Criar uma aplicação móvel com jQuery Mobile

Introdução

Já muito foi escrito sobre a biblioteca jQuery para JavaScript [https://jquery.com/], incluindo alguns artigos na Revista Programar (por exemplo, "Mitos do jQuery" [https://www.revista-programar.info/artigos/mitos-do-jquery/] e "jQuery: Usar ou Não Usar?" [https://www.revista-programar.info/artigos/jquery-usar-ou-nao-usar/]).

No entanto, existem outros projectos "irmãos" do projecto jQuery que são igualmente interessantes para programadores e designers Web, como as *frameworks* jQuery Mobile e jQuery UI. Neste artigo, foco-me na jQuery Mobile explicando a sua filosofia de programação, e mostrando alguns dos componentes principais para a criação de uma aplicação móvel.

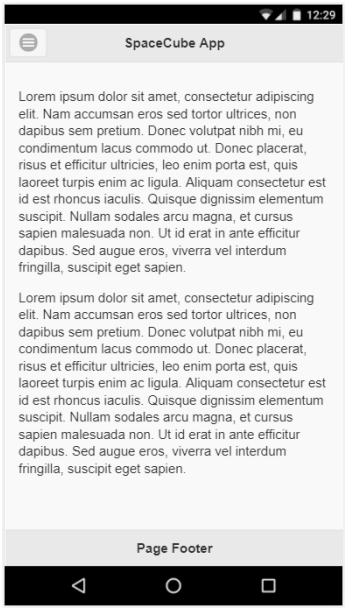
O que é, e porquê usar?

jQuery Mobile [http://jquerymobile.com/] é, ao contrário da biblioteca jQuery, uma *framework* para aplicações móveis. Usar jQuery Mobile implica seguir uma estrutura pré-definida para a nossa aplicação. Esta *framework* permite-nos criar rapidamente aplicações Web com um *look and feel* semelhante a aplicações móveis nativas. A Figura 1 mostra um exemplo do que se consegue facilmente obter com jQuery Mobile. A imagem ilustra uma aplicação com uma gaveta de navegação lateral (Figure 1, à direita) acessível através de um botão no cabeçalho (Figura 1, à esquerda) da aplicação.

Uma das vantagens da jQuery Mobile é que a estrutura da aplicação é definida inteiramente através de HTML. Por exemplo, a gaveta de navegação da aplicação da Figura 1, é construída através do código HTML seguinte:

```
<div data-role="panel" id="mypanel1" data-
display="overlay">
 <img src="space-cube.svg" width="240" >
 <ul data-role="listview">
   <li><a href="#first-page">Page 1</a></li>
   <li><a href="#second-page">Page 2</a></li>
   <li><a href="#third-page">Page 3</a></li>
 </ul>
</div>
```

Essencialmente, a *framework* jQuery Mobile dá-nos um conjunto de elementos de interface gráfica com aspecto visual e comportamento adequados a aplicações móveis. Esta *framework* pode ser usada na construção de aplicações móveis finais, mas pode também ser útil para criação de protótipos funcionais. O facto de ser baseada em tecnologias Web, pode torná-la especialmente útil para designers multimédia interessados em criar e testar um protótipo funcional de uma aplicação móvel: é possível definir todos os ecrãs e transições entre ecrãs que não dependam de lógica específica da aplicação sem necessidade sequer de usar JavaScript.



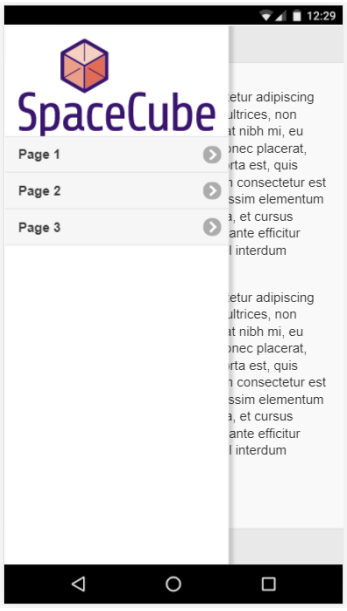


Figura 1. Possível aspecto de uma aplicação desenvolvida com jQuery Mobile.

