

# 02

Interfaces Pessoa  
Máquina

## Laboratório 2

HTML, CSS, JS – Tópicos intermédios



# Tópicos a abordar

**Eventos em HTML e JS**

**Apresentação dinâmica de conteúdo**

Ciclos e condições em JS

Posicionamentos em CSS

**Armazenamento de informação**

**Notificações**

**Histórico de navegação**

# 01

## Exercício

# Encontrar uma sala livre na faculdade



# Encontrar uma sala livre na faculdade

## Onde estudar?

Sala: S1

16/02/2018, 18:19:54

8h	PE
10h	EO
12h	
14h	ASA
16h	
18h	

# 02

## Escolher a sala

# Carregar informação das salas

## Objecto JS guardado no ficheiro *rooms.js*

```
var roomdata =  
{ rooms :[  
  { "roomname":"S1",  
    "coords":[7, 51, 129, 183],  
    "classes":[  
      {"time":"8h", "classname":"PE"},  
      {"time":"10h", "classname":"EO"},  
      (...)  
      {"time":"18h", "classname":"none"} ] },  
  (...) ]};
```

# Carregar informação das salas

## Objecto JS guardado no ficheiro *rooms.js*

```
var roomdata =
```

Atributos de uma sala

```
{ rooms :[
```

```
  { "roomname":"S1",
```

```
    "coords":[7, 51, 129, 183],
```

```
    "classes":[
```

```
      {"time":"8h", "classname":"PE"},
```

```
      {"time":"10h", "classname":"EO"},
```

```
      (...)
```

```
      {"time":"18h", "classname":"none"} ] ],
```

```
(...) ]};
```

Atributos de uma aula



# Carregar informação das salas

## Objecto JS guardado no ficheiro *rooms.js*

```
var roomdata =
```

```
{ rooms :[
```

```
  { "roomname":"S1",
```

```
    "coords" [7, 51, 129, 183],
```

```
    "classes":[
```

```
      {"time":"8h", "classname":"PE"},
```

```
      {"time":"10h", "classname":"EO"},
```

```
      (...)
```

```
      {"time":"18h", "classname":"none"} ] },
```

```
  (...)]};
```

Array de salas

Array de coordenadas

Array de aulas

# Carregar informação das salas

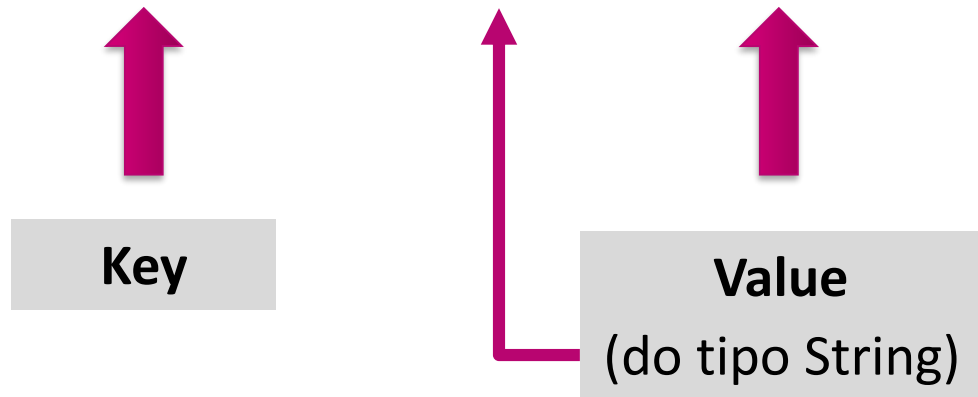
## Ficheiro *lab2.js*

```
function loadAllRoomData(roomData) {  
    // TODO  
}
```

# Carregar informação das salas

## Guardar objecto *roomData* no local storage

```
function loadAllRoomData(roomData) {  
    localStorage.setItem("AllRoomData", JSON.stringify(roomData));  
}
```




# Carregar informação das salas

## Guardar objecto no local storage

```
function loadAllRoomData(roomData) {  
    localStorage.setItem("AllRoomData", JSON.stringify(roomData));  
}
```

