Tema 9

Integração de componentes em páginas *Web*

Objetivos:

- Componentes de *Twitter Bootstrap*.
- Gráficos usando *Highcharts JS*.
- Mapas e elementos georeferenciados.

9.1 Páginas Web

As páginas *Web* atuais necessitam de possuir uma estrutura e estilos muito sofisticados. Cada vez mais a interação faz-se através de interfaces *Web* e menos através de aplicações que os utilizadores necessitem de instalar no seu computador.

Bons exemplos desta tendência são aplicações como o Google Mail [1], o Google Docs [2] ou o Microsoft Office 365 [3].

Como vimos na aula dedicada ao tema de páginas *Web*, existem componentes que permitem acelerar o desenvolvimento, fornecendo, de uma maneira simplificada, estrutura e estilo agradável. Estes sistemas permitem ainda melhorar a interação através da inclusão de componentes "pré-feitos" que aumentam a atratividade e funcionalidade de uma página.

Isto reforça a ideia de que o desenvolvimento de um sistema deve favorecer componentes já existentes, disponibilizados por outras entidades, não devendo o programador pretender desenvolver toda a programação e estilo. Só em casos muito específicos se pode tomar esta atitude, à custa de um maior esforço de desenvolvimento e de manutenção do código desenvolvido.

As páginas *Web* são um dos domínios onde mais se utilizam recursos externos, sendo o resultado visível na quantidade e complexidade das páginas disponíveis na *Internet*. Este guião irá abordar alguns componentes que podem ser utilizados para enriquecer uma página *Web*, com foco em componentes dinâmicos para a representação de gráficos e de mapas.

Para tal, irá fazer-se uso de uma página construída usando o *Twitter Bootstrap*, sendo que serão adicionados componentes de forma a enriquecer as suas funcionalidades.

Relembra-se que uma página mínima, fazendo uso de *Twitter Bootstrap* pode ser criada fazendo inclusão da página de estilos correta. De forma a suportar um modelo de interação mais rico, poderá também ser adicionado um conjunto de *scripts*.

O exemplo que se segue resulta numa página muito minimalista com *Twitter Bootstrap* mas que poderá servir de base para o restante guião:

```
<!DOCTYPE html>
<html lang="pt">
    <head>
        <meta charset="UTF-8">
        <title>Integração de Componentes em Páginas WEB</title>

        <!-- Bootstrap minified CSS -->
        <link rel="stylesheet"
            href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
        <!-- Optional theme -->
        <link rel="stylesheet"
            href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap-theme.min.css">
        <!-- Personal Styles -->
        <link rel="stylesheet" href="css/style.css">
        <!-- jQuery library -->
        <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
        <!-- Bootstrap JavaScript -->
        <script
            src="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>
    </head>

    <body>
        <div class="container">
            Conteúdo da página
        </div>
    </body>
</html>
```

De realçar que ao contrário do exemplo fornecido numa aula anterior, neste caso a página inclui componentes de Cascading Style Sheets (CSS)[4] e de JavaScript (JS)[5]. Existe igualmente um elemento ** ** com a classe **container**, no qual se deverá incluir o conteúdo da página.

Exercício 9.1

No diretório da disciplina crie um subdiretório chamado **projeto** onde deve executar os exercícios deste guião.

Entre no diretório recém criado e crie uma página (designada por **index.html**) com o conteúdo do exemplo anterior. Verifique o funcionamento correto desta página.

9.2 Componentes

Uma página *Web* é composta por várias secções, definindo a sua estrutura, sendo que em cada secção existirão componentes variados tais como menus, sub-menus, painéis, imagens, entre outros. A linguagem HyperText Markup Language (HTML)[6] fornece já um grande conjunto de ferramentas que possibilitam a criação de páginas ricas, no entanto por vezes é insuficiente aplicar as marcas de forma isolada, sendo necessário conjugar a marca (HTML), um estilo (CSS) e ações (JS). Para uma lista completa dos componentes disponíveis para *Twitter Bootstrap*, consultar `http://getbootstrap.com/components`. Neste guião será abordado um número reduzido de componentes.