Filosofia de programação

A filosofia de programação jQuery Mobile assenta essencialmente na utilização de atributos data-* nos elementos HTML e classes CSS pré-definidas. Por exemplo, a definição de um ecrã da aplicação é feita incluindo simplesmente um com um id e com o atributo data-role="page" (em jQuery Mobile, um ecrã é designado por *page*):

```
<body>
  <div data-role="page" id="pagina-1">
    <!-- conteúdo do ecrã -->
  </div>
</body>
```

Apesar de não ser absolutamente obrigatório, a *framework* jQuery Mobile é orientada principalmente para aplicações de página única (*Single Page Applications* - *SPA*), pelo que uma aplicação com dois ecrãs definiria simplesmente dois como descendentes do <body> do documento HTML:

```
<body>
  <div data-role="page" id="pagina-1">
    <!-- conteúdo do primeiro ecrã -->
  </div>

  <div data-role="page" id="pagina-2">
    <!-- conteúdo do segundo ecrã -->
  </div>
</body>
```

Configuração

A configuração da *framework* no projecto Web passa simplesmente pela inclusão dos ficheiros CSS e JavaScript (as versões poderão variar):

```
<link rel="stylesheet"
href="http://code.jquery.com/mobile/1.4.5/jquery.
mobile-1.4.5.min.css" />
<script src="http://code.jquery.com/jquery-
1.11.1.min.js"></script>
<script
src="http://code.jquery.com/mobile/1.4.5/jquery.m
obile-1.4.5.min.js"></script>
```

De notar que a jQuery Mobile depende da biblioteca jQuery pelo que é importante incluir primeiro o ficheiro JavaScript jQuery e só depois o ficheiro jQuery Mobile.

Igualmente importante é a inclusão de um elemento <meta> no cabeçalho documento HTML para garantir que a aplicação é mostrada correctamente num dispositivo móvel:

```
<meta name="viewport" content="width=device-
width, initial-scale=1">
```

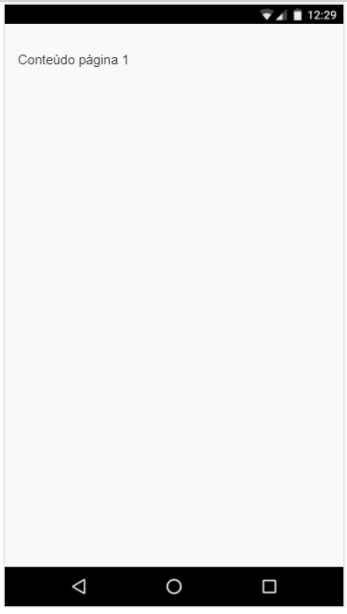
Principais componentes

Páginas

Uma página é o componente básico em jQuery Mobile que representa um ecrã da aplicação. Todos os outros componentes que iremos ver são inseridos dentro de uma página. Em termos de estrutura HTML, uma página é definida através de um com um atributo data-role="page". É importante atribuir também um "id" ao , uma vez que este será a forma de nos referirmos à página para, por exemplo, permitir navegação entre ecrãs da aplicação. O conteúdo visível da página deve ser colocado dentro de um com role="main" e class="ui-content">. O Exemplo 1

```
<div data-role="page" id="pagina-1">
  <div role="main" class="ui-content">
    <p>Conteúdo página 1</p>
  </div>
  <!-- /content -->
</div>
<!--/pagina-1 -->

<div data-role="page" id="pagina-2">
  <div role="main" class="ui-content">
    <p>Conteúdo página 2.</p>
  </div>
  <!-- /content -->
</div>
<!-- /pagina-2 -->
```



Exemplo 1 – Páginas.

