HTML5

HTML 5 (HyperText Markup Language, versión 5) es la quinta revisión importante del lenguaje básico de la World Wide Web, HTML. HTML5 especifica dos variantes de sintaxis para HTML: una «clásica», HTML (text/html), conocida como HTML5, y una variante XHTML conocida como sintaxis XHTML 5 que deberá servirse con sintaxis XML (application/xhtml+xml). Esta es la primera vez que HTML y XHTML se han desarrollado en paralelo. La versión definitiva de la quinta revisión del estándar se publicó en octubre de 2014.

Al no ser reconocido en viejas versiones de navegadores por sus nuevas etiquetas, se recomienda al usuario común actualizar su navegador a la versión más nueva, para poder disfrutar de todo el potencial que provee HTML 5.

El desarrollo de este lenguaje de marcado es regulado por el Consorcio W3C.

Nuevos elementos
HTML5 establece una serie de nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos. Algunos de ellos son técnicamente similares a las etiquetas y , pero tienen un significado semántico, como por ejemplo <nav> (bloque de navegación del sitio web) y <footer>.

Novedades
Incorpora etiquetas (canvas 2D y 3D, audio, vídeo) con codecs para mostrar los contenidos multimedia. Actualmente hay una lucha entre imponer codecs libres (WebM + VP8) o privados (H.264/MPEG-4 AVC).
Etiquetas para manejar grandes conjuntos de datos: Datagrid, Details, Menu y Command. Permiten generar tablas dinámicas que pueden filtrar, ordenar y ocultar contenido en cliente.
Mejoras en los formularios. Nuevos tipos de datos (eMail, number, url, datetime …) y facilidades para validar el contenido sin Javascript.
Visores: MathML (fórmulas matemáticas) y SVG (gráficos vectoriales). En general se deja abierto a poder interpretar otros lenguajes XML.
Drag & Drop. Nueva funcionalidad para arrastrar objetos como imágenes.
Web Semántica
Añade etiquetas para manejar la Web semántica (Web 3.0): header, footer, article, nav, time (fecha del contenido), link rel=‘’ (tipo de contenido que se enlaza).
Estas etiquetas permiten describir cuál es el significado del contenido. Por ejemplo su importancia, su finalidad y las relaciones que existen. No tienen especial impacto en la visualización, se orientan a buscadores.
Los buscadores podrán indexar e interpretar esta meta información para no buscar simplemente apariciones de palabras en el texto de la página.
Permite incorporar a las páginas ficheros RDF / OWL (con meta información) para describir relaciones entre los términos utilizados.
Además, ofrece versatilidad en el manejo y animación de objetos simples, imágenes etc.
Nuevas API y Javascript
API para hacer Drag & Drop. Mediante eventos.
API para trabajar Off-Line. Permite descargar todos los contenidos necesarios y trabajar en local.
API de Geolocalización para dispositivos que lo soporten.
API Storage. Facilidad de almacenamiento persistente en local, con bases de datos (basadas en SQLite) o con almacenamiento de objetos por aplicación o por dominio Web (Local Storage y Global Storage). Se dispone de una Base de datos con la posibilidad de hacer consultas SQL.
WebSockets. API de comunicación bidireccional entre páginas. Similar a los Sockets de C.

WebWorkers. Hilos de ejecución en paralelo.
Estándar futuro. System Information API. Acceso al hardware a bajo nivel: red, ficheros, CPU, memoria, puertos USB, cámaras, micrófonos... Muy interesante, pero con numerosas salvedades de seguridad.
Ejemplos de códigos HTML5
Código HTML5 para reproducir audio sin la necesidad de plugins
Para video es algo similar.