9.2.1 Menus e Sub-Menus

Os menus e sub-menus são vitais para a navegação numa página. Tipicamente estes são implementados através do recurso a marcas **<ul>** e **li**. Ou seja, um menu é tratado como uma lista de items. O estilo aplicado a essa lista é que lhe atribui o aspeto típico de um menu. Usando *Twitter Bootstrap*, o menu principal de uma página chama-se barra de navegação e é incluída dentro de elementos **<nav>** e ** **.

```
<nav class="navbar navbar-default">
    <div class="navbar-header">
        <a class="navbar-brand" href="#">Introdução à Engenharia Informática</a>
    </div>

    <div class="navbar-collapse collapse">
        <ul class="nav navbar-nav">
            <li class="active"><a href="#">Topic A - Popups</a></li>
            <li><a href="#">Topic B - Graphics</a></li>
        </ul>

        <ul class="nav navbar-nav navbar-right">
            <li class="active"><a href="about.html" target="_blank">
                <span class="glyphicon glyphicon-user"></span> About</a>
            </li>
        </ul>
    </div>
</nav>
```

O elemento **<nav>** inicial possui classes (**navbar** e **navbar-default**) que servem para definir o posicionamento e o aspeto (borda, sombra).[1] Este elemento inclui marcas que indicam o conteúdo, respetivamente: uma ** ** com classe **navbar-header** indicando o nome da página; outra ** ** com classes para definirem estilo; duas marcas **<ul>** para a construção do menu em si. Cada elemento (**<li>**) da lista será uma opção do menu.

Exercício 9.2

Adicione uma barra de navegação à sua página. A barra deve ser incluída no início do elemento com classe **container** em alternativa ao texto **Conteúdo da página**. Verifique o funcionamento correto desta página.

Exercício 9.3

Repare que a barra de navegação apresenta do lado direito um ícone que representa o criador da página com uma ligação para uma página designada por **about.html**.

Crie esta página com o código seguinte e personalize-a acrescentando o seu nome e a data atual. Verifique se a página está a funcionar corretamente.

```
<!DOCTYPE html>
<html lang="pt">
    <head>
        <meta charset="UTF-8">
        <title>About</title>

        <link rel="stylesheet"
            href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
        <!-- Personal Styles -->
        <link rel="stylesheet" href="css/style.css">
    </head>

    <body class = "style_about">
        <h2>Introdução à Engenharia Informática</h2>
        <br><br>
        <h2>Integração de Componentes - Component Integration</h2>
        <h2>(Popups - Graphics - Maps - Images)</h2>
        <br><br>
        <img src="images/ua.png" alt="UA">
        <h2>Author - Date</h2>
        <br>
        <input type="button" class="btn btn-success" value="Close" onclick="self.close()">
    </body>
</html>
```

[1] A marca **<nav>** foi introduzida no HTML5. Em versões anteriores poderá usar-se uma marca ** **.

Exercício 9.4

Para obter o efeito desejado terá que acrescentar o ficheiro de estilo personalizado que está incluído no cabeçalho da página.

Começe por criar um diretório **css** no diretório **`projeto`**. Entre no diretório recém criado e crie o ficheiro **style.css** com o código seguinte:

```
/* style for the about page */

.style_about {
    color: black;
    background-color: red;
    text-align: center;
}

/* Create two columns (left and right boxes) with the same size */
/* Put map and image side by side */
.columnleft { float: left; height: 450px; width: 550px; margin: 10px; }
.columnright { float: right; height: 450px; width: 550px; margin: 10px; }
.row:after { content: ""; display: table; clear: both; }
```

Crie também um diretório **images** no diretório **`projeto`** e coloque lá o ficheiro **ua.png** disponível no *elearning*.

Depois verifique se a página está a funcionar corretamente. Pode alterar as cores a seu gosto.