Código completo: https://goo.gl/Byg1jc | Ver no dispositivo móvel: https://goo.gl/Y9Tfgk

demonstra uma aplicação com apenas duas páginas.
De notar que ao "executar" a aplicação, apenas vemos o conteúdo da primeira página. Para vermos a segunda página temos de adicionar alguma forma de navegação entre ecrãs da nossa aplicação. Uma forma de permitir navegação entre ecrãs é criando simplesmente links internos tal como faríamos numa página web tradicional, usando os ids como URL internos: Link para página 1.
O Exemplo 2 ilustra esta ideia, permitindo-nos navegar entre os dois ecrãs, pressionando nos links respectivos.

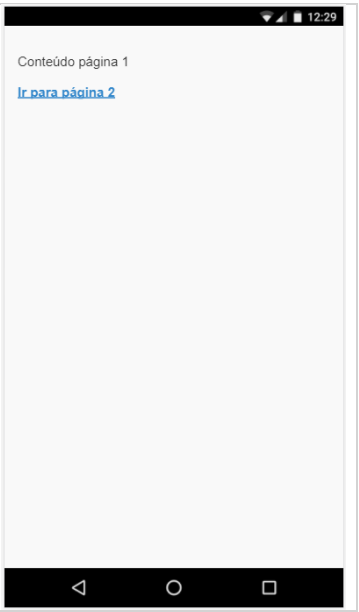
Numa aplicação móvel, no entanto, não é usual pressionarmos em links para navegar - tipicamente, usamos botões.

Botões

A criação de botões ilustra bem a vantagem da utilização da *framework* jQuery Mobile: para criarmos um botão apenas temos de adicionar um atributo ao elemento <a> que usámos para criar a navegação entre ecrãs – o atributo data-role="button". Isto cria um botão que ocupa toda a largura do ecrã, como se pode ver no "Botão 1" do Exemplo 3. Se quisermos que o botão ocupe apenas a largura necessária para mostrar o texto no interior, podemos acrescentar o atributo data-inline="true" ("Botão 2"). Podemos também

```
<div data-role="page" id="pagina-1">
 <div role="main" class="ui-content">
  <p> Conteúdo página 1 </p>
  <a href="#pagina-2">Ir para página 2</a>
 </div>
 <!-- /content -->
</div>
<!--/pagina-1 -->

<div data-role="page" id="pagina-2">
 <div role="main" class="ui-content">
  <p>Conteúdo página 2.</p>
  <a href="#pagina-1">Ir para página 1</a>
 </div>
 <!-- /content -->
</div>
<!-- /pagina-2 -->
```



Exemplo 2

Código completo: https://goo.gl/N9YBZL | Ver no dispositivo móvel: https://goo.gl/7WMS62

```
<div data-role="page" id="pagina-1">
 <div role="main" class="ui-content">
  <p> Conteúdo página 1 </p>
  <a data-role="button" href="#pagina-2">Botão 1</a>
  <a data-role="button" data-inline="true" href="#pagina-2">Botão 2</a>
  <a data-role="button" data-inline="true" data-icon="navigation"
href="#pagina-2">Botão 3</a>
  <a class="ui-btn ui-btn-icon-left ui-icon-navigation ui-btn-inline ui-
shadow" href="#pagina-2">Botão 4</a>
 </div>
 <!-- /content -->
</div>
<!--/pagina-1 -->
```

