**ITW – Resumão**

World Wide Web (WWW)

Tim Berners Lee  
- Criador e ainda atual mentor da WWW  
- Uniform Resource Locator (URL),  
- Hypertext Transfer Protocol (HTTP),  
- Hypertext Markup Language (HTML)

Marc Andreessen  
- Criador do Mosaic  
- Fundador da Netscape Communications Corporation

**Hypertext Transfer Protocol (HTTP)**Características gerais:  
Protocolo da camada de aplicação utilizado para transferir documentos de  
hipertexto e seus recursos (imagens, ...) entre máquinas remotas.  
-Modelo de funcionamento baseado em pedido-resposta;  
-Cabeçalho das mensagens é texto puro (não binário);  
-Não orientado a conexões;

**Sintaxe geral de uma URL**

<protocolo>://<servidor>:<porta>/<caminho>/<recurso>?dados

O **HTTPS (Hyper Text Transfer Protocol Secure - protocolo de transferência de hipertexto seguro)** é uma implementação do protocolo HTTP sobre uma camada adicional de segurança que utiliza o protocolo SSL/TLS.

O **SSL (Secure Sockets Layer)** usa um sistema de criptografia que utiliza duas chaves para criptografar os dados, uma chave pública conhecida por todos e  
uma chave privada conhecida apenas pelo destinatário.

**HTML (HyperText Markup Language)**

<marcador>texto</marcador>

Um elemento HTML é formado por um marcador com o nome de etiqueta (tag),  
atributos, valores e filhos (que podem ser outros elementos ou texto).  
Os atributos modificam os valores padrão dos elementos e os valores caracterizam essa mudança.  
Exemplo de um elemento HTML simples (não possui filhos):  
<hr/>  
Exemplo de um elemento HTML com atributos:  
<a href="http://www.ua.pt/">Universidade de Aveiro</a>  
Exemplo de um elemento HTML com filhos e atributos:  
<p>A <a href="http://www.ua.pt/">Universidade de Aveiro</a> é a  
minha Universidade.</p>

Os elementos básicos de um documento html são:  
<html> – define o início de um documento html e indica ao navegador que todo  
conteúdo posterior a este marcador deve ser tratado como uma série de marcadores html.  
Neste marcador pode ter como atributo a língua em que o documentos está escrito. Essa informação é importante para os motores de pesquisa / indexação.  
<head> – define o cabeçalho do documento e possui informações sobre o documento  
que vai ser representado;  
<body> – define o conteúdo principal do documento e é a paret que é exibida no  
navegador. No marcador do corpo podem-se definir atributos comuns a toda a página,  
como cor de fundo, margens, e outras formatações.

Dentro do cabeçalho podemos ter os seguintes elementos:  
<title> – define o título da página; é exibido na barra de título do browser;  
<style type="text/css"> – define a formatação das etiquetas em CSS;  
<script type="text/javascript"> – define a programação de certas funções da página  
escritas na linguagem JavaScript;  
<link> – define ligações da página com outros arquivos, como por exemplo, feeds, CSS,  
scripts, tipos de letra, etc.  
<meta> – define propriedades da página, como, por exemplo, a codificação dos caracteres  
(UTF-8, ISO-8859-1, ...), a descrição da página (autor, keywords,...)

<h1>, <h2>, ... <h6> – define os cabeçalhos e títulos no documento em  
diversos tamanhos;  
<p>...</p> – define um novo parágrafo;  
<br /> – impõe uma quebra de linha num texto;  
<b>...</b>, <i>...</i>, <u>...</u> e <s>...</s> – determina diversas  
formatações de letra: negrito, itálico, sublinhado e riscado, respectivamente.  
<a>...</a> – cria uma hiperligação para um outro local, seja uma página, um  
e-mail ou outro serviço.

Criar listas ordenadas – Marcador <ol>  
Elementos da lista – Marcador <li>

Criar listas não ordenadas – Marcador <ul>  
Elementos da lista – Marcador <li>

Permite a inserção de uma linha de texto  
Atributos:  
Type – define o tipo de input. Para uma linha de texto, "texto";  
Name – define o nome do atributo. Obrigatório. De outra forma essa informação não é  
passada à entidade processadora da informação;  
Value – (pode estar vazio ou omisso). Quando preenchido contém o valor a apresentar pela linha de texto;  
Placeholder – texto que aparece na caixa quando esta está vazia. Normalmente serve de ajuda ao utilizador.  
Exemplos:  
<body>  
<input type="text" name="FirstName" value="Joaquim" placeholder="Nome" />  
<input type="text" name="LastName" placeholder="Sobrenome" />  
</body>

CSS – Cascading Style Sheets  
O que é?  
CSS é o acrónimo de Cascading Style Sheets, ou numa tradução literal para  
português, folhas de estilos encadeados.  
Os estilos CSS permitem fazer a separação entre a estrutura do documento HTML e a sua representação.

Cada uma das camadas pode ser configurada individualmente.  
Content - conteúdo de um elemento de texto ou imagem  
Padding - espaçamento entre o conteúdo e a bordadura  
Border - bordadura à volta do elemento  
Margin - distância entre o border e o limite exterior do elemento

CSS Icons

<i class="fa fa-address-book" aria-hidden="true"></i>

Responsive web design (RWD)

Web design responsivo (RWD) é uma abordagem de desenvolvimento web que cria  
mudanças dinâmicas na aparência de um site, dependendo do tamanho e da  
orientação do dispositivo utilizado na visualização.  
O RWD é uma abordagem para o atual problema de design de websites onde  
coexistem simultaneamente uma grande variedade de dispositivos, com  
características muito distintas e que vão desde os pequenos telefones até aos  
monitores de mesa enormes.