Para obter sub-menus é necessário utilizar a classe **`dropdown`** num item do menu. Os elementos do sub-menu são implementados através de uma nova lista.

```
<li class="dropdown">
    <a href="#" class="dropdown-toggle" data-toggle="dropdown">
                Topic C<span class="caret"></span></a>
        <ul class="dropdown-menu">
            <li><a href="#">Topic C - Map</a></li>
            <li class="divider"></li>
            <li><a href="#">Topic C - Image</a></li>
        </ul>
</li>
```

Exercício 9.5

Adicione o exemplo acima à sua página logo a seguir ao elemento **`Topic B`** e alinhe-o corretamente. Verifique o funcionamento correto desta página.

O resultado deverá ser o apresentado na Figura 9.1.

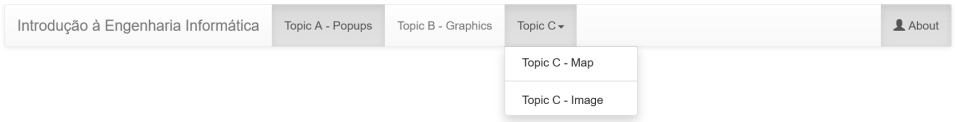

Figura 9.1: Barra de navegação

De forma a navegar na página, é possível criar ligações internas. Estas ligações funcionam através do atributo **id** de cada elemento. Por exemplo, se existir um elemento com **id=topicA**, uma marca **<a>** pode enviar o utilizador para esse elemento usando **#topicA** no seu atributo **href**.

O exemplo seguinte demonstra este caso.

```
<li class="active"><a href="#topicA">Topic A - Popups</a></li>
...
<h2 id="topicA" style="text-align:center">Topic A - Popups</h2>
```

Exercício 9.6

Na barra de navegação crie ligações (**href="#..."**) que possibilite ir rapidamente para cada um dos dois tópicos principais (Popups e Graphics) e para os dois sub-tópicos (Map e Image).

Adicione duas marcas **<h2>** para cada um dos dois primeiros tópicos e **execute os exercícios por baixo do tópico correspondente.**

9.2.2 Popups

Os *Popups* são bastante úteis para apresentar mensagens importantes aos utilizadores. Tipicamente sobrepõem-se à página e necessitam de uma ação explícita para desaparecerem, tal como clicar num botão. Usando *Twitter Bootstrap*, os *Popups* são implementados com recurso à classe **modal** aplicado à marca ** **.

Tal como demonstrado no exemplo que se segue, existe uma marca ** ** inicial da classe **modal** que depois possui diversas outras marcas ** ** com cada uma das áreas do elemento. Pode identificar um cabeçalho (**modal-header**), um corpo (**modal-body**) e um rodapé (**modal-footer**).

Note também a existência de duas marcas **button** correspondendo ao botão com o símbolo **X**, que tipicamente existe no topo de uma janela, e ao botão com o texto *Close*.

```
<h2 id="topicA" style="text-align:center">Topic A - Popups</h2>
<div id="myPopup" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button
                    type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;
                </button>
                <h4 class="modal-title">Popup Title</h4>
            </div>
            <div class="modal-body">
                <p>Popup Contents</p>
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
            </div>
        </div>
    </div>
</div>
```

Existem vários outros atributos, tais como **`data-dismiss`**, ou **`aria-hidden`** que servem para adicionar funcionalidades. O resultado poderá ser o apresentado na Figura 9.2.

