

# Sistemas Multimédia

---

## Tecnologia e Prática da Web II

### 01

## JavaScript: Introdução

- Javascript
  - Breve história
  - Modelo de programação
  - JavaScript versus Java
  - Sintaxe
  - Usar JavaScript
  - Operadores
    - Atribuição
    - Comparação
    - Lógicos
    - String
  - Janelas de dialogo
  - Condições
  - Ciclos
  - Funções

# HTML5

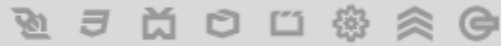
---

- HTML5  $\approx$  HTML 5 + CSS 3 + JavaScript
- HTML5 é conjunto de ferramentas para:
  - Marcação (HTML 5)
  - Apresentação (CSS 3)
  - Interacção (DOM, Ajax, APIs, js)

# HTML 5 – Funcionalidades

---



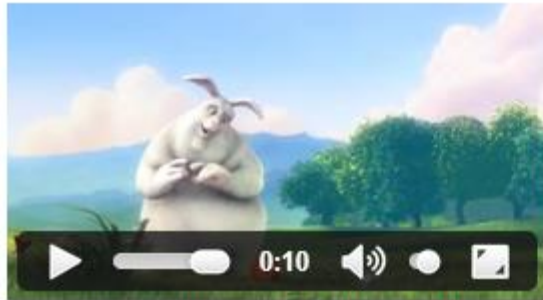


## HTML



## HTML5

- New Elements
- New Attributes
- Full CSS3 Support
- Video and Audio
- 2D/3D Graphics
- Local Storage
- Local SQL Database
- Web Applications

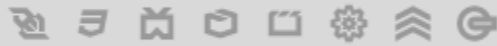


Video courtesy of [Big Buck Bunny](#)

## HTML5 Multimedia

With HTML5, playing video and audio is easier than ever.

- HTML5 [<video>](#)
- HTML5 [<audio>](#)



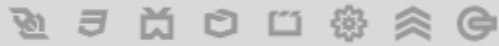
A 3D graphic of a white cube with a blue face, rendered on a blue gradient background.

## HTML5 Graphics

With HTML5, drawing graphics is easier than ever:

- Using the [<canvas>](#) element
- Using inline [SVG](#)
- Using [CSS3 2D/3D](#)

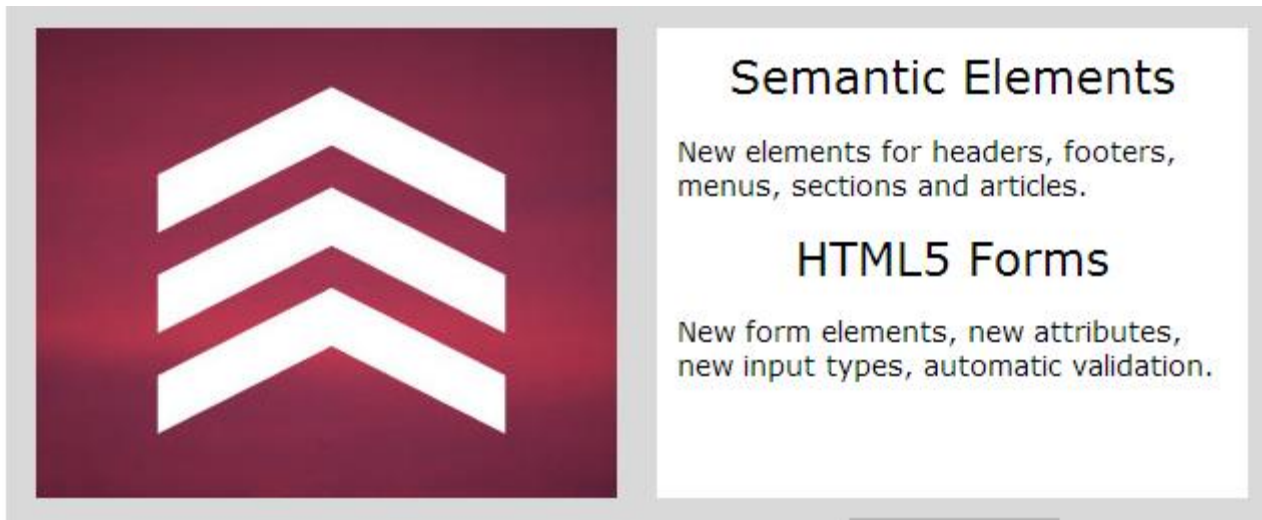
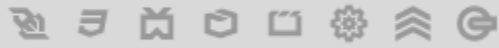


A presentation slide titled 'HTML5 Applications'. The slide has a light gray border. On the left side, there is a large white 'G' logo on a blue background with white clouds. On the right side, the title 'HTML5 Applications' is at the top. Below the title, a paragraph states 'With HTML5, web application development is easier than ever.' followed by a bulleted list of features.

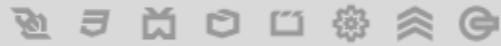
## HTML5 Applications

With HTML5, web application development is easier than ever.

- Local data storage
- Local file access
- Local SQL database
- Application cache
- Javascript workers
- XMLHttpRequest 2



# HTML 5 – Funcionalidades



## HTML5 uses CSS3

- New Selectors
- New Properties
- Animations
- 2D/3D Transformations
- Rounded Corners
- Shadow Effects
- Downloadable Fonts

Read more in our [CSS3 tutorial](#).

# HTML 5 – Funcionalidades

## Exemplos

---

HTML5  $\approx$  HTML 5 + CSS 3 + JavaScript

<http://slides.html5rocks.com>

# HTML 5 – Funcionalidades: Exemplos

## Offline / Storage

Expect the unexpected



## JS Web Storage

```
// use localStorage for persistent storage
// use sessionStorage for per tab storage
saveButton.addEventListener('click', function () {
  window.localStorage.setItem('value', area.value);
  window.localStorage.setItem('timestamp', (new Date()).getTime());
}, false);
textarea.value = window.localStorage.getItem('value');
```

Save text value on the client side (crash-safe)

Type your text here...

Save

## JS Web SQL Database

```
var db = window.openDatabase("DBName", "1.0", "description", 5*1024*1024); //5MB
db.transaction(function(tx) {
    tx.executeSql("SELECT * FROM test", [], successCallback, errorCallback);
});
```

See the generated database: Developer > Developer Tools > Storage

## JS IndexedDB

```
var idbRequest = window.indexedDB.open('Database Name');
idbRequest.onsuccess = function(event) {
    var db = event.srcElement.result;
    var transaction = db.transaction([], IDBTransaction.READ_ONLY);
    var curRequest = transaction.objectStore('ObjectStore Name').openCursor();
    curRequest.onsuccess = ...;
};
```




## JS Application Cache

```
<html manifest="cache.appcache">
```

```
window.applicationCache.addEventListener('updateready', function(e) {  
  if (window.applicationCache.status == window.applicationCache.UPDATEREADY) {  
    window.applicationCache.swapCache();  
    if (confirm('A new version of this site is available. Load it?')) {  
      window.location.reload();  
    }  
  }  
}, false);
```

cache.appcache:

```
CACHE MANIFEST  
# version 1.0.0  
  
CACHE:  
/html5/src/logic.js  
/html5/src/style.css  
/html5/src/background.png  
  
NETWORK:  
*
```

Turn off your internet connection and refresh this page! 