Com esta funcionalidade, o mesmo código HTML é enviado para todos os  
dispositivos que, usando media queries, determinam qual o layout da página  
em cada momento.  
Este processo é dinâmico e a avaliação do tamanho do dispositivo/espaço  
de visualização, é feita permanentemente.  
Com isto, em vez ser necessário criar um site separado e uma base de código  
correspondente para cada dispositivo – que varia desde os ecrãns de grandes  
dimensões, passando pelos desktops, laptops, tablets e telefones de todos os  
tamanhos, é criada apenas uma única base de código que pode oferecer  
suporte a utilizadores com visores de tamanhos diferentes.  
Sempre que o utilizador altera a largura do browser (girando o telemóvel  
ou o tablet) ou redimensionando o browser, num computador normal, a  
representação é recalculada e adaptada.

Bootstrap by Twitter

Pros:  
• Responsiveness, Consistent & Flexible, HTML, CSS, and JS framework, Customizable, Large Community, Excellent Documentation  
Cons:  
• Look alike websites  
• Doesn’t work without an enabled JavaScript  
• Ideal for:  
• A beginner who is new to CSS, as he or she can kick start Bootstrap without any hurdles.  
• A developer with little knowledge of JavaScript who can still use Bootstrap componentes without writing a line in JS.  
• A back-end developer who wants to make some UI changes even if he or she is new to both HTML and CSS.

Uma imagem com mesa

Descrição gerada automaticamente

Linguagens de programação  
O que são?  
Uma “linguagem de programação” é uma linguagem que possui uma sintaxe  
(formato) e uma semântica (significado).  
É usada para expressar uma sequência de ações computacionais que formam  
um “programa” que, por sua vez, é executado num dispositivo.

Numa linguagem compilada, o compilador verifica a sintaxe do código escrito para garantir que esta está de acordo com a semântica adequada e, caso tudo esteja correto, gera um código executável a partir do código fonte escrito pelo programador.  
O código executável não possui o conteúdo do código fonte, portanto programas de  
linguagens compiladas são melhores de distribuir quando o programador não quer que o seu código seja público.  
A versão compilada do programa tipicamente é armazenada, de forma que o programa  
pode ser executado um número indefinido de vezes sem que seja necessária nova  
compilação, o que compensa o tempo gasto na compilação

Se o texto do programa for executado à medida que vai sendo traduzido,linha a linha, comando a comando, como por exemplo em linguagens como o JavaScript, Basic, Python ou Perl, então diz-se que o programa está a ser interpretado e que o mecanismo utilizado para a tradução da linguagem em código máquina é um interpretador.  
Programas interpretados são geralmente mais lentos do que os compilados, mas são  
também geralmente mais flexíveis, já que podem interagir com o ambiente mais  
facilmente – tipicamente, em programação web, basta alterar a página, guardá-la e  
atualizá-la, tudo isto sem ser necessário parar a execução da aplicaçao.

Vantagens e desvantagens do Javascript  
Como é uma linguagem interpretada, é processada aos blocos e interpretada à medida que é necessário converter as diversas estruturas para uma representação capaz de ser executada.  
A vantagem clara desta aproximação é que aparentemente basta executar diretamente o código escrito pelo programador.  
A desvantagem é que muitos erros só são detectados quando o fluxo de execução  
atinge a linha onde o erro está presente – o que pode provocar paragens na execução.

Versatilidade vs segurança

Conforme referido, a linguagem Javascript é bastante poderosa e o facto poder ser executada em qualquer browser de qualquer sistema operativo / dispositivo, permite desenvolver aplicações que podem ser distribuídas de forma muito eficaz – elevada versatilidade.  
No entanto, o código Javascript é sempre enviado ao cliente na sua forma textual, podendo, por isso, ser rapidamente copiado – segurança reduzida.  
Para dificultar a leitura do código, protegendo a autoria do mesmo, e para poupar no espaço ocupado pelo ficheiro, de modo a não prejudicar o carregamento e posterior apresentação da página, este código é muitas vezes “minimizado” (tradução livre de minified).

A sintaxe básica da linguagem Javascript é baseada em instruções, devendo cada uma das instruções ser terminada com o carater ; (ponto-e-vírgula).  
A linguagem Javascript é case-sensitive, o que significa que se deve ter  
cuidado na escrita (Sim ≠ sim ≠ SIM).

Document Object Model (DOM)

O grande potencial da linguagem Javascript quando é executado no browser advém da possibilidade de aceder a qualquer elemento da página HTML.

A caraterística que possibilita esta interação é chamada de Document Object Model (DOM).

No HTML DOM, tudo são nós:  
O documento em si é um nó do tipo document;  
Todos os elementos HTML são nós do tipo element;  
Todos os atributos HTML são nós de tipo attribute;  
O texto dentro dos elementos HTML é um nó do tipo texto ;  
Comentários são nós do tipo comment;

Exemplo:  
Cada objeto (ainda (?) não foi abordado nas disciplinas de programação) possui um  
conjunto de propriedades, métodos e eventos.  
Assim, um elemento <a>...</a> é um objeto; um elemento <p>...</p> também é um  
objeto; ...  
Para aceder programaticamente a este objeto, cujo id="URL\_UA" faremos:  
<a id="URL\_UA" href="http://www.ua.pt">Universidade de Aveiro</a>  
var x = document.getElementById("URL\_UA");

O evento window.onload é executado só “quando a janela (window) estiver  
completamente carregada”.

