TEMA 01 – INTRODUCCIÓN A LENGUAJE DE MARCAS

Qué es

* Modo de codificar un documento de texto donde junto al contenido se incluyen etiquetas, marcas o anotaciones
* Fin: añadir información relativa a la estructura del texto o su formato de presentación
* Algunas de las marcas más empleadas se delimitan con los símbolos “<” y “>”
* No es un lenguaje de programación
  + Puede contener partes de códigos de otros lenguajes de programación

Características generales

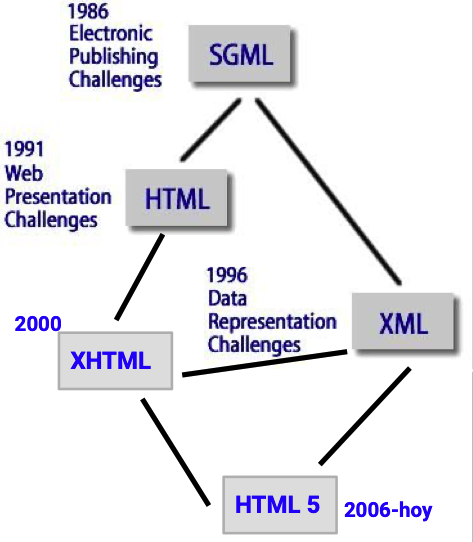
* Texto plano: pueden ser editados con cualquier procesador
* Lenguaje compacto: las etiquetas se mezclan con el contenido
* Facilidad de procesa-: fácil desarrollar, comprender y aprender
  + Especializados el función del tipo de contenido
* Flexibilidad: posibilidad de combinación con otros lenguajes (gráficos vectoriales, servicios web, interfaces…)

Definición del tipo

* Todo lenguaje de marcas está definido en un documento DTD (Document Type Definition)
* Se establecen:
  + Marcas y elementos
  + Etiquetas y atributos
  + Sintaxis
  + Normas de uso
* Tipos
  + Lenguajes de presentación (ej.: Rich Text Format de Microsoft)
    - Define el formato (apariencia) del texto
    - Suelen ocultar las etiquetas y mostrar el texto con su formato
  + Lenguajes de procedi-s (LaTeX, HTML)
    - También orientado a la presentación
    - El programa que representa el documento debe interpretar las etiquetas para realizar acciones en función de ellas
  + Lenguajes descriptivos o semánticos (XML)
    - Describen las diferentes partes en las que se estructura
    - Es decir, definen su contenido, pero sin especificar cómo deben representarse

Evolución

* Generalized Markup Language (GML): nace ante la necesidad de gestionar documentación entre distintos departamentos y aplicaciones
* Standard GML (SGML): evolución de 1986 a la estandarización del GML
* HyperText MMarkup Language (HTML) (1991)
  + Con el nacimiento del World Wide Web
  + Combinación de ASCII y SGML
  + Desventajas
    - No soporta tareas de impresión y diseño
    - No es flexible (etiquetas limitadas)
    - No permite mostrar contenido dinámico
    - La estructura y el diseño están mezclados en el documento
* eXtensible Markup Language (XML) (1996)
  + Estándar del W3C
  + Lenguaje estructural puro
  + Ventajas
    - Permite definir etiquetas propias
    - Permite asignar atributos a las etiquetas
    - Utiliza un esquema para definir de forma exacta las etiquetas y los atributos
    - Estructura y diseño independientes
* eXtensible HyperText Markup Language (2000)
  + para expresar el HTML como un lenguaje XML válido
  + convertido de un lenguaje de marcas de procedi-s a un lenguaje de marcas descriptivo
  + sin mucho éxito
* HTML5
  + HTML semántico
  + Las etiquetas de presentación quedan obsoletas (deprecated)



Sintaxis

* Etiquetas entre símbolos “<” y “>”
* Todas las etiquetas tienen su correspondiente etiqueta de cierre: <h1>**</h1>**
* Etiquetas y atributos en minúsculas
* Valor de los atributos entre comillas simples o dobles

TEMA 02 – INTRODUCCIÓN A LENGUAJE DE MARCAS XML

XML: eXtensible Markup Language

* 1996
* Lenguaje de marcas porque está definido por marcas o etiquetas
* Metalenguaje porque define las reglas a cumplir por un lenguaje de marcas
* Define la estructura de un documento
* Extensión: .xml
* Documento en texto plano
* Objetivo: estandarización del HTML -> XHTML (sin éxito)
* Uso actual:
  + Intercambio de info entre aplicaciones
  + Computación distribuida
  + Documentación empresarial, etc.