## Realtime / Communication

Stay connected



# HTML 5 – Funcionalidades: Exemplos

## JS Web Workers

main.js:

```
var worker = new Worker('task.js');  
worker.onmessage = function(event) { alert(event.data); };  
worker.postMessage('data');
```

task.js:

```
self.onmessage = function(event) {  
  // Do some work.  
  self.postMessage("recv'd: " + event.data);  
};
```



Find route with Workers

Find route without Workers

Try dragging the map while the complex route is being calculated (you will only be able to do that with Workers!)

## JS WebSocket

```
var socket = new WebSocket('ws://html5rocks.websocket.org/echo');
socket.onopen = function(event) {
    socket.send('Hello, WebSocket');
};
socket.onmessage = function(event) { alert(event.data); }
socket.onclose = function(event) { alert('closed'); }
```

Full-duplex, bi-directional communication over the Web: Both the server and client can send data at any time, or even at the same time. Only the data itself is sent, without the overhead of HTTP headers, dramatically reducing bandwidth.

Use the echo demo below to test a WebSocket connection from your browser. Both the message you send and the response you receive travel over the same WebSocket connection.

**Location:**

☐ Use secure WebSocket (TLS/SSL)

**Message:**

**Output:**

Demo powered by **KAAZING**



## JS Notifications

```
if (window.webkitNotifications.checkPermission() == 0) {  
    // you can pass any url as a parameter  
    window.webkitNotifications.createNotification(tweet.picture, tweet.title,  
        tweet.text).show();  
} else {  
    window.webkitNotifications.requestPermission();  
}
```

Set notification permissions for this page

Note: Use this button if you also want to *reset* the permissions

Enter your twitter user name to show your last tweet as a notification

Show tweet notifications

## File / Hardware Access

Deeper integration with the Operating System



## JS Native Drag & Drop

```
document.addEventListener('dragstart', function(event) {  
    event.dataTransfer.setData('text', 'Customized text');  
    event.dataTransfer.effectAllowed = 'copy';  
}, false);
```



Select text and drag  
(original text will be  
dropped)

Select text and drag  
(dragged text data  
will be altered from  
original)

Drop Area

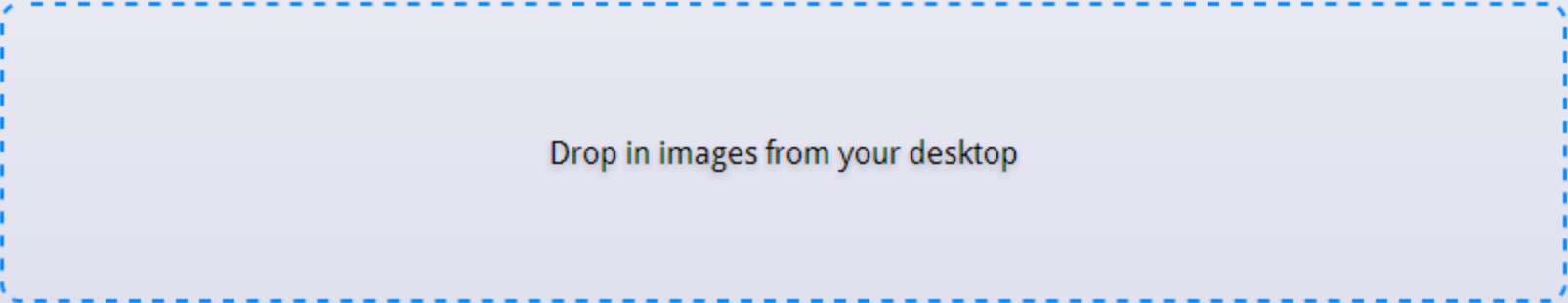
Source Data



## JS Desktop Drag-In (File API)

Drag files in from the desktop:

```
document.querySelector('#dropzone').addEventListener('drop', function(e) {  
  var reader = new FileReader();  
  reader.onload = function(evt) {  
    document.querySelector('img').src = evt.target.result;  
  };  
  
  reader.readAsDataURL(e.dataTransfer.files[0]);  
}, false);
```



Drop in images from your desktop



## JS Desktop Drag-Out

Drag files out onto the desktop:

```
<a href="src/star.mp3" draggable="true" class="dragout" data-downloadurl="MIMETYPE:FILENAME:ABSOLUTE_URI_TO_FILE">download</a>
```

```
var files = document.querySelectorAll('.dragout');
for (var i = 0, file; file = files[i]; ++i) {
  file.addEventListener('dragstart', function(e) {
    e.dataTransfer.setData('DownloadURL', this.dataset.downloadurl);
  }, false);
}
```

Drag each of these files onto your desktop: .pdf file .mp3 file

( this feature is only available in Google Chrome )

## JS FileSystem APIs

Asynchronously write a file to a sandboxed file system using JavaScript:

```
window.requestFileSystem(window.TEMPORARY, 1024 * 1024, function(fs) {  
    // fs.root is a DirectoryEntry object.  
    fs.root.getFile('log.txt', {create: true}, function(fileEntry) {  
        fileEntry.createWriter(function(writer) { // writer is a FileWriter object.  
            writer.onwrite = function(e) { ... };  
            writer.onerror = function(e) { ... };  
  
            var bb = new BlobBuilder();  
            bb.append('Hello World!');  
  
            writer.write(bb.getBlob('text/plain'));  
        }, opt_errorHandler);  
    }, opt_errorHandler);  
}, opt_errorHandler);
```

( The FileSystem API is currently only implemented in Google Chrome 9+ )

# HTML 5 – Funcionalidades: Exemplos

## JS Geolocation

```
if (navigator.geolocation) {  
  navigator.geolocation.getCurrentPosition(function(position) {  
    var latLng = new google.maps.LatLng(  
      position.coords.latitude, position.coords.longitude);  
    var marker = new google.maps.Marker({position: latLng, map: map});  
    map.setCenter(latLng);  
  }, errorHandler);  
}
```

Show Position



## JS Device Orientation

```
window.addEventListener('deviceorientation', function(event) {  
  var a = event.alpha;  
  var b = event.beta;  
  var g = event.gamma;  
}, false);
```



## HTML Speech Input

```
<input type="text" x-webkit-speech />
```



## Semantics & Markup

More meaningful elements



## HTML

## Better semantic tags

```
<body>
  <header>
    <hgroup>
      <h1>Page title</h1>
      <h2>Page subtitle</h2>
    </hgroup>
  </header>
```

```
<nav>
  <ul>
    Navigation...
  </ul>
</nav>
```

```
<section>
  <article>
    <header>
      <h1>Title</h1>
    </header>
    <section>
      Content...
    </section>
  </article>
  <article>
    <header>
```

```
    <h1>Title</h1>
  </header>
  <section>
    Content...
  </section>
</article>
</section>
```

```
<aside>
  Top links...
</aside>
```

```
<figure>
  
  <figcaption>Chart 1.1</figcaption>
</figure>
```

```
<footer>
  Copyright @
  <time datetime="2010-11-08">2010</time>.
</footer>
</body>
```



## HTML Markup for applications

```
<input list="cars"/>
<datalist id="cars">
  <option value="BMW"/>
  <option value="Ford"/>
  <option value="Volvo"/>
</datalist>
```

You should see an autocomplete menu as you type


```
<menu>
  <command type="command" disabled label="Publish" />
</menu>
```

```
<details>
  <summary>HTML 5</summary>
  This slide deck teaches you everything you need to know about HTML 5.
</details>
▶ HTML 5
```

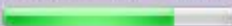
```
<meter min="0" max="100" low="40" high="90" optimum="100" value="91">A+</meter>
```