## Chamar função ao carregar *index.html*

```
<body onload="loadAllRoomData(getData());">
```



Retorna  
*roomdata.rooms* em  
*rooms.js*

# Obter coordenadas da sala

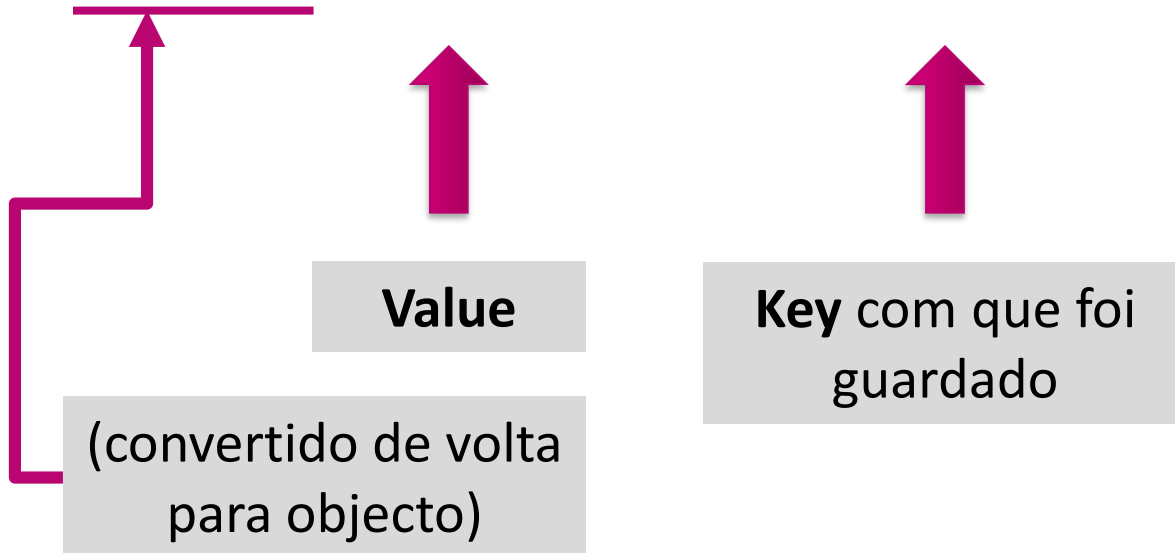
## Aceder às coordenadas guardadas

```
function getRoomCoords(roomName) {  
    // TODO  
}
```

# Obter coordenadas da sala

## Aceder às coordenadas guardadas


```
function getRoomCoords(roomName) {  
    roomData = JSON.parse(localStorage.getItem("AllRoomData"));  
}
```



# Obter coordenadas da sala

## Aceder às coordenadas guardadas

```
function getRoomCoords(roomName) {  
    roomData = JSON.parse(localStorage.getItem("AllRoomData"));  
    for(let i = 0; i < roomData.length; i++) {  
    }  
}
```



Iterar sobre  
todas as salas

Mesma sintaxe  
de C / Java!

# Obter coordenadas da sala

## Aceder às coordenadas guardadas

```
function getRoomCoords(roomName) {  
    roomData = JSON.parse(localStorage.getItem("AllRoomData"));  
    for(let i = 0; i < roomData.length; i++) {  
        if (roomData[i].roomname == roomName) {  
            return roomData[i].coords;  
        }  
    }  
}
```

Retornar as aulas  
da sala escolhida

Mesma sintaxe de  
C / Java!



# Posicionar pin de localização



# Posicionar pin de localização

## Definir posição de origem do pin

```
#target {  
    /* TODO */  
    visibility: hidden;  
}
```

# Posicionar pin de localização

## Definir posição de origem do pin

```
#target {
```

```
    position: absolute;
```

```
    top: 0px;
```

```
    left: 0px;
```

Posição em relação ao  
elemento que o contém

```
    visibility: hidden;
```

```
}
```

# Posicionar pin de localização

## Definir posição de origem do pin

```
#target {  
    position: absolute;  
    top: 0px;  
    left: 0px;  
    z-index: 10;  
    visibility: hidden;  
}
```



Mostrar o pin à  
frente do mapa

# Posicionar pin de localização

## Actualizar posição quando o pin é movido

```
function setLocationPin(x1, y1, x2, y2) {
```

```
    var positionX = x1 + (x2 - x1) / 2.0 - 45;
```

```
    var positionY = y1 + (y2 - y1) / 2.0 - 45;
```

```
    // TODO
```

```
    document.getElementById("target").style.visibility = "visible";
```

```
}
```

Calcular  
“aproximadamente”  
o centro da *<area>*

# Posicionar pin de localização

## Atualizar posição quando o pin é movido

```
function setLocationPin(x1, y1, x2, y2) {
```

```
    var positionX = x1 + (x2 - x1) / 2.0 - 45;
```

```
    var positionY = y1 + (y2 - y1) / 2.0 - 45;
```

```
    // TODO
```

```
    document.getElementById("target").style.visibility = "visible";
```

```
}
```