Características básicas

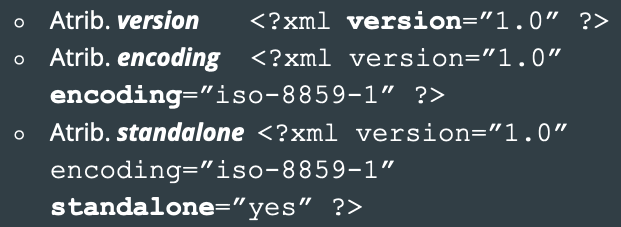
* Concebido para trabajar en la web
  + Compatible con protocolos HTTP
* Todo documento que verifique las deglas de XML está conforme con SGML
* Fáciles de crear
  + No se requieren conocí-s de programación para tareas sencillas
* Diseño formal y conciso
* Extensible y adaptable a variedad de situaciones
* Orientado a objetos
* Se compone exclusivamente de datos de marcado y datos de carácter entremezclados

Algunos conceptos

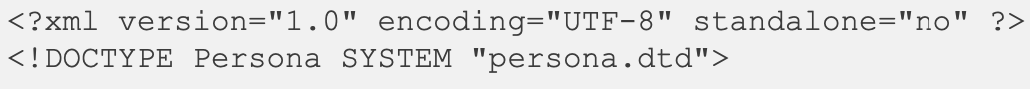
* Caracteres y codificación: ISO-8859-1, UTF-8, etc.
* Procesador XML (XML parser)
  + La aplicación recurre a un procesador XML para leer un documento XML
  + Este procesador es el que lee, analiza y le pasa la información en un formato estructurado a la aplicación
  + Etiquetas (tags) o marcas (mark-up)
    - Las etiquetas son marcas que sirven para identificar un contenido concreto del resto del contenido del documento
* Pueden estar formados por 2 partes:
  + Prólogo (opcional)
    - Declaración XML
    - Declaración del tipo de documento
  + Ejemplar: contenido del documento

Prólogo

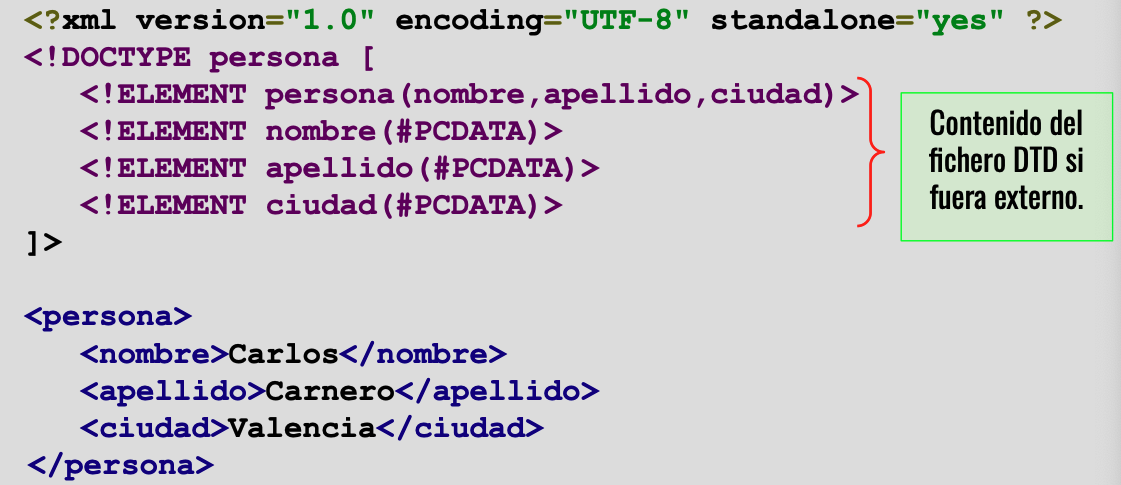
* Debe preceder al ejemplar del documento
* Facilita el procesado de la información del ejemplar
* Declaración XML
  + Opcional, pero obligatorio si se escribe el atributo “versión”
  + Encoding y standalone son opcionales
  + Encoding: siempre después de versión y standalone el último
  + Standalone indica si el documento XML depende de un DTD externo (sirve a nivel de validación)