Your score is: 

```
<progress>working...</progress>
```

Download is: 

```
<progress value="75" max="100">3/4 complete</progress>
```

Goal is: 



## HTML Descriptive link relations

```
<link rel="alternate" type="application/rss+xml" href="http://myblog.com/feed"/>
<link rel="icon" href="/favicon.ico"/>
<link rel="pingback" href="http://myblog.com/xmlrpc.php"/>
<link rel="prefetch" href="http://myblog.com/main.php"/>
...

<a rel="archives" href="http://myblog.com/archives">old posts</a>
<a rel="external" href="http://notmysite.com">tutorial</a>
<a rel="license" href="http://www.apache.org/licenses/LICENSE-2.0">license</a>
<a rel="nofollow" href="http://notmysite.com/sample">wannabe</a>
<a rel="tag" href="http://myblog.com/category/games">games posts</a>
...
```

# HTML 5 – Funcionalidades: Exemplos

HTML

## Microdata

```
<div itemscope itemtype="http://example.org/band">
  <p>My name is <span itemprop="name">Neil</span>.</p>
  <p>My band is called <span itemprop="band">Four Parts Water</span>.</p>
  <p>I am <span itemprop="nationality">British</span>.</p>
</div>
```

The screenshot shows the 'Rich Snippets Testing Tool' interface within Google Webmaster Tools. On the left, there's a sidebar with 'Rich Snippets' and 'Help with:' links for 'Documentation' and 'Tips & Tricks'. The main area is titled 'Rich Snippets Testing Tool' with a 'Beta' tag. It explains that Rich Snippets enhance Google search results by marking up web pages with Microformats, RDFa, or Microdata. A section 'Test your website' prompts the user to enter a URL, with an example 'http://www.urbanspoon.com/r/6/765421/restaurant/Pizza-My-Heart-Santa-Cruz' and a 'Preview' button. Below this, a 'Google search preview' shows a search result for 'Pizza My Heart - Santa Cruz | Urbanspoon' with a 5-star rating, 10 reviews, and a price range of 'Under \$10 per entree'. It also shows an excerpt from the page and a link to the full page. A note states that there is no guarantee that a Rich Snippet will be shown for this page on actual search results. At the bottom, a section 'Extracted Rich Snippet data from the page' displays structured data in a JSON-LD format, including 'hreview-aggregate', 'item hcard', 'fn = Pizza My Heart', 'org', and 'organization name = Pizza My Heart'.

Rich Snippets Testing Tool at <http://www.google.com/webmasters/tools/richsnippet>

## HTML ARIA attributes

```
<ul id="tree1"  
  role="tree"  
  tabindex="0"  
  aria-labelledby="label_1">  
<li role="treeitem" tabindex="-1" aria-expanded="true">Fruits</li>  
<li role="group">  
  <ul>  
    <li role="treeitem" tabindex="-1">Oranges</li>  
    <li role="treeitem" tabindex="-1">Pineapples</li>  
    ...  
  </ul>  
</li>  
</ul>
```

## HTML New form types

```
<style>
  [required] {
    border-color: #88a;
    -webkit-box-shadow: 0 0 3px rgba(0, 0, 255, .5);
  }
  :invalid {
    border-color: #e88;
    -webkit-box-shadow: 0 0 5px rgba(255, 0, 0, .8);
  }
</style>
```

```
<input type="text" required />
<input type="email" value="some@email.com" />
<input type="date" min="2010-08-14" max="2011-08-14" value="2010-08-14"/>
<input type="range" min="0" max="50" value="10" />
<input type="search" results="10" placeholder="Search..." />
<input type="tel" placeholder="(555) 555-5555"
       pattern="^\(?[0-9]{3}\)?[-\s][0-9]{3}[-\s][0-9]{4}.*?$" />
<input type="color" placeholder="e.g. #bbbbbb" />
<input type="number" step="1" min="-5" max="10" value="0" />
```

The image shows a visual representation of the HTML5 form controls defined in the code. It includes a text input with a red border (required), an email input with the value 'some@email.com', a date input showing '14-08-2010', a range input with a slider between 0 and 50, a search input with a magnifying glass icon and placeholder 'Search...', a tel input with a placeholder '(555) 555-5555' and a pattern attribute, a color input showing a black color swatch, and a number input showing the value '0' with up/down arrows.

# HTML 5 – Funcionalidades: Exemplos

## HTML Form field types on mobile

type="text"



Android Device

type="number"



Android Device

type="email"



iPhone Device

type="tel"



iPhone Device

## Graphics / Multimedia

2D & 3D





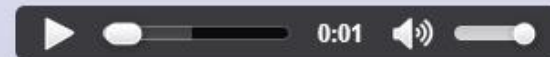
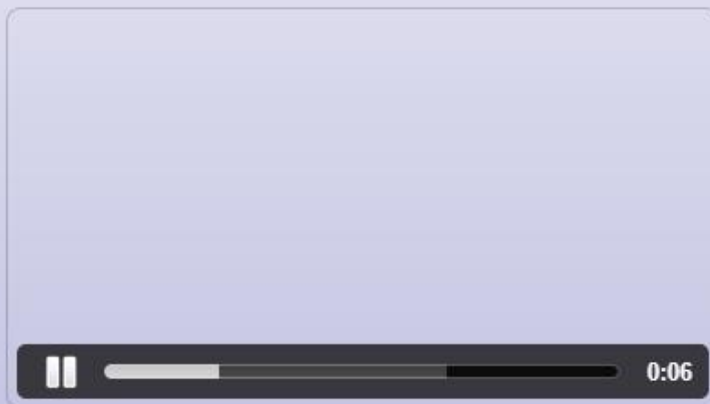
# HTML 5 – Funcionalidades: Exemplos

## HTML JS Audio + Video

```
<audio id="audio" src="sound.mp3" controls></audio>  
document.getElementById("audio").muted = false;
```

```
<video id="video" src="movie.webm" autoplay controls></video>  
document.getElementById("video").play();
```

Add CSS reflection to video



Play Pause Mute

Play Pause Mute



# HTML 5 – Funcionalidades: Exemplos

HTML

JS

## Canvas 2D

```
<canvas id="canvas" width="838" height="220"></canvas>

<script>
  var canvasContext = document.getElementById("canvas").getContext("2d");
  canvasContext.fillRect(250, 25, 150, 100);

  canvasContext.beginPath();
  canvasContext.arc(450, 110, 100, Math.PI * 1/2, Math.PI * 3/2);
  canvasContext.lineWidth = 15;
  canvasContext.lineCap = 'round';
  canvasContext.strokeStyle = 'rgba(255, 127, 0, 0.5)';
  canvasContext.stroke();
</script>
```