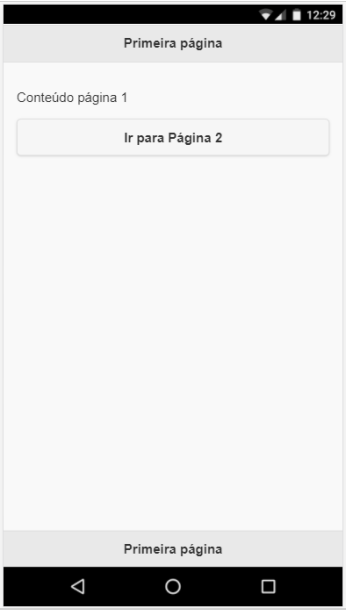
Exemplo 3 - Botões

Código completo: https://goo.gl/NLvn7M | Ver no dispositivo móvel: https://goo.gl/ZY4MPs

```html
<div data-role="page" id="pagina-1">
 <div data-role="header" data-position="fixed">
  <h1>Primeira página</h1>
 </div>
 <!-- /header -->

 <div role="main" class="ui-content">
  <p> Conteúdo página 1 </p>
  <a data-role="button" href="#pagina-2">Ir para Página 2</a>
 </div>
 <!-- /content -->

 <div data-role="footer" data-position="fixed">
  <h1>Primeira página</h1>
 </div>
 <!-- /footer -->
</div>
<!--/pagina-1 -->
```



Exemplo 4 – Cabeçalho e rodapé

Código completo: https://goo.gl/r7BVp7 | Ver no dispositivo móvel: https://goo.gl/AjavMT

acrescentar um ícone ao botão indicando o nome do ícone no atributo data-icon ("Botão 3") – a lista de ícones pré-definidos pode ser consultada em https://api.jquerymobile.com/icons/. O Exemplo 3 mostra alguns dos tipos de botões que podemos criar desta forma. De notar que estes atributos "data-*" são na verdade usados pela *framework* jQuery Mobile para determinar que classes CSS irá atribuir ao elemento. O último botão no Exemplo 3 mostra como podemos obter o mesmo efeito indicando directamente as classes CSS a usar ("Botão 4").

Cabeçalho e rodapé

É comum as aplicações móveis terem um cabeçalho e/ou um rodapé com o logótipo da aplicação e com botões de navegação. Estes elementos podem ser incluídos num ecrã da aplicação jQuery Mobile inserindo um elemento com data-role="header" (ou data-role="footer") dentro da página correspondente ao ecrã. Por exemplo, para adicionar um cabeçalho e rodapé à nossa primeira página teríamos o código do Exemplo 4. (Como normalmente queremos que o cabeçalho e rodapé fiquem sempre no topo e fundo da janela da aplicação devemos usar também o atributo data-position="fixed", caso contrário ficarão no fluxo de conteúdo da página e sujeitos ao *scroll*.)

De notar que o código do cabeçalho e do rodapé tem de ser repetido em cada página em que pretendamos que estejam presentes, caso contrário alguns ecrãs da nossa aplicação apresentarão cabeçalho/rodapé e outros não.

Botões e navegação no cabeçalho ou rodapé

Os cabeçalhos ou rodapés das aplicações móveis servem, muitas vezes, como ponto de acesso a funcionalidades da aplicação ou como forma de navegação entre os vários ecrãs. Se quisermos incluir botões no cabeçalho ou rodapé podemos fazê-lo através de um elemento com data-role="controlgroup" dentro do elemento que representa o cabeçalho. Este elemento permite-nos incluir vários botões e controlar o seu posicionamento no cabeçalho ou rodapé. O Exemplo 5 mostra o código a incluir no cabeçalho para criar um ecrã com dois botões no cabeçalho. No elemento que define o *controlgroup*, de notar a utilização dos atributos data-type="horizontal" para garantir que os botões são dispostos horizontalmente e da classe CSS "ui-btn-right" para alinhar os botões à direita (podiamos obviamente usar "ui-btn-left" para alinhar à esquerda). De notar também o uso do atributo data-iconpos="notext" nos botões para esconder o texto, ficando apenas o ícone do botão.