* La declaración define el tipo de documento
  + Se define el tipo de documento y dónde encontrar la información sobre su DTD
  + Mediante un identificador público (PUBLIC) que hace referencia a dicho DTD
  + O mediante Identificador Universal de Recursos (URI) precedido por la palabra SYSTEM
  + Este archivo valida y define los datos que contiene nuestro documento XML
  + DTD en archivo externo:

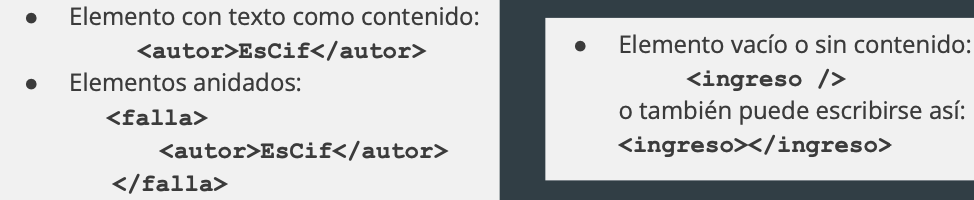


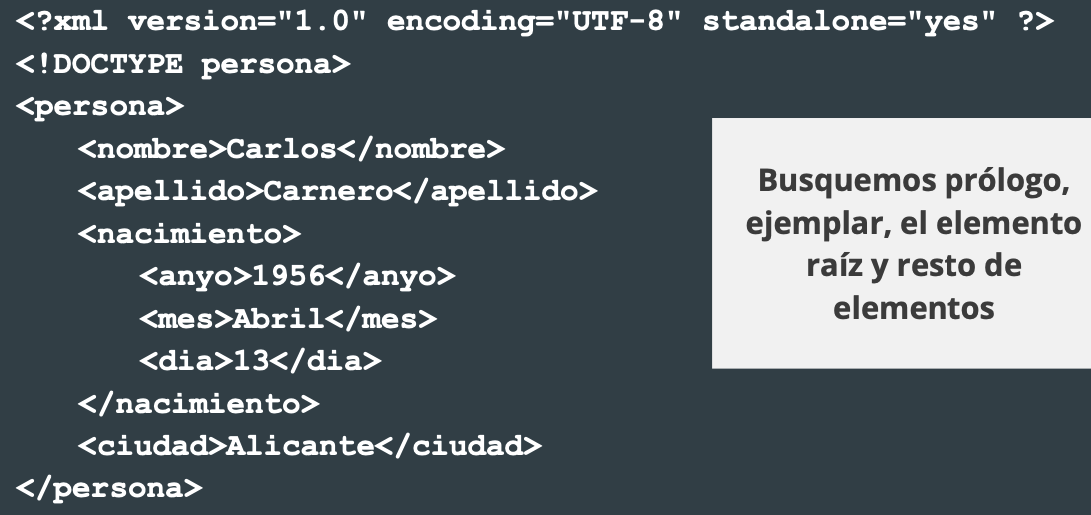
DTD embebido en documento



Ejemplar y elementos

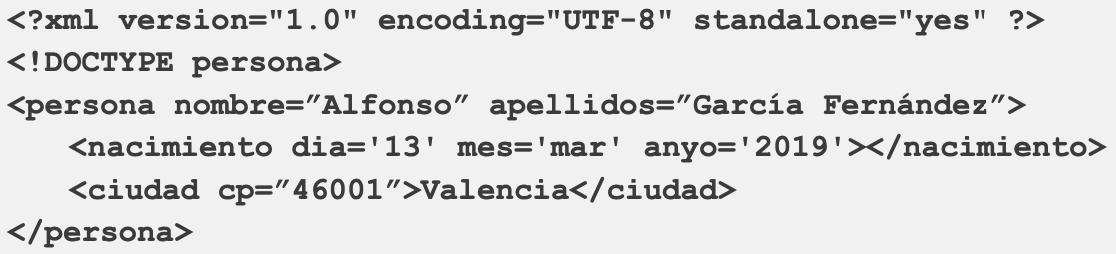
* El ejemplar contiene los datos reales del documento XML (contenido)
* Formado por un elemento raíz que a su vez contendrá otros elementos anidados
* Elemento: componente lógico de un documento XML
  + Comienza por una etiqueta de apertura y termina con la etiqueta de cierre correspondiente (o bien consiste en una única etiqueta vacía)
  + El contenido de un elemento es todo lo que se encuentra entre las etiquetas
  + También existen elementos vacíos





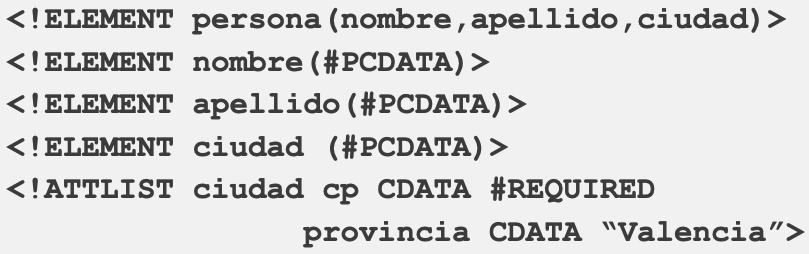
Atributos

* Componente de las etiquetas que consiste en una pareja nombre/valor
* Se puede encontrar en las etiquetas de apertura o vacías
* No puede +1 atributo con mismo nombre por etiqueta