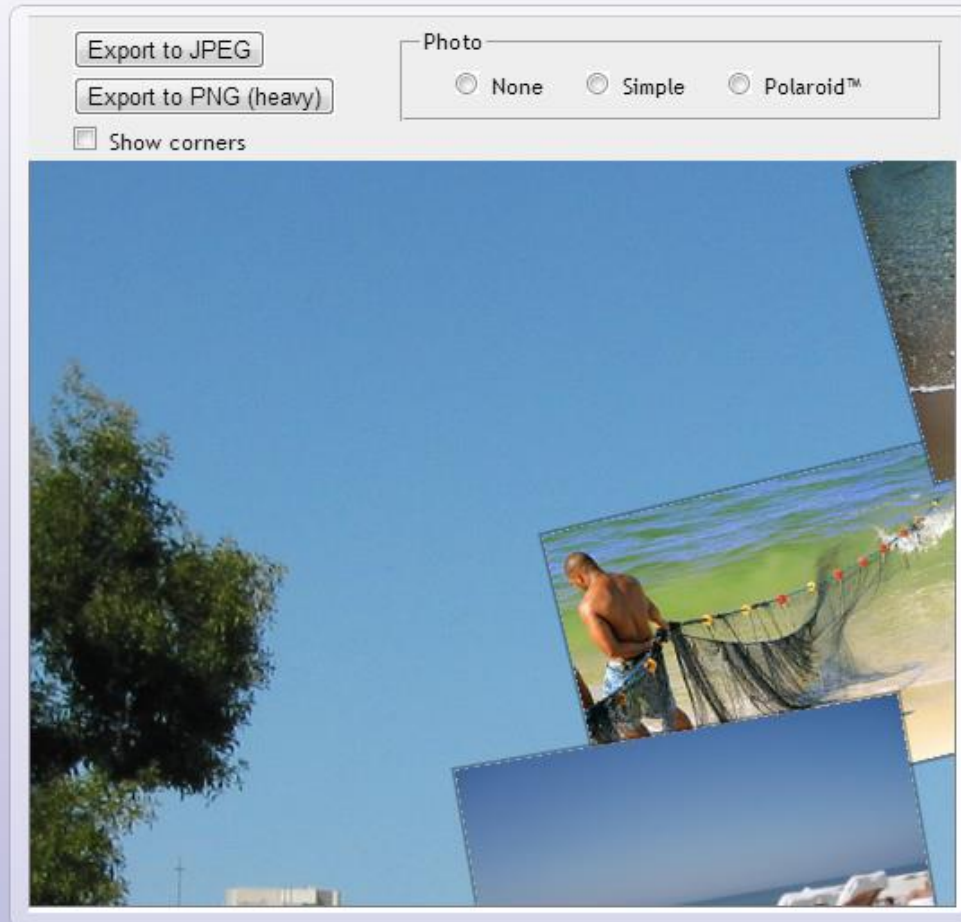


# HTML 5 – Funcionalidades: Exemplos

HTML

JS

## Canvas example



# HTML 5 – Funcionalidades: Exemplos

HTML

JS

## Canvas 3D (WebGL)

```
<canvas id="canvas" width="838" height="220"></canvas>

<script>
  var gl = document.getElementById("canvas").getContext("experimental-webgl");
  gl.viewport(0, 0, canvas.width, canvas.height);
  ...
</script>
```



[Original demo](#) by Jetro Lauha WebGL port by Kenneth Waters



## HTML Inline SVG

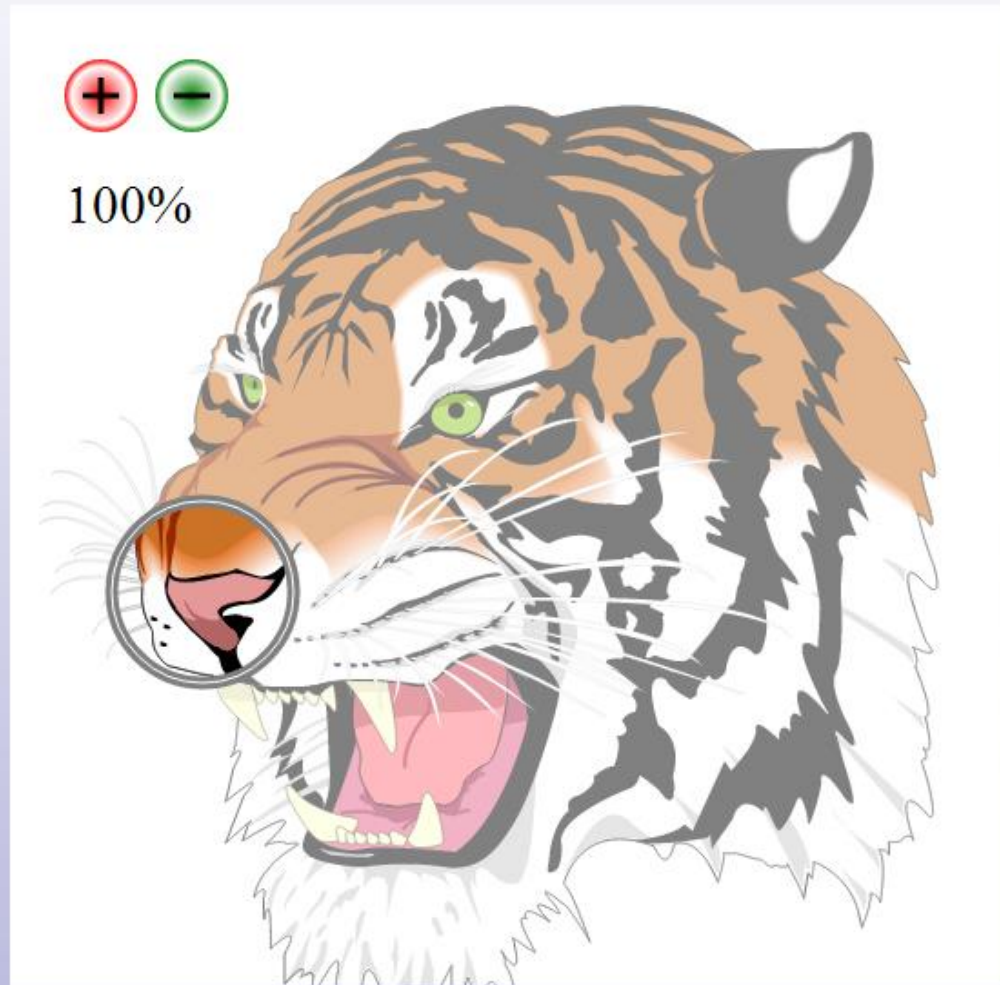
```
<html>
  <svg>
    <circle id="myCircle" class="important" cx="50%" cy="50%" r="100"
      fill="url(#myGradient)"
      onmousedown="alert('hello');"/>
  </svg>
</html>
```



# HTML 5 – Funcionalidades: Exemplos

HTML

## SVG example



# HTML 5 – Funcionalidades: Exemplos



## CSS CSS Selectors

### Selectors

```
.row:nth-child(even) {  
  background: #dde;  
}  
.row:nth-child(odd) {  
  background: white;  
}
```

Row 1

Row 2

Row 3

Row 4

### Image-like display

```
div {  
  display: inline-block;  
}
```

### Specific attributes

```
input[type="text"] {  
  background: #eee;  
}
```

### Negation

```
:not(.box) {  
  color: #00c;  
}  
:not(span) {  
  display: block;  
}
```

### More specific targeting

```
h2:first-child { ... }  
  
div.text > div { ... }  
h2 + header { ... }
```

CSS2 selectors and CSS3 selectors introduced some of the ones demonstrated here.



## CSS Webfonts

```
@font-face {  
  font-family: 'LeagueGothic';  
  src: url(LeagueGothic.otf);  
}  
  
@font-face {  
  font-family: 'Droid Sans';  
  src: url(Droid_Sans.ttf);  
}  
  
header {  
  font-family: 'LeagueGothic';  
}
```

**LeagueGothic font with no image replacement**



## CSS Text wrapping

```
div {  
  text-overflow: ellipsis;  
}
```

A long cold winter delayed the blossoming of the millions of cherry,


A long cold winter delayed the blossoming of the millions of cherry, apric

A long cold winter delayed the blossoming of the millions of cherry, ...