Mostrar o pin



# Posicionar pin de localização

## Actualizar posição quando o pin é movido


```
function setLocationPin(x1, y1, x2, y2) {  
    var positionX = x1 + (x2 - x1) / 2.0 - 45;  
    var positionY = y1 + (y2 - y1) / 2.0 - 45;  
  
    document.getElementById("target").style.left = positionX;  
    document.getElementById("target").style.top = positionY;  
  
    document.getElementById("target").style.visibility = "visible";  
}
```

Actualizar  
*left e top*

# Mover pin (via *click*)

## Função a chamar

```
function selectRoom(roomName) {  
    localStorage.setItem("roomName", roomName);  
    coords = getRoomCoords(roomName);  
    setLocationPin(coords[0], coords[1], coords[2], coords[3]);  
}
```




Guardar  
a sala



# Mover pin (via *click*)

## Função a chamar

```
function selectRoom(roomName) {  
    localStorage.setItem("roomName", roomName);  
    coords = getRoomCoords(roomName);   
    setLocationPin(coords[0], coords[1], coords[2], coords[3]);  
}
```

Obter coordenadas da sala

# Mover pin (via *click*)

## Função a chamar

```
function selectRoom(roomName) {  
    localStorage.setItem("roomName", roomName);  
    coords = getRoomCoords(roomName);  
    setLocationPin(coords[0], coords[1], coords[2], coords[3]);  
}
```



Situar o pin nas  
coordenadas

# Mover pin (via *click*)

Para cada <area>:

```
<area target="" alt="S1" title="S1" coords="7,51,129,183"  
  shape="rect">
```

(...)

```
<area target="" alt="Lab4" title="Lab4" coords="442,486,563,616"  
  shape="rect">
```

# Mover pin (via *click*)

Para cada <area>:

```
<area target="" alt="S1" title="S1" coords="7,51,129,183"  
  shape="rect" onclick="selectRoom('S1')" >
```

(...)


```
<area target="" alt="Lab4" title="Lab4" coords="442,486,563,616"  
  shape="rect" onclick="selectRoom('Lab4')" >
```

# Mover pin (via *drag and drop*)

## Função a chamar quando o pin é *dragged*

Elemento a  
mover

Informação do  
evento



```
function drag(target, e) {  
    e.dataTransfer.setData('pin_id', target.id);  
}
```

# Mover pin (via *drag and drop*)

Função a chamar quando o pin é *dragged*

```
function drag(target, e) {  
    e.dataTransfer.setData('pin_id', target.id);  
}
```



Informação do pin

# Mover pin (via *drag and drop*)

## Função a chamar quando o pin é *dropped*



Destino do pin

```
function drop(target, e, roomName) {  
    pin_id = e.dataTransfer.getData('pin_id')  
    pin = document.getElementById(pin_id);  
    target.appendChild(pin);  
    selectRoom(roomName);  
}
```

# Mover pin (via *drag and drop*)

## Função a chamar quando o pin é *dropped*

```
function drop(target, e, roomName) {  
    pin_id = e.dataTransfer.getData('pin_id')  
    pin = document.getElementById(pin_id);  
    target.appendChild(pin);  
    selectRoom(roomName);  
}
```

Obter o pin





# Mover pin (via *drag and drop*)

## Função a chamar quando o pin é *dropped*

```
function drop(target, e, roomName) {  
    pin_id = e.dataTransfer.getData('pin_id')  
    pin = document.getElementById(pin_id);  
    target.appendChild(pin);  
    selectRoom(roomName);  
}
```



Mover o pin para a  
<area> de destino

# Mover pin (via *drag and drop*)

## Função a chamar quando o pin é *dropped*

```
function drop(target, e, roomName) {  
    pin_id = e.dataTransfer.getData('pin_id')  
    pin = document.getElementById(pin_id);  
    target.appendChild(pin);  
    selectRoom(roomName);  
}
```



Seleccionar sala

# Mover pin (via *drag and drop*)

## No pin de localização

```

```

# Mover pin (via *drag and drop*)

## No pin de localização

```

```



Tornar o pin  
arrastável



Comportamento  
quando se agarra o pin

# Mover pin (via *drag and drop*)

## Para cada <area>

```
<area target="" alt="S1" title="S1" coords="7,51,129,183"  
  shape="rect" onclick="selectRoom('S1')">
```

# Mover pin (via *drag and drop*)

## Para cada <area>

```
<area target="" alt="S1" title="S1" coords="7,51,129,183"  
  shape="rect" onclick="selectRoom('S1')"
```

```
  ondrop="drop(this, event, 'S1');"
```