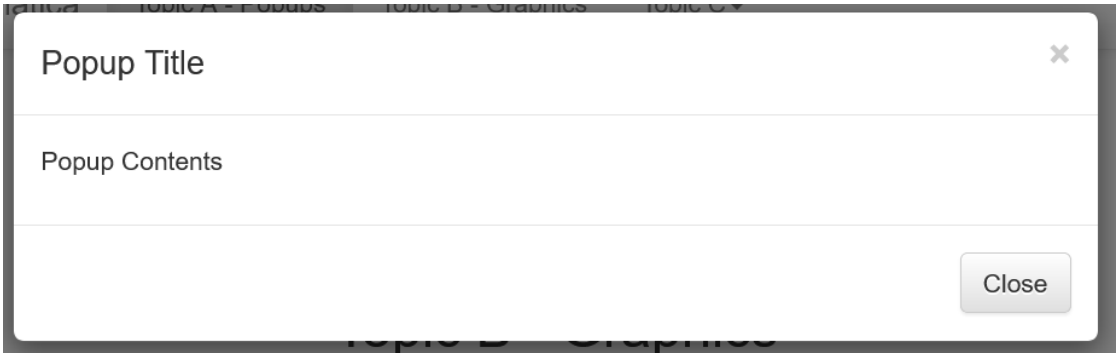

Figura 9.2: *Popup*

Exercício 9.7

Adicione o exemplo acima. Repare que nada é apresentado. No entanto pode ver no código fonte que isto se deve apenas a um atributo do estilo.

Os *Popups* não são sempre apresentados apesar do seu conteúdo permanecer na página. Uma maneira simples de ativar o *Popup* será através de um botão, usando o atributo **`data-target`**. O valor deste atributo terá de corresponder ao valor do atributo **`id`** do *Popup* a apresentar.

```
<button class="btn btn-primary" data-toggle="modal" data-target="#myPopup">
    Popup Throw
</button>
```

Exercício 9.8

Implemente um exemplo com um botão que ativa um *Popup* e adicione-o à sua página. Verifique que o *Popup* já é apresentado.

9.3 Gráficos

O HTML5 suporta gráficos através dos elementos **<canvas>** e **<svg>**. No entanto, há recursos externos de JS que facilitam muito a criação de gráficos complexos. É o caso da biblioteca *Highcharts JS* que iremos introduzir nesta secção.

Para adicionar um gráfico com esta biblioteca é preciso:

1. Incluir, na marca **head**, os recursos de JS que permitem desenhar gráficos. No caso do *Highcharts JS* é necessária uma linha:

   ```
   <!-- Highcharts JavaScript -->
   <script src="http://code.highcharts.com/highcharts.js"></script>
   ```

2. Incluir na marca **body** da página (no tópico B) um elemento que irá conter o gráfico.

   ```
   <h2 id="topicB" style="text-align:center">Topic B - Graphics</h2>
   <div id="myGraph" style="width: 400px; height: 300px;"></div>
   ```

   Neste caso, o gráfico terá 400px por 300px de dimensão. O atributo **id** permitirá identificar o elemento nos scripts.

3. Incluir na marca **head** da página o script:

   ```
   <!-- Graphic Draw JavaScript -->
   <script src="js/graphic.js"></script>
   ```

   cujo conteúdo (ficheiro JS **graphic.js** que deve ser colocado num diretório designado por **js**) deve configurar o gráfico e os dados:

```
function draw() {
  $("#myGraph").highcharts({
    chart: { type: "line" },
    title: { text: "Temperatures Average" },
    xAxis: { categories: ["Jan", "Fev", "Mar", "Abr", "Mai", "Jun",
                          "Jul", "Ago", "Set", "Out", "Nov", "Dez"]
           },
    series:[
             { name: "Lisboa",
               data: [7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6]
             },
           ]
  });
}
```

4. Invocar a função de JS que desenha o gráfico (a seguir ao elemento do ponto 2).

```
<button class="btn btn-primary" onclick="draw()">Graphic Draw</button>
```

Ao compor este exemplo, o resultado deverá ser semelhante ao apresentado na Figura 9.3.