Play with the slider on this and further pages!



## CSS Columns

```
-webkit-column-count: 2;   
-webkit-column-rule: 1px solid #bbb;  
-webkit-column-gap: 2em;
```

In March 1936, an unusual confluence of forces occurred in Santa Clara County.

A long cold winter delayed the blossoming of the millions of cherry, apricot, peach, and prune plum trees covering hundreds of square miles of the Valley floor. Then, unlike many years, the rains that followed were light and too early to knock the blossoms from their branches.

Instead, by the billions, they all burst open at once. Seemingly overnight, the ocean of green that was the Valley turned into a low, soft, dizzyingly perfumed cloud of pink and white. Uncounted bees and yellow jackets, newly born, raced out of their hives and holes, overwhelmed by this impossible banquet.

Then came the wind.

It roared off the Pacific Ocean, through the nearly uninhabited passes of the Santa Cruz Mountains and then, flattening out, poured down into the great alluvial plains of the Valley. A tidal bore of warm air, it tore along the columns of trees, ripped the blossoms apart and carried them off in a fluttering flood of petals like foam rolling up a beach.

This perfumed blizzard hit Stevens Creek Boulevard, a two-lane road with a streetcar line down its center, that was the main road in the West Valley. It froze traffic, as drivers found themselves lost in a soft, muted whiteout. Only the streetcar, its path predetermined, passed on...

## CSS Text stroke

```
div {  
  -webkit-text-fill-color: black;  
  -webkit-text-stroke-color: red;  
  -webkit-text-stroke-width: 1.75px;  
}
```

Text stroke example

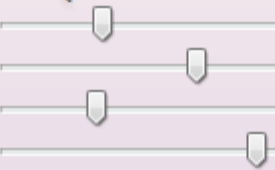
## CSS Opacity

```
color: rgba(255, 0, 0, 0.75);  
background: rgba(0, 0, 255, 0.75);
```



## CSS Hue/saturation/luminance color

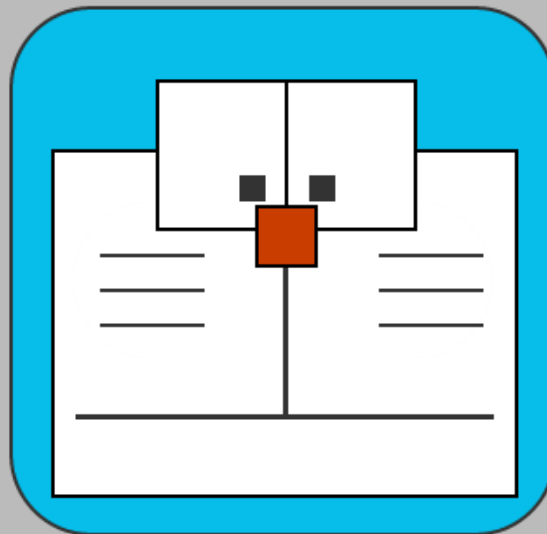
```
color: hsla(  
  128  
  75%,  
  33%,  
  1.00);
```



# HSL example

## CSS Rounded corners

```
face: border-radius: 47px;
left eye: border-radius: 0px;
right eye: border-radius: 0px;
base white: border-radius: 0px;
mouth: border-radius: 0px;
nose: border-radius: 0px;
left black eye: border-radius: 0px;
right black eye: border-radius: 0px;
```



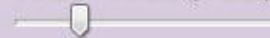


## CSS Gradients

```
background: -webkit-gradient(linear, left top, left bottom,  
                             from(#00abee), to(white),  
                             color-stop(0.5, white), color-stop(0.5, #66cc00))
```



```
background: -webkit-gradient(radial, 430 50, 0, 430 50, 170, from(red), to(#000))
```



## CSS Shadows

### text-shadow:

rgba(64, 64, 64, 0.5)

0px 

6px 


0px; 

### box-shadow:

rgba(0, 0, 128, 0.25)




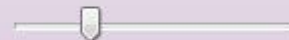
0px 

0px 

0px; 

Shadows example

## CSS Instant Web 2.0 (just add sliders)

```
text-shadow: rgba(0, 0, 0, 0.5) 0 4px 4px;   
background:  
  -webkit-gradient(linear, left top, left bottom,   
    from(rgba(200, 200, 240, 0.47)), to(rgba(255, 255, 255, 0.47)))  
border-radius: 16px;   
-webkit-box-reflect: below 10px  
  -webkit-gradient(linear, left top, left bottom,   
    from(transparent), to(rgba(255, 255, 255, 0.24)));
```

Web 2.0 Logo Creatr

MeP 5'0 Logo Creatr



## CSS Background enhancements

### Background sizing

```
#logo {  
  background: url(logo.gif) center center no-repeat;  
  background-size:  
    ;  
}
```

Resize me! »



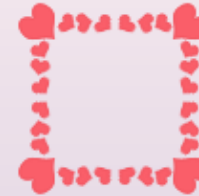
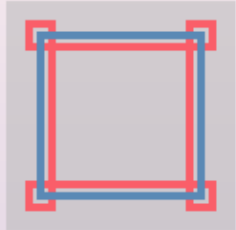
### Multiple backgrounds

```
div {  
  background: url(src/zippy-plus.png) 10px center no-repeat,  
             url(src/gray_lines_bg.png) 0 center repeat-x;  
}
```

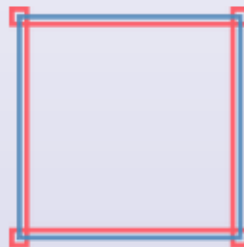
+ Test

# HTML 5 – Funcionalidades: Exemplos

## CSS Border image



`-webkit-border-image: url(border_image_1.png) 20% stretch stretch;`  
`border-width: 13px;`



## CSS Flexible Box Model

```
.box {  
  display: -webkit-box;  
  -webkit-box-orient: horizontal;  
}  
.box .one, .box .two {  
  -webkit-box-flex: 1;  
}  
.box .three {  
  -webkit-box-flex: 3;  
}
```

Box one

Box two

Box three

## CSS Flexible Box Model

```
.box {  
  display: -webkit-box;  
  -webkit-box-pack: center;  
  -webkit-box-align: center;  
}
```

change the dropdowns and  
resize this <textarea>

## CSS Transitions



```
#box.left {  
  margin-left: 0;  
}  
#box.right {  
  margin-left: 1000px;  
}
```

```
document.getElementById('box').className = 'left';
```

Left

```
document.getElementById('box').className = 'right';
```

Right



```
#box {  
  -webkit-transition: margin-left 1s ease-in-out;  
}
```

```
document.getElementById('box').className = 'left';
```

Left

```
document.getElementById('box').className = 'right';
```

Right

## CSS Transforms

Hover over me:

```
-webkit-transform: rotateY(45deg);  
-webkit-transform: scaleX(25deg);  
-webkit-transform: translate3d(0, 0, 90deg);  
-webkit-transform: perspective(500px)
```

```
#threed-example {  
  -webkit-transform: rotateZ(5deg);  
  
  -webkit-transition: -webkit-transform 2s ease-in-out;  
}  
#threed-example:hover {  
  -webkit-transform: rotateZ(-5deg);  
}
```

Now press **3** !