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>fuente de múltiples elementos</title>
  </head>
  <body>
    <audio id="audioTestElem" autobuffer controls >
      <source src="test.m4a">
      <source src="test.ogg" type="audio/ogg; codecs=vorbis">
      <source src="url">
      no audio for you
    </audio>
  </body>
</html>
```

Ejemplo de WebWorker (Hilo de ejecución en paralelo)
Es necesario el uso de javascript.
Para el archivo Prueba.html

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Worker example: One-core computation</title>
 </head>
 <body>
  <p>The highest prime number discovered so far is: <output id="result"></output></p>
  <script>
   var worker = new Worker('worker.js');
   worker.onmessage = function (event) {
     document.getElementById('result').textContent = event.data;
   };
  </script>
 </body>
</html>
```

Para el archivo worker.js (fichero con la tarea del nuevo hilo de ejecución infinito)

```
var n = 1;
search: while (true) {
  n += 1;
  for (var i = 2; i <= Math.sqrt(n); i += 1)
    if (n % i == 0)
      continue search;
  // found a prime!
  postMessage(n);
}
```

Ejemplo de Canvas 2D utilizando el API de dibujo

```
<!DOCTYPE HTML>
<html>
```

```
<head>
  <title>HTML5 Canvas example</title>
  <script>
    function drawPicture(){
      // Primero se recupera el objeto canvas a modificar
      var canvas = document.getElementById('example');
      // Luego se le indica la forma de trabajar 2D o 3D
      var context = canvas.getContext('2d');
      // Se comienza a dibujar en el lienzo utilizando objetos
      // gráficos
      context.fillStyle = "rgb(0,255,0)";
      context.fillRect (25, 25, 100, 100);
      context.fillStyle = "rgba(255,0,0, 0.6)";
      context.beginPath();
      context.arc(125,100,50,0,Math.PI*2,true);
      context.fill();
      context.fillStyle = "rgba(0,0,255,0.6)";
      context.beginPath();
      context.moveTo(125,100);
      context.lineTo(175,50);
      context.lineTo(225,150);
      context.fill();
    }
  </script>
  <style type="text/css">
    canvas { border: 2px solid black; }
  </style>
</head>
<body onload="drawPicture();">
  <canvas id="example" width="260" height="200">
  There is supposed to be an example drawing here, but it's not important.
  </canvas>
</body>
</html>
```

Ejemplo de un formulario con nuevos tipos de datos

Elimina muchas validaciones en Javascript. (La clave está en el atributo Type).

```
<!DOCTYPE HTML>
<html>
 <body>
 <form>
  <input name="form_number" id="form_number" type="number" min="1" max="10" >
  <input name="form_date" id="form_date" type="date">
  <input name="form_month" id="form_month" type="month">
  <input name="form_week" id="form_week" type="week">
  <input name="form_time" id="form_time" type="time">
  <input name="form_url" id="form_url" type="url" list="url_list">
  <datalist id="url_list">
     <option value="http://www.google.com" label="Google">
     <option value="http://net.tutsplus.com" label="NetTuts+">
  </datalist>
  <input name="form_email" id="form_email" type="email" list="email_list" multiple>
  <datalist id="email_list">
```

```
      <option value="jane.doe@test.com" label="Jane Doe">
      <option value="john.doe@test.com" label="John Doe">
    </datalist>
    <input name="form_telephone" id="form_telephone" type="tel">
    <input name="form_color" id="form_color" type="color">
    <label>
       Attachments:
       <input type="file" multiple name="att">
    </label>
    <input name="x" type="range" min="100" max="700" step="9.09090909" value="509.090909">
  </form>
  </body>
</html>
```

Ejemplo de geoposicionamiento

```
<!DOCTYPE HTML>
<html>
<head>
<title> Geo  Localizations </title>
</head>
<body>
<script language="javascript">
function obtener_localizacion() {
  navigator.geolocation.getCurrentPosition(coordenadas);
}
function coordenadas(position) {
  var latitud = position.coords.latitude;
  var longitud = position.coords.longitude;
  alert('Tus coordenadas son: ('+latitud+','+longitud+')');
}
</script>
<a href="javascript:obtener_localizacion();">Mostrar Posición</a>
</body>
</html>
```