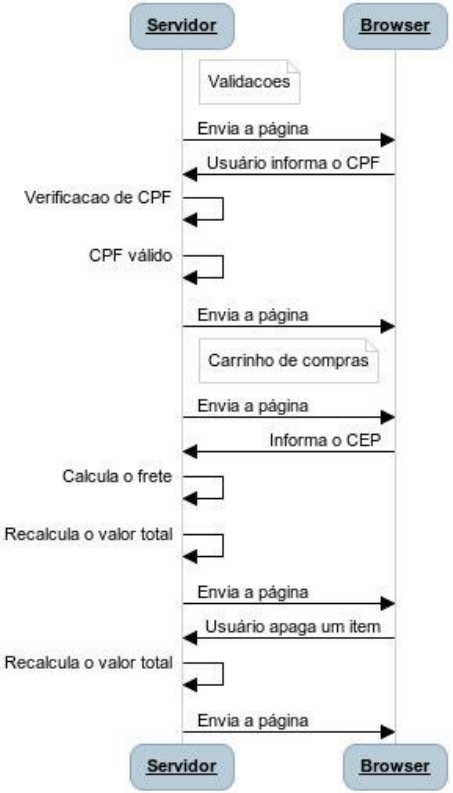
# Ajax

**Ajax** – “**A**synchronous **J**avascript **a**nd **X**ML”

Tecnologia de **comunicação assíncrona** com o Servidor

Cria requisições ao servidor depois que a página já estiver carregada. Com essas informações se cria elementos ou ações sem a necessidade de carregar a página (reload)

## Site antigo




## Site novo

# Ajax

```
function loadAjax() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (xhttp.readyState == 4 && xhttp.status == 200) {
      document.getElementById("demo").innerHTML = xhttp.responseText;
    }
  };
  xhttp.open("GET", "./ajax/ajax_info.txt", true);
  xhttp.send();
}
```

# Mas e o JSON?

**JSON** – “**J**ava**S**cript **O**bject **N**otation”

http://www.json.org/json-pt.html

É uma formatação (**padrão**) leve para troca de dados.

Vantagens:

- Leve. Mais leve que XML
- Fácil de Ler e Escrever (Humanos e máquinas)
- Independente de linguagem (não precisa ser Javascript)

# Exemplos de JSON

```
{
        “id” : 1,
        “nome” : “Paulo Garcia”,
        “item” : [“garrafa”, “livro”, “calculadora”]
}
```

Outros exemplos: color.json; flickr.json; google_maps.json;

# TO DO List com Ajax

- Inserir um botão Ajax
- Carregar elementos que estão dentro do arquivo JSON

# Referências

- BALDUINO, Plínio. “***Dominando JavaScript com jQuery***”. São Paulo, Casa do Código, 2012

- FLANAGAN, David. “***JavaScript: O Guia Definitivo***”. Editora Bookman, 6ª edição, 2012

- Introdução JS http://pt.slideshare.net/fernandosimeone/javascript-30043260

- Peculiariedades do JS http://leobetosouza.com.br/Palestra-Peculiaridades-do-JavaScript/#/

# Links úteis

**- Front-End:** (http://willianjusten.com.br/como-se-tornar-um-desenvolvedor-front-end/)

**- Front-End-Performance** (https://github.com/davidsonfellipe/awesome-wpo)

- **Introdução (js4girls)** (https://github.com/Webschool-io/js4girls/blob/master/material-didatico/etapa-1/js-introduction.md)

**- Performance Client Side** (https://developer.yahoo.com/blogs/ydn/high-performance-sites-importance-front-end-performance-7160.html)

- **Linguagens que compilam para JS** (https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js)

**- Dicas para usar o DevTool** (https://www.turbosite.com.br/blog/15-dicas-para-desenvolver-utilizando-o-chrome-devtools/)

# Links úteis

**- IIFE – Chamada imediata de função:** (http://benalman.com/news/2010/11/immediately-invoked-function-expression/)

- **Escopo e Hoisting** (http://loopinfinito.com.br/2014/10/29/hoisting-e-escopo-em-javascript/)

**- Callback HELL** (callbackhell.com)

**- JSON:** (http://www.json.org/json-pt.html)