## CSS Animations

```
@-webkit-keyframes pulse {  
  from {  
    opacity: 0.0;  
    font-size: 100%;  
  }  
  to {  
    opacity: 1.0;  
    font-size: 200%;  
  }  
}  
  
div {  
  -webkit-animation-name: pulse;  
  -webkit-animation-duration: 2s;  
  -webkit-animation-iteration-count: infinite;  
  -webkit-animation-timing-function: ease-in-out;  
  -webkit-animation-direction: alternate;  
}
```

\*Please make a better use of it. We don't want a new blink tag ;)

Pulse!

# Javascript

---



- ❑ O JavaScript é uma linguagem do tipo *script* mas com a particularidade de ser baseada em objectos.
  - Linguagem interpretada
- ❑ Este script foi desenvolvido pela Netscape e posteriormente adaptado para permitir às páginas de WEB uma maior interactividade bem como ter novas funcionalidades.
  - A Netscape lançou em 1995 com o nome de LiveScript
  - Posteriormente a Netscape aliou-se a sun
  - Depois a Microsoft lançou a JSript – uma cópia do JavaScript
  - Para evitar uma guerra de tecnologias, a Netscape decidiu que o melhor seria torna-la uma norma
    - ❑ Desta forma, em 1997 submeteu a linguagem JavaScript 1.1 ao ECMA (*European Computer Manufacturers Association*).
      - Norma ECMA-262
    - ❑ Por isso muitas vezes aparece designada como ECMAScript
- ❑ <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

□ <https://www.tiobe.com/tiobe-index/>

■ Março 2021

Mar 2021	Mar 2020	Change	Programming Language	Ratings	Change
1	2	▲	C	15.33%	-1.00%
2	1	▼	Java	10.45%	-7.33%
3	3		Python	10.31%	+0.20%
4	4		C++	6.52%	-0.27%
5	5		C#	4.97%	-0.35%
6	6		Visual Basic	4.85%	-0.40%
7	7		JavaScript	2.11%	+0.06%
8	8		PHP	2.07%	+0.05%
9	12	▲	Assembly language	1.97%	+0.72%
10	9	▼	SQL	1.87%	+0.03%
11	10	▼	Go	1.31%	+0.03%
12	18	▲▲	Classic Visual Basic	1.26%	+0.49%
13	11	▼	R	1.25%	-0.01%
14	20	▲▲	Delphi/Object Pascal	1.20%	+0.48%
15	36	▲▲	Groovy	1.19%	+0.94%
16	14	▼	Ruby	1.18%	+0.13%
17	17		Perl	1.15%	+0.24%
18	15	▼	MATLAB	1.04%	+0.05%
19	13	▼▼	Swift	0.95%	-0.28%
20	19	▼	Objective-C	0.91%	+0.17%

- ❑ O JavaScript é usado essencialmente para:
  - Criar páginas Web interactivas;
  - Validação dos campos dos formulários;
  - Cálculo de valores a partir de dados introduzidos pelo utilizador;
  - Desenho de barras de ferramentas (ToolBars);
  - Menus;
  - Controlo e gestão da navegação e atributos do próprio browser.

## ☐ Segurança

- A linguagem Web foi projectada para ser segura, portanto foi-lhe introduzida algumas restrições:
  - ☐ Não pode abrir, ler, gravar ficheiros no computador do utilizador;
    - A única informação que o java pode aceder é a que está na página Web.
  - ☐ Não pode abrir, ler, gravar ficheiros no Servidor Web;
  - ☐ Não pode ser usada para criar um “vírus” que danifique alguma coisa no computador.
    - O pior que pode acontecer é uma página mal projectada fazer aparecer uma mensagem de erro – e termos que encerrar o browser.
- Em resumo:
  - ☐ JavaScript existe apenas no seu pequeno mundo particular – o mundo de página Web.

## ❑ Modelo de programação

- Todos os conceitos fundamentais de programação estão presentes

- ❑ Variáveis

- var

- ❑ Operadores

- Altimétricos
  - Comparação
  - Lógicos,
  - etc..

- ❑ Leitura de dados

- window.prompt
  - prompt

- ❑ Escrita de dados

- Document.write()

## ❑ Modelo de programação (cont)

### ❑ Estruturas de controlo

- if
- Switch

### ❑ Ciclos

- for
- While

### ❑ Janelas de dialogo

- alert()
- confirm()

### ❑ Objectos

- Document
- Location
- Etc..

## □ Modelo de programação (cont)

- Arrays
- Cookies
- Métodos
  - Matemáticos
    - `abs()`, `random()`, `sqrt()`, etc.
  - Trigonométricos
    - `sin()`, `cos()`, `tan()`, etc.
  - String
    - `bold()`, `fontcolor()`, etc.
    - `toLowerCase()`, `toUpperCase()`
  - Data
    - `getDate()`
    - `getTime()`
    - `setDate()`
    - `setTime()`
    - Etc...

## □ Modelo de programação (cont)

### □ Funções

### □ Eventos

#### ■ Do sistema

- OnLoad()
- OnUnload()

#### ■ Rato

- OnClick()
- OnFocus()
- OnBlur()
- OnChange()
- OnSelect()
- OnSubmit()
- OnMouseOver()



# Javascript versus Java

---

- Apesar de ambas as linguagens, JavaScript e Java, partilharem a palavra "java" são completamente diferentes.
- Java foi desenvolvido pela Sun Microsystems e JavaScript pela Netscape
  - justifica a independência de tais linguagens.

JavaScript	Java
<ul style="list-style-type: none"><li>◆ Interpretada</li><li>◆ Baseada em objectos, mas sem a noção de classe e herança</li><li>◆ Código embebido no HTML</li><li>◆ Variáveis não declaradas</li><li>◆ Referência aos objectos no acto de execução</li></ul>	<ul style="list-style-type: none"><li>◆ Compilada no servidor antes da execução no browser do cliente</li><li>◆ Totalmente orientada por objectos</li><li>◆ Acedido a partir do HTML</li><li>◆ Variáveis declaradas</li><li>◆ Referência aos objectos testada na pre-compilação.</li></ul>

- Exemplo "Hellow world" nas duas linguagens

JavaScript	Java
<b>Hello.html</b> <HTML> <BODY>  <SCRIPT Language="JavaScript"> document.write("Olá mundo") </SCRIPT>  </BODY> </HTML>	<b>Hello.html</b> <HTML> <BODY> <APPLET Code="hello.class" Width=100 Height=25> </APPLET> </BODY> </HTML>  <b>hello.java</b> Import java.awt.Graphics; Import java.applet.Applet; Public class hello extends Applet{ Public void paint(Graphics g){ g.drawString("Olá Mundo",25,25); } }

- Em ambos os exemplos a frase "Olá mundo" surge na página.
  - No entanto, enquanto o JavaScript o faz para a janela do browser, em java tem obrigatoriamente de abrir uma *applet*, ou seja, um ficheiro próprio para programas em java.
- Há a realçar que o código JavaScript está no próprio documento, enquanto em Java temos de fazer o código num ficheiro com a extensão java, transformá-lo (compilar) em hello.class, com as ferramentas adequadas, e posteriormente criar um documento html que irá abrir a classe compilada.