Validação de formulário com o bootstrap  
É possível validar formulários utilizando o bootstrap.  
Valida elementos do tipo:  
Como é feita a validação?  
O Bootstrap procura pelas classes .invalid-\* e .valid-\* filhas da classe pai .was-  
validated, geralmente aplicada ao <form>.  
Deste modo, qualquer campo obrigatório (atributo required no elemento html) será  
validado quando for carregado o botão de envio do formulário.

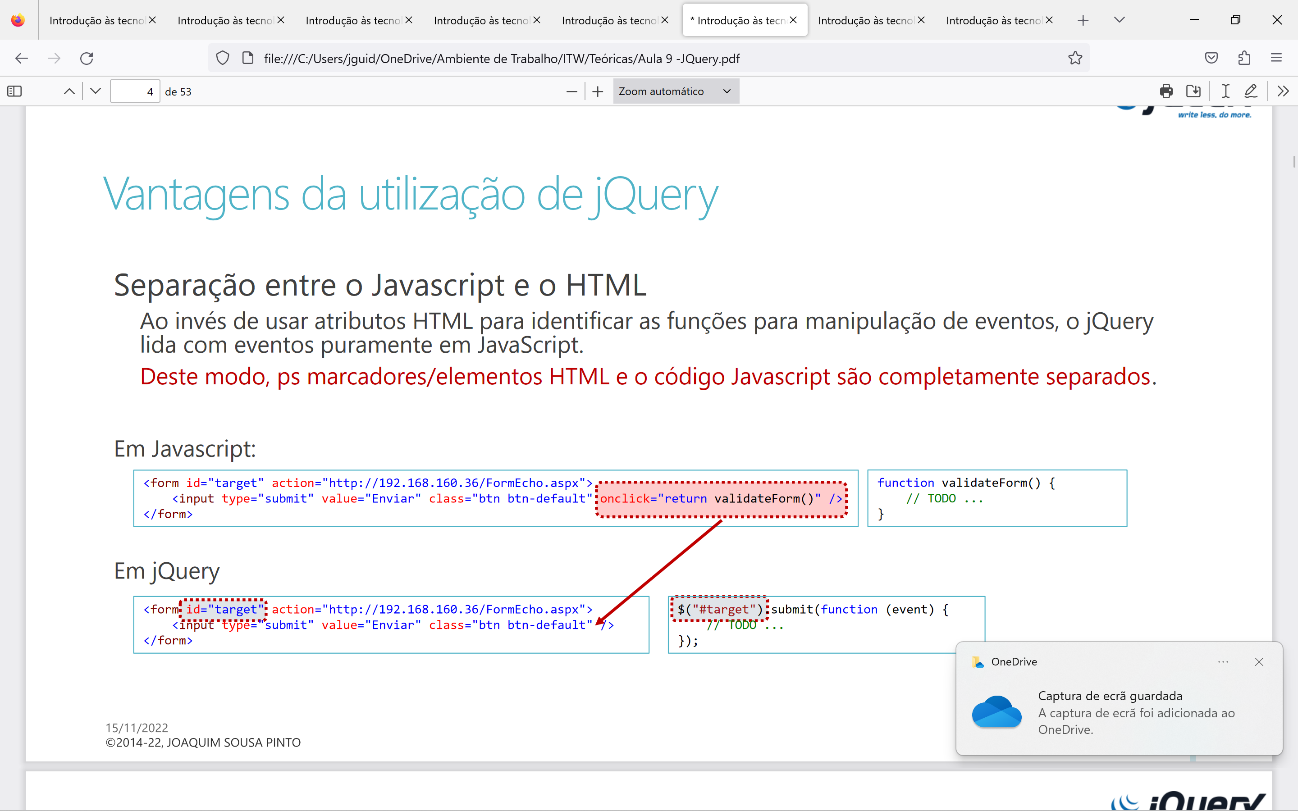
Invalid-feedback

Valid-feedback

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

A sintaxe jQuery foi feita a pensar especialmente na seleção de elemento(s)  
HTML e na execução de alguma ação sobre o(s) mesmo(s).  
A sintaxe básica é: $("selector").action()  
Um sinal $ para definir/aceder à biblioteca jQuery  
Um (seletor) para "consultar/encontrar" elementos HTML no documento  
Uma ação jQuery () a ser executada no(s) elemento(s)

Seletores:  
<form ...></form> ◊ $(“form”)  
id=“myId” ◊ $(“#myId”)  
class=“myClass” ◊ $(“.myClass”)  
<input name=“myName"> ◊ $( "input[name\*='Nam']" )

O seletor jQuery #id usa o atributo id de um elemento HTML para o localizar  
no documento.

O seletor de classe jQuery localiza ocorrências dessa classe específica.  
Para encontrar elementos com uma classe específica, escreva “.”+ nome da  
classe.

Uma imagem com texto

Descrição gerada automaticamente

O evento $(document).ready()

Também pode ser utilizada outra notação equivalente:  
$().ready() ou $("document").ready()

Métodos para manipulação do DOM com jQuery - GET  
Os três métodos para recolha de informação de elementos DOM são:  
text() - Retorna o conteúdo de texto dos elementos selecionados;  
html() - Retorna o conteúdo dos elementos selecionados;  
val() - Retorna o valor dos campos de um formulário (<input>).