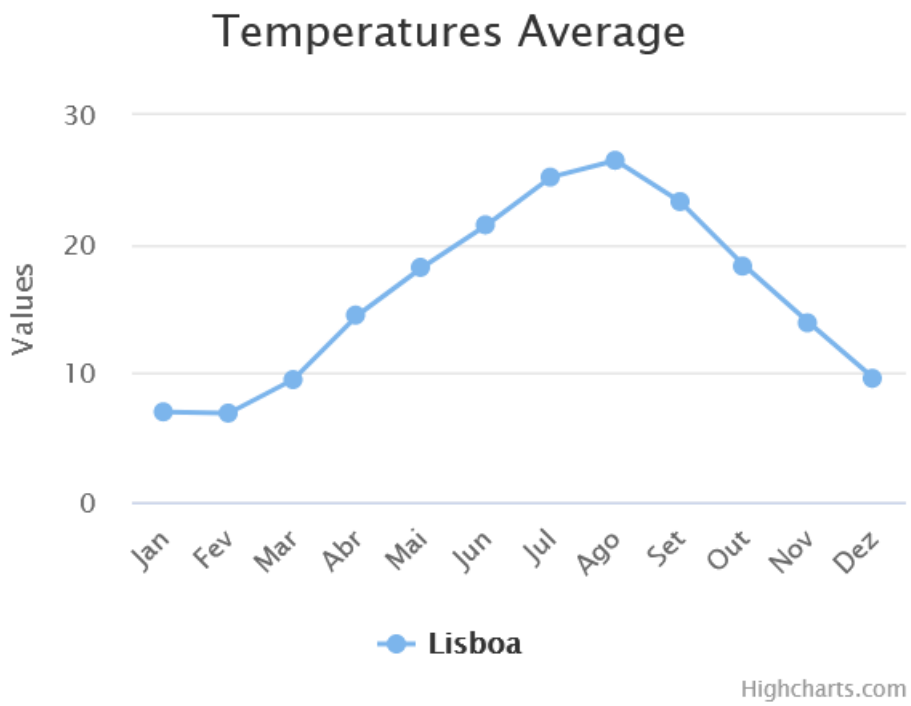

Figura 9.3: Gráfico de linhas

Exercício 9.9

Começe por criar um diretório **js** no diretório **projeto**. Entre no diretório recém criado e crie o ficheiro **graphic.js** com o código apresentado. Verifique o funcionamento da página com o desenho do gráfico de linhas.

Repare que no exemplo anterior apenas é mostrada a série de temperaturas relativas a Lisboa. No entanto é possível ter várias séries simultaneamente. Para isso basta que o campo **series** possua o seguinte formato:

```
series: [
          {
             name: "nome-da-serie-1",
             data: [.... valores ....]
          },
          {
             name: "nome-da-serie-2",
             data: [.... valores ....]
          }
]
```

Exercício 9.10

Adicione uma segunda linha com as temperaturas de Aveiro de forma a comparar os dois locais.

A biblioteca *Highcharts JS* possibilita muitos outros tipos de gráficos, tais como: **pie**, **column**, **scatter**, **bar**, **area** [7]. Frequentemente, modificar a aparência do gráfico apenas requer alterar o seu tipo, definido da seguinte forma:

```
$("#myGraph").highcharts({
  chart: { type: "column" },
  title: { text: "Temperatures Average" },
  ...
```

Assim sendo é muito simples desenvolver uma função com um parâmetro de entrada que especifica o tipo de gráfico que se pretende desenhar, tal como se apresenta no código seguinte.

Depois basta adicionar um botão de tipo seletor na página *Web* e passar o parâmetro à função, usando para esse efeito a função **graphic**. Vamos considerar que por defeito se pretende um gráfico de linhas.

```
var type = "line"; // line graphic by default

function graphic(element) {
    var e = document.getElementById( "selection" );
    type = e.options[e.selectedIndex].value;
}

function draw () {
    $("#myGraph").highcharts({
        chart: { type: type },
        title: { text: "Temperatures Average" },

        // data series
}
```

Exercício 9.11

Altere o ficheiro **graphic.js** para desenhar um gráfico à escolha do utilizador. E na página **index.html** substitua o botão por um botão seletor que permita escolher o tipo de gráfico a desenhar.