# Sintaxe Javascript

---

- ❑ Não tem em conta os espaço em branco.
  - Ignora quaisquer espaços em branco ou linhas
- ❑ Case sensitive
  - Faz distinção entre maiúsculas e minúsculas
- ❑ Não se define o tipo de variáveis
  - Ao longo do script a mesma variável pode assumir vários valores
- ❑ Terminação de linhas com ;
  - Não sendo obrigatório, devemos colocar
- ❑ Podemos usar comentários
  - `// comentário de uma linha`
  - `/* comentarios com  
várias linhas */`

# Usar Javascript

---

- ❑ A grande vantagem é a possibilidade de ser embebido directamente nos documentos html através do elemento script, como se pode ver de seguida:

```
<script>
```

Programação em código JavaScript

```
</script>
```

## Exemplo:

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo JS</title>
</head>
<body>
```

```
<script>

  document.write("Olá mundo!");

</script>
```

```
</body>
</html>
```

Colocar o script aqui  
O que acontece

**DOM**  
Document Object Model



- ❑ Podemos colocar o código javascript num ficheiro externo (.js) e no documento HTML fazer o link.
- ❑ Exemplo:

- .js

```
document.write("Olá mundo!")
```

- .html

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo JS</title>
  <script src="js/exemplo.js"></script>
</head>
<body>

</body>
</html>
```

## ❑ Elemento <noscript>

- Permite definir uma mensagem para ser mostrada ao utilizador cujo Browser não permite a execução de javascript.

- ❑ Nos browsers podemos desactivar a execução de javascript.

```
<!DOCTYPE html>
<html lang="pt">
  <head>
    <title>Exemplo JavaScript</title>
  </head>

  <body>

    <noscript>
      <p>Bem vindo ao sitio Web</p>
      <p>Esta pagina requer a execução de javascript! Por favor activar!</p>
    </noscript>

  </body>

</html>
```

# JavaScript

## Funções

---

## □ Funções

### ■ No próprio documentos HTML

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Exemplo JS - Funções</title>
```

```
<script>

  function EscreveDados() {

    document.write("<h2>" + "Olá Mundo" + "</h2>");

  }

</script>>
```

```
</head>
<body>

  <h1>Funções JS</h1>
```

```
<script>
  EscreveDados();

  EscreveDados();
</script>
```

```
</body>
</html>
```

## □ Funções

- Num ficheiro externo (.js)

```
//  
// Funções JS  
//  
function EscreveDados() {  
  
    document.write("<h2>" + "Olá Mundo" + "</h2>");  
  
}
```

exemplos.js

```
<!DOCTYPE html>  
<html lang="pt">  
<head>  
    <meta charset="UTF-8">  
    <meta http-equiv="X-UA-Compatible" content="IE=edge">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Exemplo JS</title>
```

```
<script src="js/exemplo.js"></script>
```

```
</head>  
<body>
```

```
<h1>Funções JS Externas</h1>
```

```
<script>
```

```
    EscreveDados();  
    EscreveDados();
```

```
</script>
```

```
</body>  
</html>
```

html

# JavaScript

## Variáveis

---

- ❑ As variáveis são representadas por nomes chamados de identificadores. Os identificadores devem ser utilizados segundo as regras seguintes:
  - devem iniciar, obrigatoriamente, por uma letra, por um cifrão "\$" ou por um sinal de underscore ("\_");
  - seguir à letra ou underscore podem conter qualquer quantidade de números ou letras, maiúsculas ou minúsculas;
  - não é conveniente utilizar letras acentuadas nas variáveis;
  - exemplos de variáveis válidas:  
\$casa, \_rua, X11, \$EfeitosVisuais, etc.
- ❑ Para declarar uma variável:
  - var
  - let
- ❑ Para declarar um constante:
  - const

## ❑ Exemplo:

- Directamente como, por exemplo:

nome = "Aprender JavaScript";

- Por meio do uso da palavra reservada "var" como, por exemplo:

var nome = "Aprender JavaScript";

```
var carName = "Volvo";
```

Variables declared **Globally** (outside any function) have **Global Scope**.

```
// code here can use carName
```

```
function myFunction() {  
  // code here can also use carName  
}
```

**Global** variables can be accessed from anywhere in a JavaScript program.

```
// code here can NOT use carName
```

```
function myFunction() {  
  var carName = "Volvo";  
  // code here CAN use carName  
}
```

Variables declared **Locally** (inside a function) have **Function Scope**.

```
// code here can NOT use carName
```

**Local** variables can only be accessed from inside the function where they are declared.



```
var x = 10;  
// Here x is 10  
{  
  var x = 2;  
  // Here x is 2  
}  
// Here x is 2
```

```
var x = 10;  
// Here x is 10  
{  
  let x = 2;  
  // Here x is 2  
}  
// Here x is 10
```

```
const PI = 3.141592653589793;  
PI = 3.14;      // This will give an error  
PI = PI + 10;   // This will also give an error
```

# JavaScript

## Operadores

---

## ❑ Operadores de atribuição

Operador	Descrição	Exemplo	Resultado
=	O operando da esquerda recebe o valor associado ao lado direito.	n1=n2;	n1=4
+=	Soma o operando da direita e da esquerda ficando guardado no primeiro.	n1+=n2;	n1=6
-=	O operador da esquerda recebe o o resultado da subtração entre o n1 e o n2.	n1-=n2;	n1=(-2)
*=	O operador da esquerda recebe o resultado da multiplicação entre o n1 e o n2.	n1*=n2;	n1=8
/=	O operador da esquerda recebe o quociente da divisão entre o n1 e o n2.	n1/=n2;	n1=2
%=	O operador da esquerda recebe o resto da divisão entre o n1 e o n2.	n1%=n2;	n1=2

(Supor que n1 e n2 tem os valor de 2 e 4)

Outros exemplos:

```
var soma = num1 + num2;  
num1++;
```

## ❑ Operadores de comparação

Operador	Descrição	Exemplo
<code>==</code>	Devolve verdade se o operando da esquerda for o mesmo do da direita.	<code>If(n1==n2){ }</code>
<code>!=</code>	Devolve verdade se os operandos forem diferentes.	<code>If(n1!=n2){ }</code>
<code>&gt;</code>	Devolve verdade se o operando da esquerda for maior que o da direita.	<code>If(n1&gt;n2){ }</code>
<code>&lt;</code>	Devolve verdade se o operando da esquerda for menor que o da direita.	<code>If(n1&lt;n2){ }</code>
<code>&gt;=</code>	Devolve verdade se o operando da esquerda for maior ou igual que o da direita.	<code>If(n1&gt;=n2){ }</code>
<code>&lt;=</code>	Devolve verdade se o operando da esquerda for menor ou igual que o da direita.	<code>If(n1&lt;=n2){ }</code>

## ❑ Operadores lógicos

Operador	Descrição	Exemplo
<b>&amp;&amp; - And</b>	Se ambos os operadores forem verdade então devolve verdade. Caso contrário falso.	<pre>If(x==2 &amp;&amp; y==1){ }</pre>
<b>   - Or</b>	Se um dos operadores for verdade então devolve verdade. Caso contrário devolve falso.	<pre>if(x==2    y==1){ }</pre>
<b>! - Not</b>	Se o operando for falso devolve verdade.	<pre>if(!x){ }</pre>

## ❑ Operadores de string

Operador	Descrição	Exemplo
=	O operando da esquerda passa a aguardar uma frase de caracteres.	<code>var str1 ="Java";</code>
+	Soma duas frases de caracteres.	<code>ans=str1+"Script";</code>

# JavaScript

## Janelas de dialogo

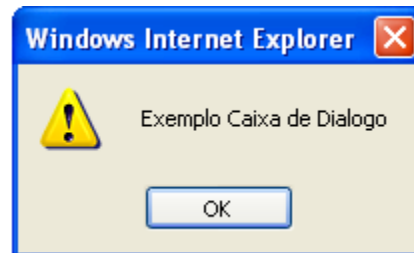
---

## □ alert()

- O método *alert()* cria uma caixa de dialogo

- Sintaxe:

- `alert("mensagem");`
  - `alert(variavel);`

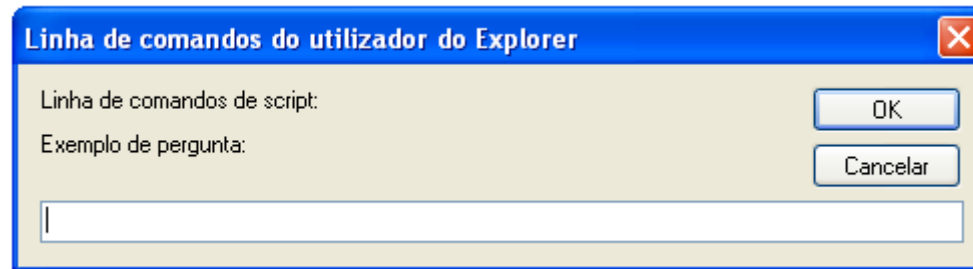




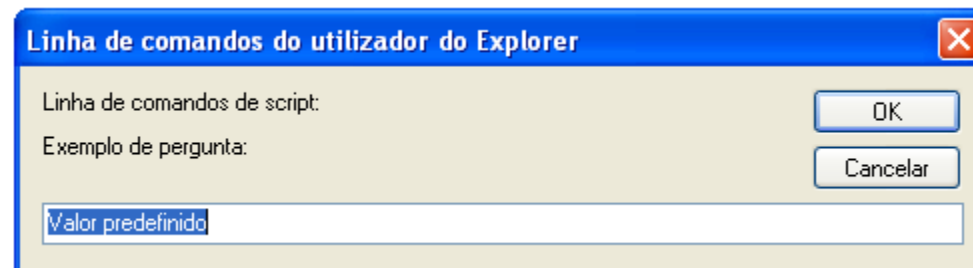
## □ `prompt()`

- O método *prompt()* recebe dados do utilizador
- Sintaxe:

- `prompt("mensagem");`



- `prompt("mensagem", "ValorPadrao");`



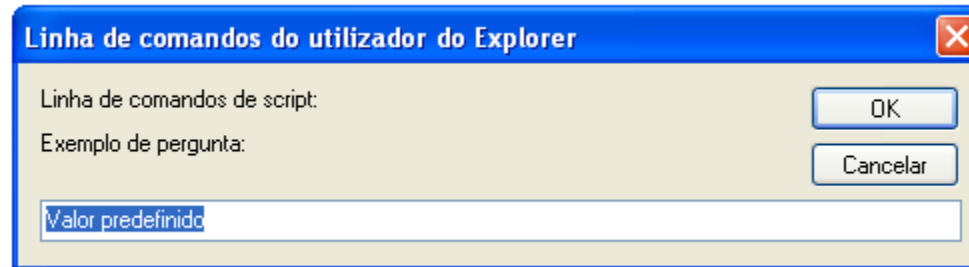
## □ prompt()

### ■ Guardar numa variável

#### □ Sintaxe:

Varivel = prompt("Mensagem");

Varivel = prompt("Mensagem" ,"ValorPadrao");

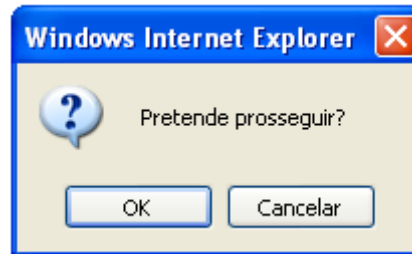


## □ confirm()

- Permite questionar o utilizador antes de executar um determinado processo

## □ Sintaxe:

- `confirm("Mensagem")`



# JavaScript

## Condições

---

- Permitem controlar que conjunto de declarações javascript será executado, com base numa condição (uma expressão ser verdadeira ou falsa)

- Sintaxe

- `if (condicao) {  
    //comandos  
}`

- `if (condicao) {  
    //comandos se condição verdadeira  
} else {  
    //comandos se condição falsa  
}`

Se a condição contiver apenas um linha de comando não é necessário as {}

## □ Selecção de casos

```
■ switch(n)
{
  case 1:
    código do bloco 1
    break;

  case 2:
    código do bloco 2
    break;

  default:
    código para qualquer outro valor
}
```

# JavaScript

## Ciclos

---

## □ for

- O ciclo *for* executa um determinado segmento de código, uma certa quantidade de vezes, até atingir uma certa condição. Este comando tem a forma seguinte:

```
for (variável; condição; incremento) {  
    comandos;  
}
```

Exemplo:

```
for(var conta=0; conta < 5; conta++)  
    alert(conta);
```

Se o ciclo contiver mais do que um linha de comandos é necessário as {}



## □ while

- O ciclo **for** é adequado para ciclos que têm um início e um fim conhecido.
- Se precisarmos de repetir um bloco de código até que uma condição que pode depender de um evento ou interacção do utilizador – a finalização não é conhecida, temos o ciclo **while** – por exemplo até que o utilizador introduza o valor zero.

- Sintaxe:  
while (condição)  
{  
 Comandos;  
}

### Exemplo:

```
var i = 1;  
while (i <= 5) {  
    document.write(i + "<BR>");  
    i++;  
}
```

## □ While (cont)

### Exemplos:

```
while ( X == 10)
{
    alert("X é igual a 10");
}
```

Neste caso, a mensagem "X é igual a 10" será mostrada apenas se X for igual a 10. So entra dentro do ciclo se o valor de X for 10. – ciclo infinito

```
do
{
    COMANDO 1;
    COMANDO 2;
    ...
    COMANDO n;
}
while (condição);
```

A diferença entre esta forma de escrita do WHILE e a anterior é que nesta o bloco de comandos será executado, pelo menos uma vez, mesmo que a condição seja falsa, pois o teste da condição só é feito depois de já ter executado uma vez o bloco.

# JavaScript

## Funções

---

## □ Sintaxe

```
function name(parameter1, parameter2, parameter3) {  
    // code to be executed  
}
```

## □ Exemplo:

```
function myFunction(p1, p2) {  
    return p1 * p2;    // The function returns the product of p1 and p2  
}
```

```
var x = myFunction(4, 3);    // Function is called, return value will end up in x
```

```
function myFunction(a, b) {  
    return a * b;    // Function returns the product of a and b  
}
```

## □ Exemplo:

```
function toCelsius(fahrenheit) {  
    return (5/9) * (fahrenheit-32);  
}  
document.getElementById("demo").innerHTML = toCelsius(77);
```

```
<!DOCTYPE html>  
<html>  
<body>  
  
<h2>JavaScript Functions</h2>  
  
<p>This example calls a function to convert from Fahrenheit to Celsius:</p>  
<p id="demo"></p>  
  
<script>  
function toCelsius(f) {  
    return (5/9) * (f-32);  
}  
document.getElementById("demo").innerHTML = toCelsius(77);  
</script>  
  
</body>  
</html>
```

## □ Exemplo:

```
function toCelsius(fahrenheit) {  
    return (5/9) * (fahrenheit-32);  
}  
document.getElementById("demo").innerHTML = toCelsius(77);
```

```
var x = toCelsius(77);  
var text = "The temperature is " + x + " Celsius";
```

```
var text = "The temperature is " + toCelsius(77) + " Celsius";
```

# JavaScript

## referencias

---



- ❑ <http://www.w3schools.com/js/default.asp>
- ❑ <http://www.w3schools.com/jsref/default.asp>



- ❑ <https://www.javascript.com/>