O método attr() é usado para obter valores de atributos. O exemplo a seguir  
demonstra como obter o valor do atributo href em um link:

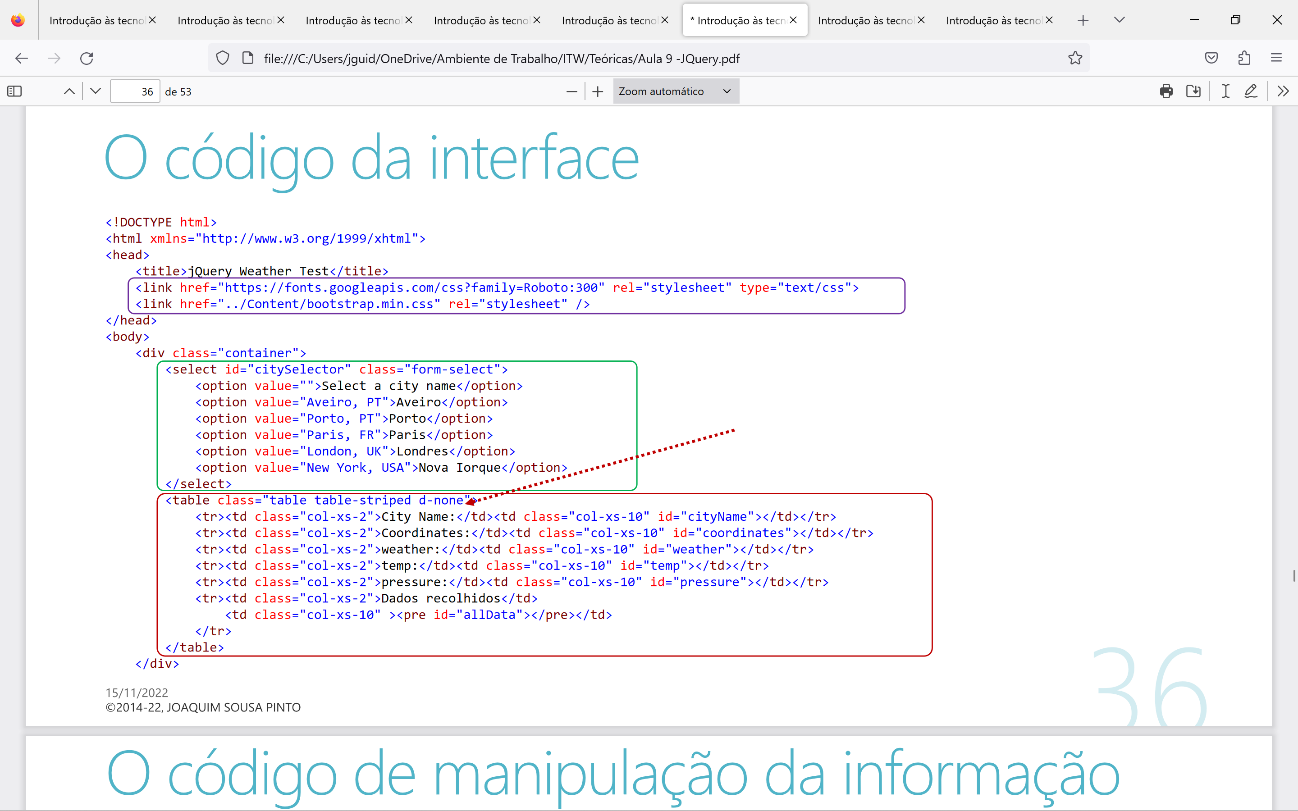
JQuery tem vários métodos para manipulação CSS. Examinaremos os  
seguintes métodos:  
addClass() - Adiciona uma ou mais classes aos elementos selecionados;  
removeClass() - Remove uma ou mais classes dos elementos selecionados;  
toggleClass() - Alterna entre adicionar / remover classes dos elementos  
selecionados;

JSON - JavaScript Object Notation.

JSON é um formato de troca/intercâmbio de dados simples que é  
independente da linguagem de programação utilizada.  
É uma linguagem auto-descritiva e, por isso, fácil de entender.  
Usa a sintaxe JavaScript, mas o formato JSON é somente texto.  
O texto pode ser lido e usado como formato de dados por qualquer  
linguagem de programação.



AJAX (Asynchronous JavaScript and XML)

AJAX é uma funcionalidade que permite que páginas HTML troquem dados  
com um servidor para atualizar apenas partes dessa página mas sem ser  
necessário recarregar toda a página.  
Através da utilização de AJAX, o carregamento de dados é feito em segundo  
plano e o resultado exibido na página da Web, sem recarregar a página.

Uma imagem com texto

Descrição gerada automaticamente

jQueryUI – jQuery User Interface

jQuery UI é uma coleção de widgets de interface gráfica, efeitos visuais  
animados e temas implementados com jQuery, CSS’s e HTML

jQuery UI Widgets  
Acordeão – grupo de contentores organizados na forma de um acordeão  
Autocomplete - caixas que permitem o prenchimento automático com base no que o utilizador digita  
Button - botão com apresentação melhorada.  
Permite que botões rádio e caixas de seleção sejam convertidos em botões  
Datepicker – componente com calendário para recolha de campos com datas  
Dialog - caixas de diálogo colocadas em cima de outros conteúdos  
Menu – componente que permite mostrar e gerir os elementos de um menu  
Progressbar - barras de progresso – animandas, ou não  
Slider – barras de arrastanento totalmente personalizáveis  
Spinner – gere o valor de um número com setas  
Tabs - manipulação interface com tabuladores  
Tooltip - Mostrar uma dica sobre um determnado conteúdo ou operação

A livraria KnockoutJS  
Knockout é uma biblioteca JavaScript que ajuda a criar interfaces de utilizador  
de exibição e edição ricas e responsivas com um modelo de dados subjacente  
limpo.  
Sempre que há seções da interface de utilizador que ncessessitam de  
atualização dinâmica (por exemplo, devido às ações do utilizador ou  
quando uma fonte de dados externa é alterada), o KO, acrónimo do  
Knockout, pode ajudar nessa implementação de forma mais simples e mais  
eficiente que utilizando apenas javascript ou mesmo jQuery.