Comportamento  
quando se larga o pin

```
  ondragenter="return false"
```

```
  ondragover="return false">
```



Ignorar estes eventos

# Mudar para a página do horário

## Botão para aceder à página do horário

```
<div id="footer">
```

```
  <!-- TODO -->
```

```
</div>
```

# Mudar para a página do horário

## Botão para aceder à página do horário

```
<div id="footer">
```

```
  <button type="button">Ver horário</button>
```

```
</div>
```



# Mudar para a página do horário

## Botão para aceder à página do horário

```
<div id="footer">
```

```
  <a href="schedule.html">
```



Link para a página  
do horário

```
    <button type="button">Ver horário</button>
```

```
  </a>
```

```
</div>
```

# Mudar para a página do horário

## Onde estudar?

Escolhe uma sala:



[Ver horário](#)



# Mudar para a página do horário

Mas...

## Onde estudar?

Sala:

19/02/2018, 23:59:26

**Falta carregar o horário!**

03

**Mostrar horário**

# Mostrar horário da sala

## Aceder ao horário guardado

```
function getRoomSchedule(roomName) {  
    // TODO  
}
```

**EXERCÍCIO!**

(pista: já fizeram algo parecido hoje...)

# Mostrar horário da sala

## Aceder ao horário guardado

```
function getRoomSchedule(roomName) {  
    roomData = JSON.parse(localStorage.getItem("AllRoomData"));  
    for(let i = 0; i < roomData.length; i++) {  
        if (roomData[i].roomname == roomName) {  
            return roomData[i].classes;  
        }  
    }  
}
```

Tal como o *getRoomCoords()* !

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {  
    var roomName = localStorage.getItem("roomName");  
  
    // TODO  
  
    currentSchedule = getRoomSchedule(roomName);  
  
    hourcol = document.getElementById("hours");  
  
    classcol = document.getElementById("classes");  
  
    // TODO  
  
}
```

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {  
  
    var roomName = localStorage.getItem("roomName");  
  
    document.getElementById("currentRoom").innerHTML +=  
    roomName;
```



Mostrar nome da sala

```
(...)
```

```
}
```



# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {
```

```
(...)
```

```
currentSchedule = getRoomSchedule(roomName);
```

```
hourcol = document.getElementById("hours");
```

```
classcol = document.getElementById("classes");
```

```
// TODO
```

```
}
```



Obter o horário da sala

<div> das horas  
e das aulas

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {
```

```
(...)
```

```
for (let i = 0; i < currentSchedule.length; i++) {
```

```
    hourcol.innerHTML += '<div class="timeslot">' + currentSchedule[i].time +  
'</div>';
```

```
    if (currentSchedule[i].classname == "none")
```

```
        classcol.innerHTML += '<div class="emptyslot"> &nbsp; </div>';
```

```
    else classcol.innerHTML += '<div class="classslot">' +  
currentSchedule[i].classname + '</div>';
```

```
}}
```

Hora de início



# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {
```

```
(...)
```

```
for (let i = 0; i < currentSchedule.length; i++) {
```

```
    hourcol.innerHTML += '<div class="timeslot">' + currentSchedule[i].time +  
'</div>';
```

```
    if (currentSchedule[i].classname == "none")
```

Slots vazios

```
        classcol.innerHTML += '<div class="emptyslot"> &nbsp; </div>';
```


```
    else classcol.innerHTML += '<div class="classslot">' +  
currentSchedule[i].classname + '</div>';
```

```
}}
```

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
function displaySchedule() {  
    (...)  
    for (let i = 0; i < currentSchedule.length; i++) {  
        hourcol.innerHTML += '<div class="timeslot">' + currentSchedule[i].time +  
        '</div>';  
        if (currentSchedule[i].classname == "none")  
            classcol.innerHTML += '<div class="emptyslot"> &nbsp; </div>';  
        else classcol.innerHTML += '<div class="classslot">' +  
            currentSchedule[i].classname + '</div>';  
    }  
}
```



Slots com aulas

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
<body onload="displayDate();">
```

# Mostrar horário da sala

## Mostrar o horário em *schedule.html*

```
<body onload="displayDate(); displaySchedule();">
```

# Mostrar horário da sala

Agora sim...

## Onde estudar?

Sala: S1

16/02/2018, 18:19:54

8h	PE
10h	EO
12h	
14h	ASA
16h	
18h	

04

**Subscrever sala**



# Notificação de sala livre

## Subscrever uma sala para obter notificação