Quando o objectivo é criar uma área de navegação entre páginas no cabeçalho ou rodapé, existe um componente especializado – a barra de navegação. A inclusão de uma barra de navegação pode ser feita no cabeçalho ou rodapé (ou até no corpo da página). No Exemplo 6, a inclusão é feita no rodapé. Uma barra de navegação é simplesmente um elemento com data-role="navbar". No interior do inserimos uma lista não ordenada com um link (elemento âncora <a>) em cada item. Automaticamente, os links são convertidos em botões (neste caso não é

```
<div data-role="header">
  <div data-role="controlgroup" data-type="horizontal" class="ui-btn-right">
    <a data-icon="star" data-role="button" data-inline="true" data-
iconpos="notext" href="#">Favorito</a>
    <a data-icon="gear" data-role="button" data-inline="true" data-
iconpos="notext" href="#">Settings</a>
  </div>
  <h1>Primeira página</h1>
</div>
```

Exemplo 5 – Botões no cabeçalho

Código completo: https://goo.gl/8LFCd5 | Ver no dispositivo móvel: https://goo.gl/hgoC8W

```
<div data-role="footer" data-position="fixed">
  <div data-role="navbar">
    <ul>
      <li>
        <a data-icon="home" class="ui-btn-active" href="#">Home</a>
      </li>
      <li>
        <a data-icon="gear" href="#">Settings</a>
      </li>
      <li>
        <a data-icon="info" href="#">Info</a>
      </li>
    </ul>
  </div>
  <h1>Primeira página</h1>
 </div>
```

Exemplo 6 – Barra de navegação

Código completo: https://goo.gl/rLoJVw | Ver no dispositivo móvel: https://goo.gl/c23dCK

necessário adicionar o atributo data-role="button"). Para indicar que um botão está activo (ou seja que o ecrã visível corresponde a um determinado botão na barra de navegação) podemos usar a classe CSS "ui-btn-active". Note-se que no Exemplo 6 o rodapé apenas contém a navegação. No entanto, é possível combinar a barra de navegação com outros elementos.