Principais características:  
Vinculações declarativas  
Associa elementos do DOM a um modelo de dados através de uma sintaxe  
concisa e legível  
Atualização automática da interface com o utilizador  
Quando o estado do modelo de dados é alterado, a interface com o  
utilizador é atualizada automaticamente  
Acompanhamento de dependências  
Implicitamente estabelece cadeias de relações entre os dados do modelo  
de modo a transformá-los e combiná-los  
Templating  
Gera rapidamente intefaces de utilizador sofisticadas como uma função dos  
dados do modelo

Outras características:  
Livre, código aberto (licença MIT)  
JavaScript puro - funciona com qualquer framework web  
Sem dependências  
Pequeno e leve - 67kb minified (@3.5.1 – 11/05/2019)  
Suporta todos os navegadores habituais, mesmo os antigos  
IE 6+, Firefox 3.5+, Chrome, Opera, Safari (desktop / mobile)  
Totalmente documentado  
Há documentos da API, exemplos e tutoriais interativos (até livros!)

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamente

Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

KO bindings

text() – o binding com text() faz com que o elemento DOM associado exiba  
o valor de texto do seu parâmetro.  
Normalmente, esta propriedade é útil com elementos que tradicionalmente  
exibem texto, como por exemplo o <span> ou o <em>, mas tecnicamente  
pode usá-lo com qualquer elemento.  
html() – o binding com html() faz com que o elemento DOM associado exiba  
o html do seu parâmetro.  
Normalmente, isso é útil quando os valores no viewmodel são sequências  
de marcação HTML.

css() – o binding css adiciona ou remove uma ou mais classes CSS ao  
elemento DOM associado.

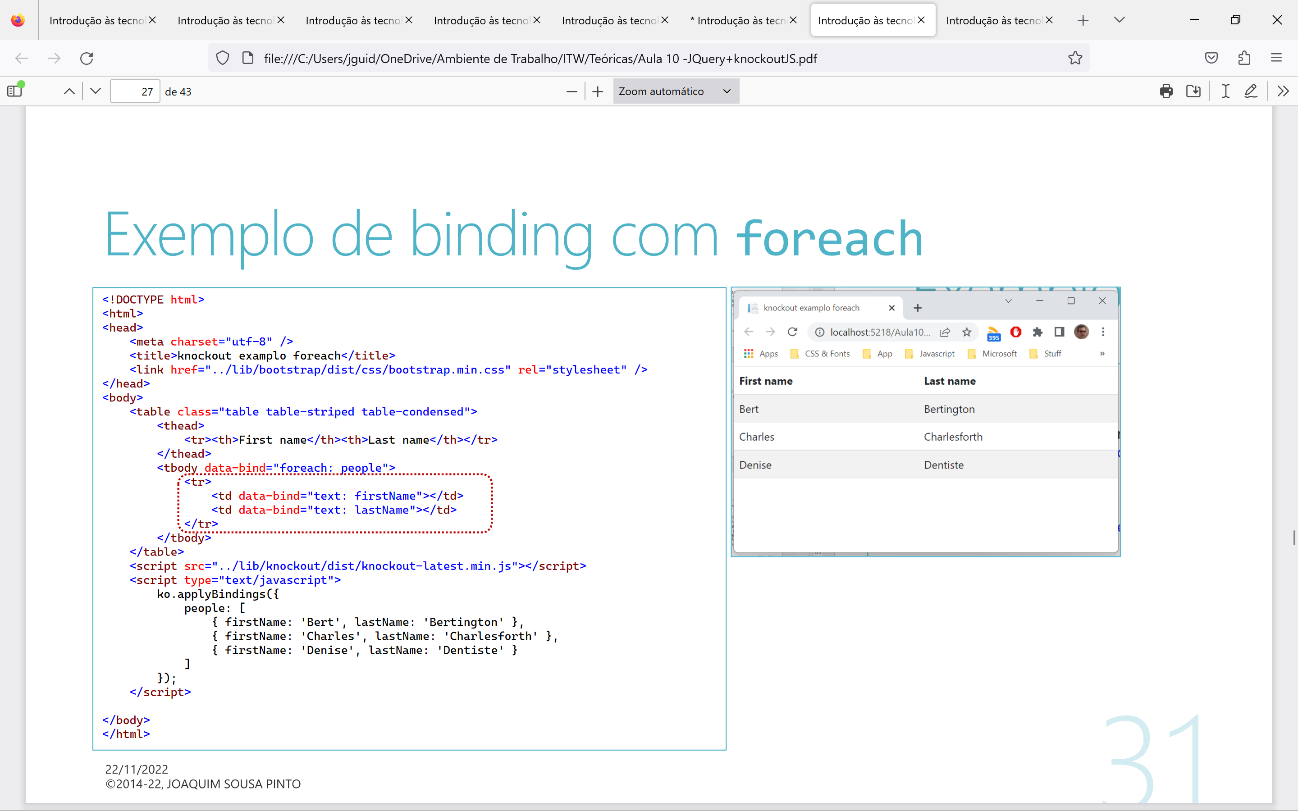
<div data-bind="css: profitStatus">Profit Information</div>