```
function notifyON() {  
    // TODO  
}
```

# Notificação de sala livre

## Subscriver uma sala para obter notificação

```
function notifyON() {
```

```
    var timer = setInterval(showAlert, 5000);
```

```
}
```



Função a  
chamar



Tempo  
(em ms)

# Notificação de sala livre

## Subscriver uma sala para obter notificação

```
function notifyON() {  
    var timer = setInterval(showAlert, 5000);  
}
```

## Mostrar o alerta com informação da sala

```
function showAlert() {  
  
}
```

# Notificação de sala livre

## Subscriver uma sala para obter notificação

```
function notifyON() {  
    var timer = setInterval(showAlert, 5000);  
}
```

## Mostrar o alerta com informação da sala

```
function showAlert() {  
    document.getElementById("alertsucccess").style.display = "block";  
}
```

# Mostrar alerta

## Mostrar alerta em *schedule.html*

```
<div id="footer">
```

```
    <button type="button"> Alertar quando sala estiver livre</button>
```

```
</div>
```

# Mostrar alerta

## Mostrar alerta em *schedule.html*

```
<div id="footer">
```

```
  <button type="button" onclick="notifyON()"> Alertar quando sala  
  estiver livre</button>
```

```
</div>
```

# Mostrar alerta

A sala que subscreveste já se encontra livre.

X

## Onde estudar?

Sala: S1

20/02/2018, 00:04:34

8h	PE
10h	EO
12h	
14h	ASA
16h	
18h	

# Mostrar alerta

## Fechar o alerta

```
function closeAlert() {  
    var div1 = document.getElementById("alertsucces");  
    div1.style.opacity = "0";  
    setTimeout(function(){ div1.style.display = "none"; }, 600);  
}
```



# Mostrar alerta

## Fechar o alerta

```
function closeAlert() {
```

```
    var div1 = document.getElementById("alertsucces");
```

```
    div1.style.opacity = "0";
```



Alerta ficará transparente

```
    setTimeout(function(){ div1.style.display = "none"; }, 600);
```

```
}
```

# Mostrar alerta

## Fechar o alerta

```
function closeAlert() {  
    var div1 = document.getElementById("alertsucces");  
    div1.style.opacity = "0";  
    setTimeout(function(){ div1.style.display = "none"; }, 600);  
}
```



Tempo para o *div*  
desaparecer

# Mostrar alerta

## Fechar o alerta

```
function closeAlert() {  
    var div1 = document.getElementById("alertsucces");  
    div1.style.opacity = "0";  
    setTimeout(function(){ div1.style.display = "none"; }, 600);  
}
```

## Fechar alerta em *schedule.html*

```
<span class="closebtn">x</span>
```

# Mostrar alerta

## Fechar o alerta

```
function closeAlert() {  
    var div1 = document.getElementById("alertsucces");  
    div1.style.opacity = "0";  
    setTimeout(function(){ div1.style.display = "none"; }, 600);  
}
```

## Fechar alerta em *schedule.html*

```
<span class="closebtn" onclick="closeAlert()">x</span>
```

05

TPC

# Mostrar data e hora actual

## Função que retorna data e hora

```
function getToday() {  
    var date = new Date();  
  
    //return date.toString();  
  
    //return date.toUTCString();  
  
    //return date.toDateString();  
  
    return date.toLocaleString();  
  
}
```

# Voltar à página anterior

## Implementar a função para voltar atrás

```
function goBack() {  
    // TODO  
}
```

# Para referência - HTML

<http://www.w3schools.com/html/>

<https://dev.w3.org/html5/html-author/>

<https://www.w3.org/TR/html-markup/>



# Para referência - CSS

<http://www.w3schools.com/css/>

<http://www.htmlhelp.com/reference/css/>

<https://jigsaw.w3.org/css-validator/>

# Para referência - JS

<http://www.w3schools.com/js/>

[https://developer.mozilla.org/en-US/Learn/Getting\\_started\\_with\\_the\\_web/JavaScript\\_basics](https://developer.mozilla.org/en-US/Learn/Getting_started_with_the_web/JavaScript_basics)

05

**PRÓXIMA AULA**

# Mais HTML

## Continuar a aprofundar conhecimentos de HTML5

Quanto mais experimentarem, mais vão aprender na aula!

Se não agora, quando? Quando o trabalho das outras cadeiras apertar?

# Questionários

## Fazer questionário para Análise Utilizadores e Tarefas

Tão próximo quanto possível do final

## Preparação

Ler Cap. 10 do livro

Rever as 11 perguntas (livro)