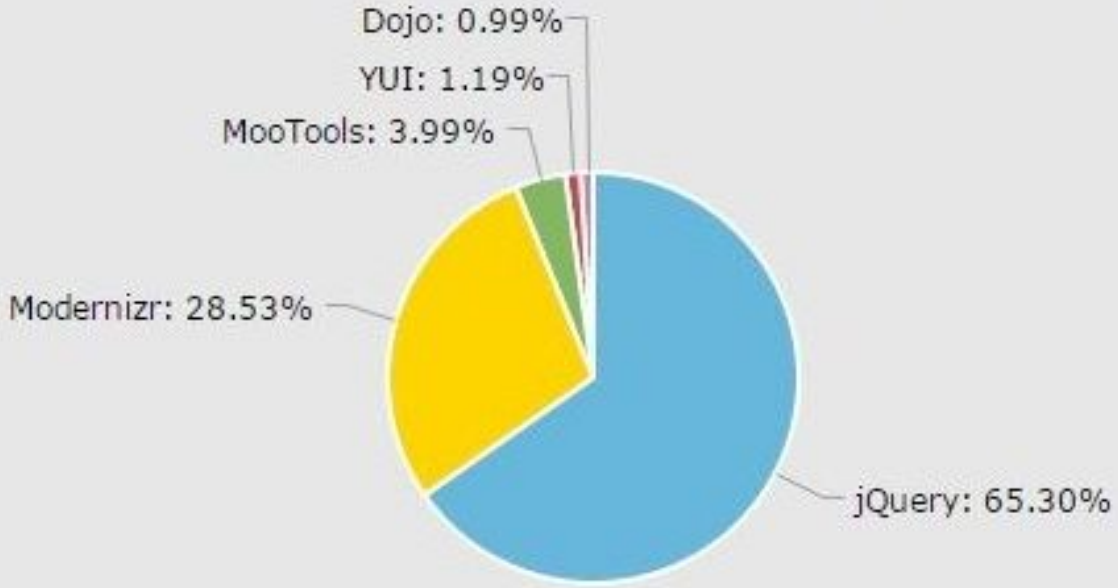
JavaScript e jQuery

Fundamentos e Biblioteca


jQuery
write less, do more.

Best JavaScript Libraries of 2015 - Beebom

Data chart was measured with Github stars retrieved on April 11, 2015



http://beebom.com/2015/04/best-javascript-frameworks-and-libraries

Beebom.com

trends.builtwith.com/javascript/javascript-library

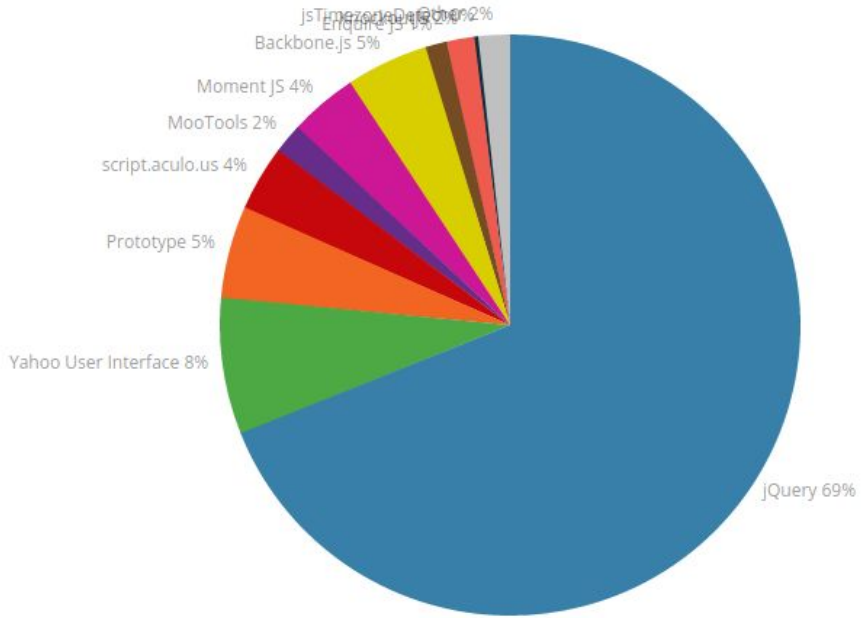
Apps ★ Bookmarks Melhorando o de Demos - jFormer Yahoo! Smush.it™ Trabalhando com Online YUI Compr Dialog | jQuery UI

Log In · Sign Up for Free

built with Trends Features Plans & Pricing Customers Resources e.g. techcrunch.com or Magento

JavaScript Library Usage

Statistics for websites using JavaScript Library technologies



Switch Chart Data

- Top 10k Sites
- Top 100k Sites
- Top Million Sites
- The Entire Internet
- Country Statistics

Top 10 Legend

- jQuery
- Yahoo User Interface
- Prototype
- script.aculo.us
- MooTools
- Moment JS
- Backbone.js
- Enquire JS
- KnockoutJS

http://jquery.com/

jQuery 2.1.3

- jquery-2.1.3.min.js - Minificada – Produção - 83kb
- jquery-2.1.3.js - Desenvolvimento – 242kb

https://developers.google.com/speed/libraries/

2.2.2 x 1.12.2

dōjō
toolkit

Sencha

http://prototypejs.org/
http://www.telerik.com/kendo-ui
https://script.aculo.us/
http://www.sencha.com/
http://mootools.net/
http://dojotoolkit.org/

Padrão comum de programação front-end.