9.4 Mapas

Outro componente muito comum em páginas *Web* são os mapas ou outros elementos que apresentam informação geo-referenciada. Os mais populares são provavelmente o *Google Maps* [8] e o *OpenStreetMap* [9].

A utilização destes elementos é análoga à dos gráficos, visto que é necessária a inclusão de um recurso externo, a existência de um elemento onde apresentar o mapa e algum código JS para personalizar o conteúdo.

Apesar de ser possível utilizar serviços como o *Google Maps* diretamente, por vezes é mais vantajoso utilizar bibliotecas que facilitem a integração e utilização. Uma das vantagens é a possibilidade de utilizar mapas de diversos fornecedores sem modificações relevantes na programação. Neste guião recomendamos a utilização da biblioteca *LeaftLet JS* [10].

Para utilizar *LeaftLet JS* é necessário efetuar os seguintes passos:

1. Incluir a definição de estilos e de código do *LeaftLet JS*.

```
<!-- LeaftLet CSS e JavaScript -->
<link rel="stylesheet" href="https://unpkg.com/leaflet@1.0.2/dist/leaflet.css">
<script src="https://unpkg.com/leaflet@1.0.2/dist/leaflet.js"></script>
```

2. Vamos colocar lado a lado um mapa e uma imagem e para isso vamos criar duas colunas na página usando os estilos que estão definidos no ficheiro **style.css** (*columnleft*, *columnright* e *row*).

   Na coluna da esquerda vamos colocar o mapa e incluir um atributo **id="myMap"** para o identificar no código JS. Para aceder rapidamente ao mapa também vamos criar uma ligação interna para este sub-tópico.

```
<h2 style="text-align:center">Topics C - Map and Image</h2>
<div class="row">
    <div class="columnleft" style="text-align: center">
        <h2 id="topicC1">Topic C - Map</h2>
        <div id="myMap" style="width: 550px; height: 450px"></div>
        <!-- JavaScript for map design and other functionalities -->
    </div>
    ...
</div>
```

3. Finalmente é necessário incluir logo a seguir o código JS que cria o mapa. Vamos colocar esse código no ficheiro **map.js** dentro do diretório **js**. Neste caso estamos a usar o código JS diretamente na página em vez de usarmos um botão que invoca a função apropriada.

```
<!-- JavaScript for map design and other functionalities -->
<script src="js/map.js"></script>
```

   Sendo que o conteúdo do ficheiro **map.js** poderá ser o seguinte:

```
var map = new L.Map("myMap", {center: [40.633258,-8.659097],zoom: 15});

var osmUrl="http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png";
var osmAttrib="Map data  OpenStreetMap contributors";
var osm = new L.TileLayer(osmUrl, {attribution: osmAttrib});

map.addLayer(osm);
```

   Neste caso o valor 40.633258 refere-se à latitude, o valor -8.659097 refere-se à longitude e o valor 15 é o nível de *zoom*.

O resultado deverá ser semelhante ao apresentado na Figura 9.4.

Exercício 9.12

Adicione este mapa à sua página e verifique o correto funcionamento da página *Web*.

Figura 9.4: Mapa utilizando *LeaftLet JS*

É possível associar funções a eventos que ocorram no mapa criado. O evento **`click`** é muito útil porque fornece as coordenadas de um ponto selecionado no mapa.[2]

Por exemplo, para mostrar as coordenadas atuais (em cor azul) poderíamos incluir um elemento `<span style="color:blue" id="coordinates"></span>` na página e acrescentar a instrução seguinte:

```
map.on("click", showCoordinates);
```

bem como a definição da respetiva função que deve ser acrescentada ao código JavaScript (ficheiro **`map.js`**):

```
function showCoordinates(e){
  var s = document.getElementById("coordinates");
  s.innerHTML = "Latitude, Longitude = "+e.latlng.lat+", "+e.latlng.lng;
}
```

Uma das utilidades dos mapas é a apresentação de pontos (ou marcadores), indicando a localização de um ponto de interesse. Esta funcionalidade é muito útil para localizar eventos, locais de interesse ou a morada de empresas e serviços.