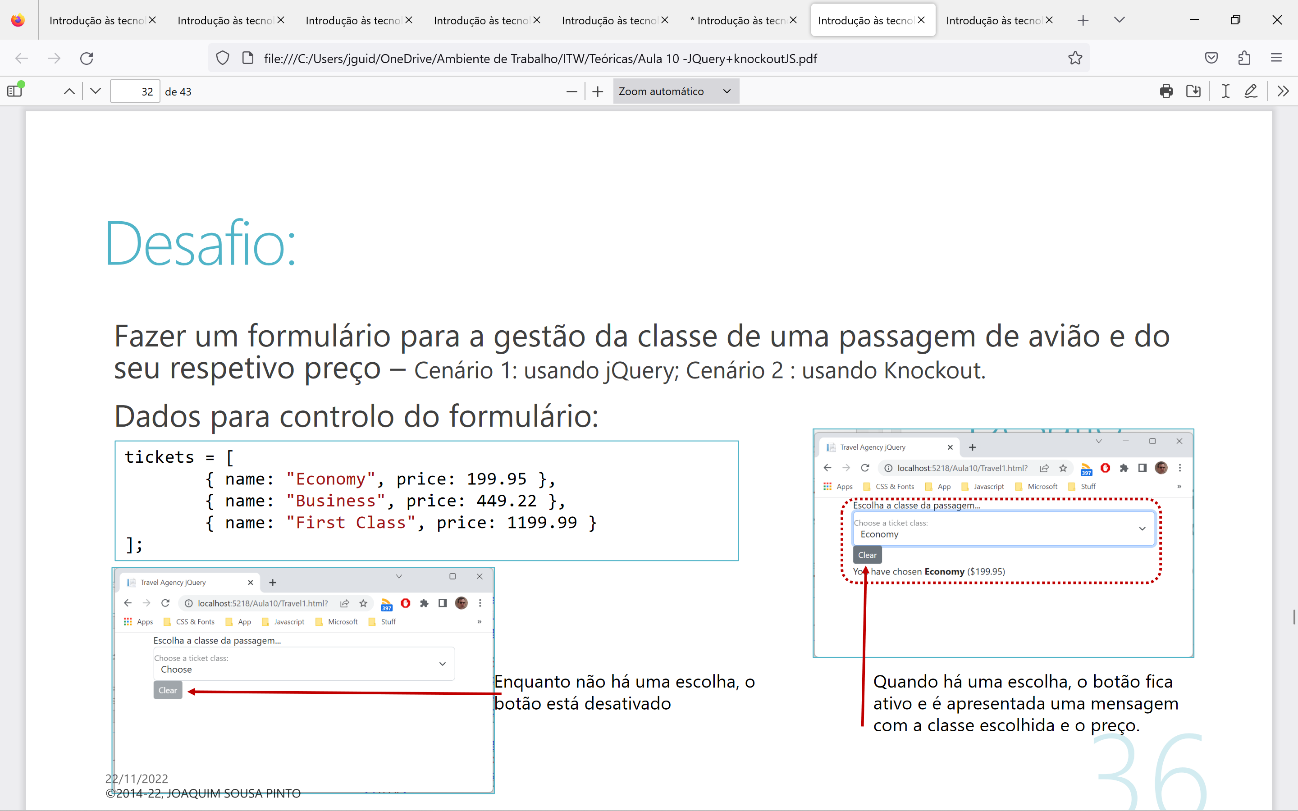
style() – o binding style adiciona ou remove um ou mais valores de estilo ao  
elemento DOM associado.  
<div data-bind="css: profitStatus">Profit Information</div>  
<div data-bind="style: { color: currentProfit() < 0 ? 'red' : 'black' }">Profit Information</div>

attr() – O binding attr fornece uma maneira genérica de definir o valor de  
qualquer atributo para o elemento DOM associado.  
Isso é útil, por exemplo, quando precisa definir o atributo de título de um  
elemento, o src de uma tag img ou o href de um link com base em valores  
no seu viewmodel, com o valor do atributo sendo atualizado  
automaticamente sempre que a propriedade correspondente no viewmodel  
muda.  
<a data-bind="attr: { href: url, title: details }">Relatório</a>  
<script type="text/javascript">  
var viewModel = {  
url: ko.observable("http://somesite.com/yearReport.html"),  
details: ko.observable("Relatório e contas referente ao corrente ano")  
};  
</script>