Primeiro obtemos um objeto da página e depois modificamos seu estado e exibição.

jQuery Object

```
1 var jsCarrinho = document.getElementById("carrinho"); // com Javascript puro
2 var jsCarrinho = $("#carrinho"); // com jQuery
```

$

No exemplo, a função **$** recebe uma **String** como argumento, **contendo um seletor**, semelhante aos seletores do **CSS**.

O retorno é um **objeto**.

Padrão de utilização

Um padrão para se utilizar o jQuery é o de escrever todo o código em uma função anônima e enviá-la para a função **$** que é um atalho para a função **$(document).ready()**

```
1  $(document).ready(function(){
2      var jsCarrinho = $("#carrinho");
3  });
4
```

```
1  $(function(){
2      var jsCarrinho = $("#carrinho");
3  });
4
5
```

A principal vantagem é que o jQuery espera todos os elementos do HTML serem construídos e carregados e a página disparar o evento “DOMContentLoaded”...

DOMContentLoaded

O evento **DOMContentLoaded** é acionado quando o documento inicial HTML foi completamente carregado e analisado, sem aguardar por folhas de estilo, imagens, e subframes para encerrar o carregamento. Um evento muito diferente - **load** - deve ser usado apenas para detectar uma página **completamente** carregada. É um engano incrivelmente popular as pessoas utilizarem load quando DOMContentLoaded seria muito mais apropriado, então seja cauteloso.

https://developer.mozilla.org/pt-BR/docs/Web/Events/DOMContentLoaded

… garantindo assim que nosso código vai funcionar sem deixar nenhum elemento “para trás”.

funções básicas

text()

Quando for necessário alterar/recuperar texto de algum elemento.

```
5 document.getElementById("titulo").textContent = "Formulário de Pedidos";  // Js puro
6 $("#titulo").text("Formulário de Pedidos"); // com jQuery
7
```

val()

Utilizada para obter e alterar o valor de elementos como input , select e textarea

```
7
8 $("email").val("Informe seu e-mail"); // Alterando o valor
9 var telefone = $("telefone").val(); // Obtendo o valor
```

addClass() e removeClass()
Adicionando e Removendo Classes

```
12
13 $("h1").addClass("tituloPrincipal");
14 $("h1").removeClass("destaque");
15
```

css()

Adicionando e obtendo valor de estilo

```
16 $("#container").css({"margin":"0 auto", "width":"960px"});
17 $("body").css({"margin":"0" , "padding":"0"});
18
19 var valorEstilo = $("h1").css("width");
```

attr()

Leitura e escrita de atributos

```
21  var urlLink = $("#link2").attr("href");
22  alert(urlLink);
```

```
20
21  var urlLink = $("#link2").attr("href");
22  //alert(urlLink);
23  $("#link2").attr("href" , "http://www.terra.com.br");
24  $("#link2").text("Terra");
25
```

eventos

```
24
25 $("#mensagem").on("click", function(event){
26     alert("você clicou no botão");
27 });
28
```

exercício

file:///C:/Users/temp.ame/Desktop/jQuery/index.html

ADP - Integrador | ADP RH | ADP - Chamado | Google | Sem Parar | Em todo... | Portal ADP | Outlook Web App

Formulário de Pedidos

Google Terra Gmail

Mensagem | Modelo 1 | Modelo 2


Aula de jQuery
file:///C:/Users/temp.ame/Desktop/jQuery/index.html
Aplicativos
ADP - Integrador
ADP RH
ADP - Chamado
Google
Sem Parar | Em todo...
Portal ADP
Outlook Web App
Cadastro de Pedidos
Bing
Uol
Outlook
Mensagem
Modelo 1
Modelo 2

encadeamento de funções

Podemos chamar uma função diretamente em seguida de outra.

```
22  //alert(urlLink);
23  $("#link2").attr("href" , "http://www.terra.com.br");
24  $("#link2").text("Terra");
25
```

```
22  //alert(urlLink);
23  $("#link2").attr("href" , "http://www.terra.com.br").text("Terra");
24
```

Seletores

```
3     //traz todos ul da página
4     var lista = $("ul");
5     //elemento com id nome
6     var nome = $("#nome");
7     // os elementos com a classe item
8     var item = $(".item");
9     //traz todos os inputs com o estado "selected"
10    var selecionados = $("input:selected");
11    // contrario do seletor de cima
12    var no_selecao = $("input:not(selected)");
13    //os "lis" da ul com id lista
14    var itens = $("#lista li");
15
```

Iterando sobre elementos com each()

```
12  <ul id="lista-pizzas">
13      <li id="pizza1">Calabresa</li>
14      <li id="pizza2">Mussarela</li>
15      <li id="pizza3">Portuguesa</li>
16  </ul>
17
18  <ul id="lista-bebidas">
19      <li id="coca">Cola-cola</li>
20      <li id="pepsi">Pepsi</li>
21      <li id="dolly">Dolly</li>
22  </ul>
```

```
4       var listaDeId = new Array();
5
6       $("li").each(function(){
7           var idElemento = this.id;
8           listaDeId.push("Id - " + idElemento);
9       });
10
11      console.dir(listaDeId);
```

a função **each()** permite iterar sobre todos os itens retornados pelo seletor.
No exemplo anterior, obtemos o id de todos os **li**’s da página, concatenando com “id - ”, para isto bastou criarmos um array fora do **each** e dentro de **each** obtemos os id’s e adicionamos no array.

```
3
4   var listaDeId = new Array();
5
6   $("li").each(function(){
7       var idElemento = this.id;
8       listaDeId.push("Id - " + idElemento);
9   });
10
11  console.dir(listaDeId);
12
13  for(var i = 0 ; i < listaDeId.length ; i++){
14      console.log(listaDeId[i]);
15  }
```


the End