[2]Lista completa de eventos em `http://leafletjs.com/reference-1.0.0.html#map-event`.

Adicionar um ponto a um mapa requer que se defina as coordenadas do ponto e de seguida que se adicione este ponto ao mapa. No exemplo seguinte é criado um *array* com vários pontos, que são depois adicionados ao mapa.

```
var pontos = [
    L.marker([40.633258, -8.659097]),
    L.marker([40.642729, -8.747899])
  ];

for(let i in pontos) {
  pontos[i].addTo(map);
}
```

Também pode ser adicionado um pequeno texto que será apresentado quando o apontador clicar em cima do ponto. Para isto, é utilizado o método **`bindPopup(msg)`** em que o argumento **`msg`** pode ser qualquer texto ou mesmo código HTML.

```
...
L.marker([40.633258, -8.659097]).bindPopup("LABI@DETI").addTo(map);
...
```

Por vezes é útil ajustar a vista de forma a contemplar todos os pontos adicionados. Aqui é necessário criar um grupo com todos os pontos e depois invocar um método que ajusta a vista de forma automática.

```
...
var grupo = new L.featureGroup(pontos);
map.fitBounds(grupo.getBounds());
```

Exercício 9.13

Adicione marcadores que identifiquem vários locais de interesse para si (morada em Aveiro, morada permanente, onde estuda, etc). Não se esqueça de adicionar textos que descrevam os locais e de ajustar o mapa aos pontos.

Frequentemente é desejável apresentar imagens diferentes dependendo da natureza do local assinalado. Isto é simples pois os elementos **`marker`** podem ser criados com uma lista de opções, onde se inclui o ícone.[3]

[3]Para uma lista completa de opções, consultar `http://leafletjs.com/reference-1.0.0.html#marker`.

```
var iconeUA = L.icon({ iconUrl: "images/ua.png" });
...
L.marker([40.633258, -8.659097], {icon: iconeUA}).bindPopup("LABI@DETI").addTo(map);
...
```

Exercício 9.14

Adicione imagens personalizadas aos seus ícones. Na página `http://mapicons.nicolasmollet.com` encontra muitos e variados ícones. Terá de escolher os ícones e colocá-los junto com a sua página.

Além de marcadores também é possível adicionar polígonos que sinalizam uma área alargada e não um ponto individual. A metodologia é semelhante à criação de marcadores, mas os polígonos são criados através de um *array* de pontos.

A Reitoria de Universidade de Aveiro pode ser indicada através de um polígono, da seguinte forma:

```
var reitoria = L.polygon(
    [ [40.63102, -8.65793], [40.63149, -8.65731],
      [40.63126, -8.65699], [40.63078, -8.65759] ],
    { color: "red" } );
reitoria.addTo(map);
```

Exercício 9.15

Adicione um polígono que delimite o DETI.

Ao polígono que criou, use o método **`bindPopup(msg)`** para mostrar informação do departamento quando se clicar no polígono.

9.5 Imagens

Outro componente muito comum em páginas *Web* são as imagens. Vamos por isso aproveitar o código anteriormente apresentado no guião de JavaScript para processar imagens e colocar uma imagem ao lado do mapa.