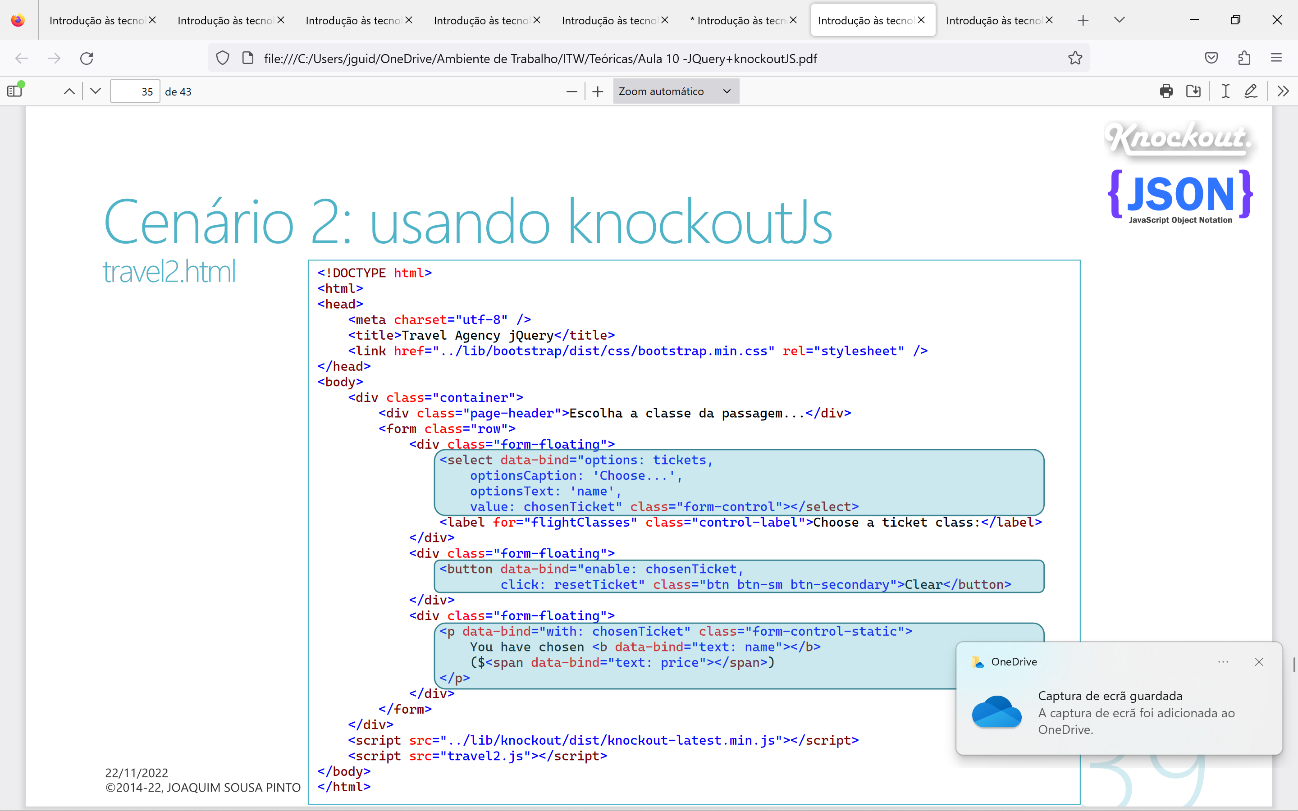
visible() – permite fazer o binding da propriedade visível a um elemento  
Dom que ficará visivel sempre que a variável de controlo do viewmodel  
tomar um valor true.

foreach() – o binding foreach duplica a “lista de entidades” para cada entrada  
numa matriz e vincula cada cópia dessa lista ao item de matriz  
correspondente.  
Isso é especialmente útil para representar listas ou tabelas.  
Assumindo que a matriz é um array de observáveis, sempre que se  
adicionar, remover, alterar ou reordenar as entradas da matriz, a ligação  
atualizará eficientemente a UI mantendo o sincronismo entre elas -  
inserindo ou removendo elementos da lista ou reordenando os elementos  
existentes, sem afetar quaisquer outros elementos DOM.  
Pode encapsular-se qualquer número de bindings foreach e utilizá-los em conjunto com  
outras ligações de controle-fluxo, como if ou with.



Uma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamenteUma imagem com texto

Descrição gerada automaticamente

O que é uma API? Para que serve?  
Uma API (acrónimo do termo em inglês Application programming interface)  
consiste na especificação de um conjunto de procedimentos/funções e  
estruturas de dados estabelecidos por um desenvolvedor de software.  
Através da utilização dessas funções e dessas estruturas de dados cria-se uma  
interface que permite a utilização dessas funcionalidades por aplicativos que  
não pretendem envolver-se nos detalhes da implementação do software, mas  
apenas na utilização dos seus serviços.

A Embed API do serviço Google Maps utiliza um pedido simples, em HTTP, para devolver um  
mapa.  
Este mapa é dinâmico e interativo e pode facilmente ser inserido numa página web,  
atribuindo a URL da API ao atributo “src” de um elemento “iframe”.