Gaveta de navegação

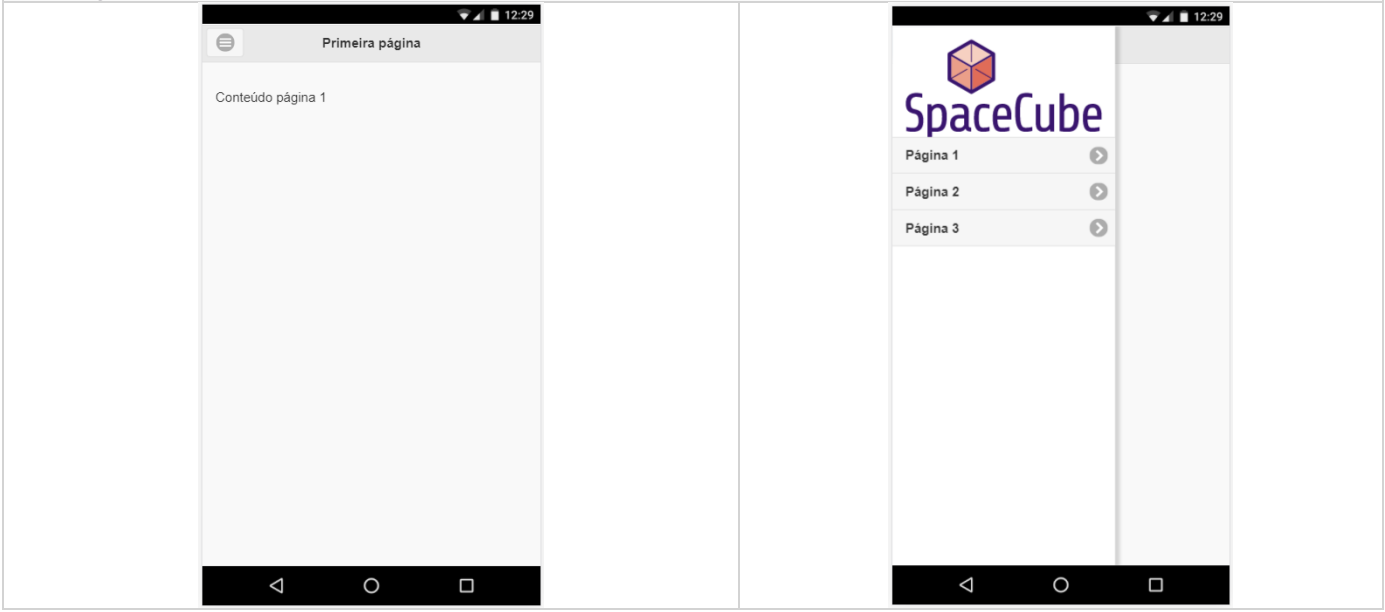
A barra de navegação pode ser usada quando o número de elementos de navegação é relativamente reduzido. Para além disso, os botões na barra de navegação consomem espaço que pode ser importante para mostrar conteúdo mais útil. Uma alternativa à barra de navegação (ou um complemento) é a gaveta de navegação. A gaveta de navegação é um painel que podemos abrir ou fechar pressionando num botão (tipicamente o botão é apresentado no cabeçalho). Para criarmos uma gaveta de navegação para uma determinada página adicionamos um elemento com data-role="panel" e com um id. Para listar as opções de navegação no interior da gaveta usa-se tipicamente uma *listview* (uma lista não ordenada com data-role="listview") com links em cada item. Para possibilitar a abertura da gaveta de navegação devemos incluir um *controlgroup* no cabeçalho com um botão que refira o id que escolhemos para o painel que representa a gaveta de navegação. O Exemplo 7 mostra o código completo de uma página com uma gaveta de navegação. O atributo data-display no painel que

```
<div data-role="page" id="pagina-1">
 <div data-role="header">
  <h1>Primeira página</h1>
  <div data-role="controlgroup" data-type="horizontal" class="ui-btn-left">
   <a data-icon="bars" data-iconpos="notext" data-role="button" href="#gaveta1">Menu</a>
  </div>
 </div>
 <!-- /header -->

 <div data-role="panel" id="gaveta1" data-display="overlay">
  <!-- Fake logo from: https://github.com/pigment/fake-logos -->
  <img src="https://s3-us-west-2.amazonaws.com/s.cdpn.io/455884/space-cube.svg" width="240">
  <ul data-role="listview">
   <li><a href="#pagina-1">Página 1</a></li>
   <li><a href="#pagina-2">Página 2</a></li>
   <li><a href="#pagina-3">Página 3</a></li>
  </ul>
 </div>
 <!-- /gaveta navegação -->

 <div role="main" class="ui-content">
  <p> Conteúdo página 1 </p>
 </div>
 <!-- /content -->

</div>
<!--/pagina-1 -->
```



Exemplo 7 – Gaveta de navegação

Código completo: https://goo.gl/P9dEHZ | Ver no dispositivo móvel: https://goo.gl/Vg99iY

representa a gaveta de navegação permite-nos configurar a forma como a gaveta surge ao utilizador ("overlay" é a forma mais comum, em que a gaveta é puxada para cima do conteúdo do ecrã; "reveal" cria uma gaveta fixa e é o conteúdo do ecrã que se move para mostrar a gaveta por baixo; "push" cria uma gaveta ao lado do conteúdo do ecrã e ambos de movem para mostrar ou esconder a gaveta.)

De notar que a gaveta de navegação tem de ser incluída em cada página (com um id diferente para cada página). É possível evitar este tipo de repetição, mas neste momento, isso obriga ao uso de algum código JavaScript, por isso não abordo essa forma de inclusão de gavetas de navegação.

Conclusão

Há, obviamente, muito mais a dizer sobre jQuery Mobile, não apenas sobre componentes de interface gráfica disponíveis, mas também sobre a API JavaScript que nos permite instanciar e controlar programaticamente os componentes de interface gráfica, e também sobre a possibilidade de customização da aparência dos componentes. Com esta introdução, o leitor terá ficado com uma ideia das possibilidades que a framework jQuery Mobile nos dá, da sua estrutura, e dos principais componentes de interface gráfica disponíveis.