Para colocar uma imagem na coluna da direita temos que usar o seguinte código e incluir um atributo **`id="myImage"`** para identificá-la no código JS. Para aceder rapidamente à imagem também vamos criar uma ligação interna para este sub-tópico.

```
<div class="row">
    ...
    <div class="columnright" style="text-align: center">
        <h2 id="topicC2">Topic C - Image</h2>
        <img id="myImage" src="images/img.jpg" alt="Ups!" style="width: 550px; height: 450px;">
        <div>
            <button onclick="decreaseImage(document.getElementById('myImage'))">
                Decrease Image
            </button>
            <button onclick="resetImage(document.getElementById('myImage'))">
                Reset Image
            </button>
        </div>
    </div>
</div>
```

Para apresentar a imagem é necessário colocar uma imagem (no exemplo apresentado está designada por **img.jpg**) no diretório **images**. E colocar o código JS para processar imagens (diminuir e repor) no ficheiro **image.js** dentro do diretório **js**. Também é preciso incluir, na marca **head**, este ficheiro JS com a seguinte linha:

```
<!-- Image Manipulation JavaScript -->
<script src="js/image.js"></script>
```

Exercício 9.16

Utilizando o código apresentado no guião de JavaScript para manipular imagens coloque uma imagem à sua escolha ao lado do mapa e verifique o correto funcionamento da página *Web*.

9.6 Colocação da aplicação *Web* no xcoa

Finalmente deve colocar a aplicação *Web* acabada de desenvolver na sua conta do **xcoa** e verificar se está a funcionar corretamente.

Exercício 9.17

Copie todo o conteúdo do diretório local **projeto** para o seu diretório **public_html** na sua conta **informatica-tXaY** do **xcoa** usando o seguinte comando:

```
scp -r projeto informatica-tXaY@xcoa.av.it.pt:/home/informatica-tXaY/public_html/
```

Verifique que pode aceder à aplicação *Web* através do endereço **http://xcoa.av.it.pt/~informatica-tXaY/projeto**.

9.7 Para aprofundar

Exercício 9.18

Na secção de `Popups` coloque um ícone à sua escolha para representar uma máquina de calcular. Depois utilize o código final da máquina de calcular desenvolvida no guião de JavaScript. Verifique o correto funcionamento da página *Web*.

Exercício 9.19

Em vez de utilizar o tipo `polygon`, se utilizar o `polyline` o resultado será uma linha e não um polígono. Utilize estes dois tipos de objetos para indicar onde mora e qual o trajeto para a Universidade de Aveiro.

Exercício 9.20

Aceda à página do *Twitter Bootstrap* e implemente uma página local com todos os componentes extra que são apresentados.

Exercício 9.21

Aceda à página do *Fuel UX* (`http://exacttarget.github.io/fuelux`) e implemente outros componentes que permitem enriquecer páginas *Web*.

Glossário

CSS	Cascading Style Sheets
HTML	HyperText Markup Language
JS	JavaScript

Referências

[1] Google, *Google Mail*, `http://www.gmail.com`, [Online; acedido em 20 de dezembro de 2022], 2013.

[2] ——, *Google Documents*, `http://doc.google.com`, [Online; acedido em 20 de dezembro de 2022], 2013.

[3] M. Corporation, *Office - Office.com*, `http://office.microsoft.com`, [Online; acedido em 20 de dezembro de 2022], 2013.

[4] W3C. (2001). «Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification», URL: `http://www.w3.org/TR/2011/REC-CSS2-20110607/`.

[5] ECMA International, *Standard ECMA-262 – ECMAScript Language Specification*, Padrão, dez. de 1999. URL: `http://www.ecma-international.org/publications/standards/Ecma-262.htm`.

[6] W3C. (1999). «HTML 4.01 Specification», URL: `http://www.w3.org/TR/1999/REC-html401-19991224/`.

[7] H. JS, *Hicharts Demos*, `http://www.highcharts.com/demo/`, [Online; acedido em 20 de dezembro de 2022], 2013.

[8] Google, *Google Maps*, `http://maps.google.com`, [Online; acedido em 20 de dezembro de 2022], 2013.

[9] OpenStreetMap, *OpenStreetMap*, `http://www.openstreetmap.com`, [Online; acedido em 20 de dezembro de 2022], 2013.

[10] V. Agafonkin, *LeafLet - a JavaScript library for mobile-friendly maps*, `http://leafletjs.com`, [Online; acedido em 20 de dezembro de 2022], 2013.