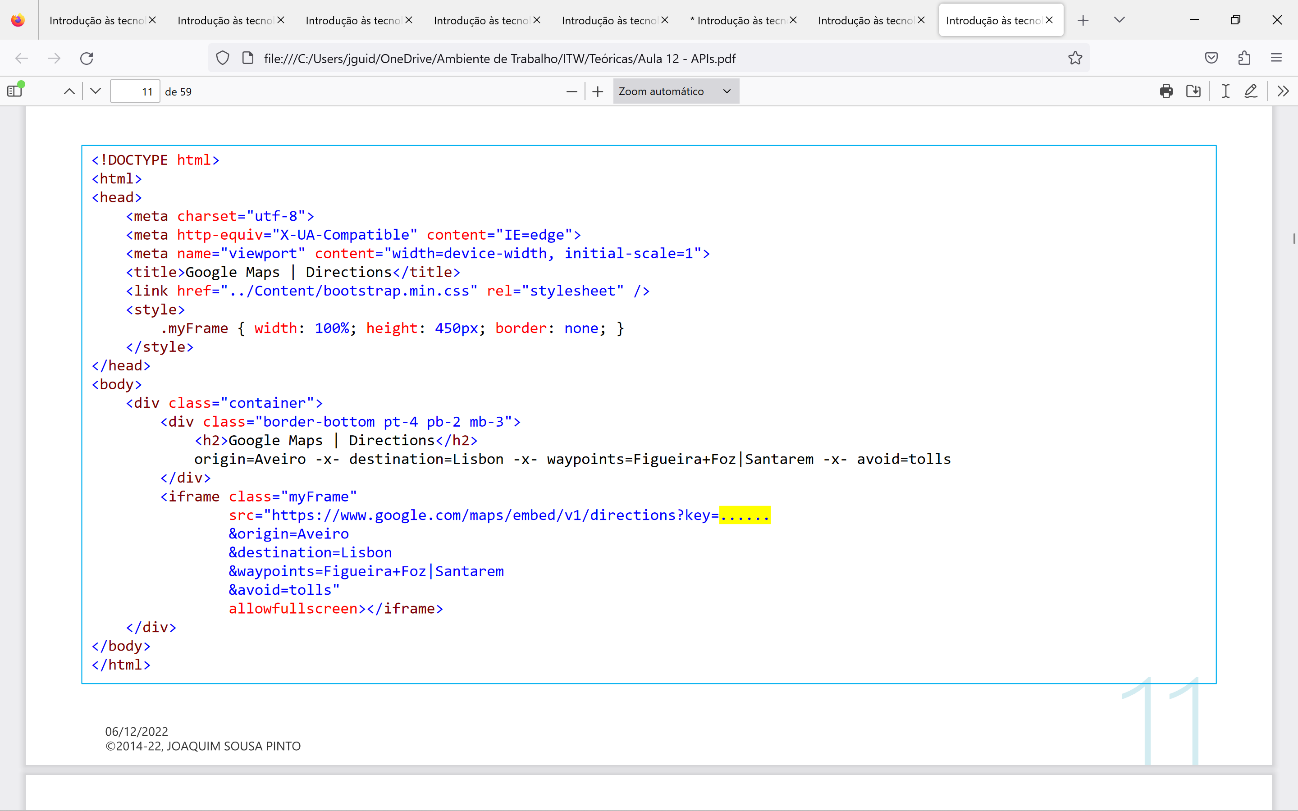
A sintaxe para a URL da API é:  
https://www.google.com/maps/embed/v1/MODO?key=CHAVE&PARÂMETROS  
MODO = “place” | “directions” | “search” | “view”  
CHAVE = chave gratuita de acesso à API  
PARÂMETROS = parâmetros específicos de cada modo ou parâmetros adicionais e opcionais

MODO = “place” – mapa de um lugar  
PARÂMETROS =  
Obrigatórios  
q – especifica o lugar a pesquisar

Uma imagem com texto

Descrição gerada automaticamente

MODO = “directions” – permite a representação de um mapa com um trajeto definido entre um  
conjunto de pontos  
PARÂMETROS =  
Obrigatórios  
origin – origem do trajeto  
destination – destino do trajeto  
Opcionais  
waypoints – pontos de passagem no trajeto  
mode – modo de viajar (driving, walking, bicycling, transit, flying)  
avoid – questões a evitar (tolls, highways)  
units – unidades de medida do trajeto (metric, imperial)



MODO = “search” – permite a representação de um mapa com pesquisa de  
características num lugar (livrarias, farmácias, restaurantes, ...)  
PARAMETROS =  
q – especifica o lugar e as características a pesquisar

Uma imagem com texto

Descrição gerada automaticamente

MODO = “view” – permite desenhar um mapa com centro num par de  
coordenadas GPS  
PARÂMETROS:  
Obrigatório  
center – par de coordenadas GPS do centro do mapa sepearadas por vírgula  
Opcionais  
zoom – nível de ampliação do mapa (0 - 21)  
maptype – tipo de mapa (roadmap, satellite)  
language – linguagem a usar nos elementos da interface  
region – mostra os limites apropriados, com base na sensibilidade geo-políticas (ccTLD – cc  
= código da região)

Uma imagem com texto

Descrição gerada automaticamente

OpenStreetMap (OSM) é um projeto de mapeamento colaborativo para criar  
um mapa livre e editável do mundo.  
É construído por uma comunidade de mapeadores que contribuem e  
mantêm dados sobre estradas, trilhos, cafés, estações ferroviárias e muito  
mais, em todo o mundo.

A biblioteca leafletJS

Leaflet é a biblioteca JavaScript de código aberto líder para mapas interativos em  
computadores desktop e compatíveis com dispositivos móveis.  
Tem cerca de 143 KB de JS possuindo os recursos de mapeamento que a maioria dos  
desenvolvedores precisa.  
Funciona com eficiência em todas as principais plataformas de desktop e móveis, pode ser estendido com vários plug-ins, está bem documentada e possui um código fonte simples e legível.

O que é a Google Chart API?  
A Google Chart API é uma ferramenta que permite criar remotamente um  
gráfico a partir de alguns dados e incorporá-lo numa página da web.  
A API do Google Chart cria uma imagem PNG de um gráfico a partir de  
dados e parâmetros de formatação enviados no pedido HTTP e devolve-o  
para representação no ecran.  
São suportados muitos tipos de gráficos e, fazendo a solicitação através de  
uma tag de imagem, é possível incluir o gráfico numa página da web.

Chart.JS  
A grande vantagem de utilização de uma livraria local reside no fato de os  
dados ficarem sempre do nosso lado;  
Nota: ao enviar os dados para a Google, nunca poderemos afirmar com  
total segurança, que estes dados não são utilizados noutros contextos.  
Outra vantagem da versão local é a possibilidade de implementar gráficos  
mais complexos e que saiam do comum, em que o utilizador configura todos  
os parâmetros.