* Sintaxis: nombreAtributo=”valorAtributo”



Definir atributos en el DTD

* Usaremos la etiqueta !ATTLIST indicando a continuación el nombre del elemento al que pertenecen los elementos que se van a definir



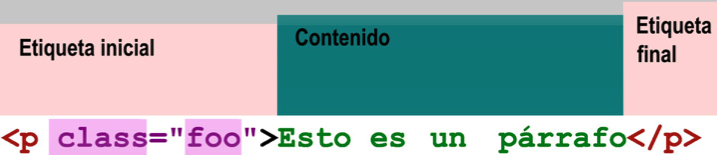
TEMA 03 – INTRODUCCIÓN A LENGUAJE DE MARCAS HTML

DTD de un documento HTML

* DTD público
* Su metaetiqueta DOCTYPE se ha simplificado al extremo
* Antes: **<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">**
* Ahora: **<!DOCTYPE html>**

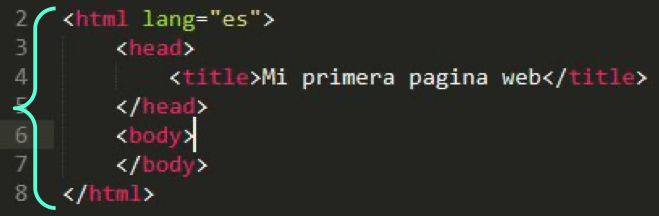
Sintaxis básica

* <>
* Etiqueta cierre (menos img, br, hr e input)
* Etiquetas y atributos en minúsculas
* Comentarios entre <!—y -->



Estructura de un documento HTML5

* Prólogo
  + DTD
* Ejemplar
  + Cabecera (head)
    - Title
    - Meta
    - Estilos (mejor CSS)
    - Enlaces (como para el CSS)
  + Cuerpo
    - Contenido
    - Scripts (enlaces)



* + Elemento raíz: <html></html>

Marcas de formato

* Encabezados: <h1></h1>… <h6></h6>
* Párrafos: <p>Lorem</p>
* Salto de línea: <br />
* Separador horizontal: <hr />
* Comentarios: <!--- … -->
* Marcas de lista: <li> … </li>
  + Ordenadas: <ol>
  + No ordenadas: <ul>
* Listas de definición
  + <dl>
    - <dt>
    - <dd>
* Enlace: <a> … </a>
  + Enlace externo:

<a href=”http://google.es”>Ir a Google</a>

* + Enlace a otro documento de la web:

<a href=”contacto.html”>Contacto</a>

* + Ancla a sección del mismo documento:

<a href=”#aficiones”>Aficiones</a>

* + - Para indicar el párrafo al que desas enlazar:

<a id=“aficiones”>aficiones</a>

* + Enlace a correo electrónico:

<a href=”mailto:micorreo@dominio.es”>Mandar email a micorreo@dominio.es</a>

* + Enlace nulo:

<a href=”#”>Sin salida</a>

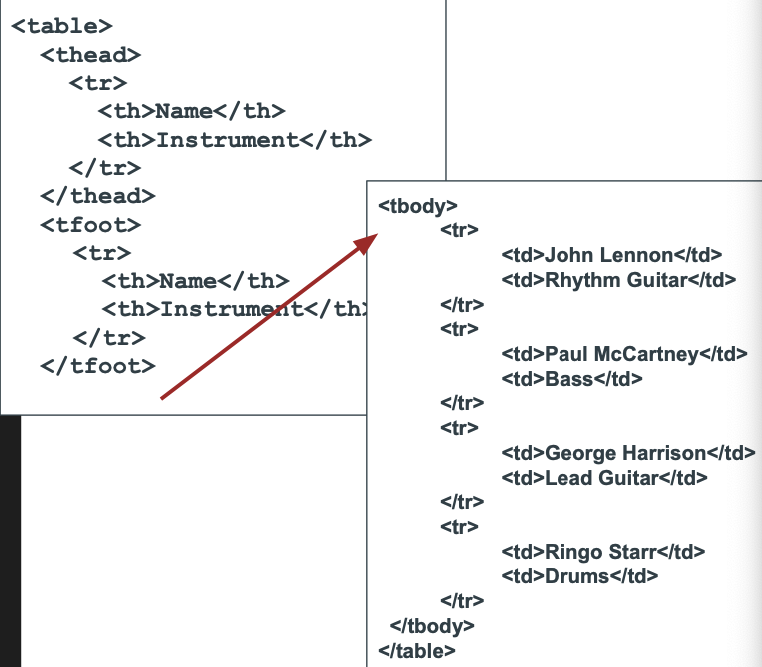
* Inserción imagen: <img src=”source”>

TEMA 04 – ETIQUETAS HTML

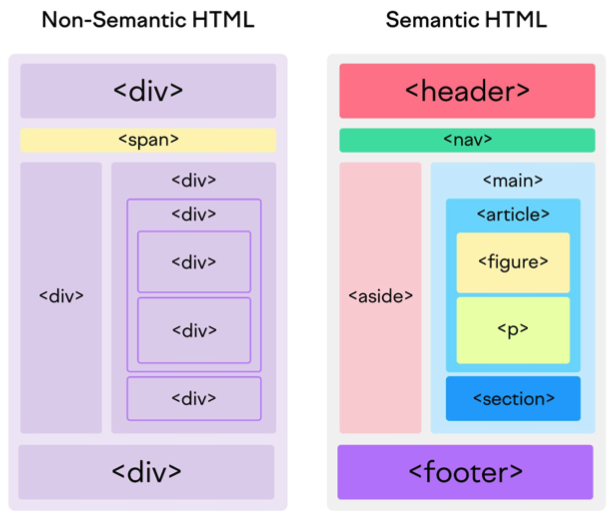
Marcas de aspecto

* Énfasis: <em> … </em>
* Negrita: <strong> … </strong>
* Marcado: <mark> … </mark>
* Entrecomillado: <q> … </q>
* Preformateado: <pre> … </pre>
* Otras:
  + Código: <code> … </code>
  + Subrayado: <ins> … </ins>
  + Tachado: <del> … </del>
  + Subíndice: <sub> … </sub>
  + Superíndice: <sup> … </sup>
  + Texto pequeño: <small> … </small>
  + Texto grande: <big> … </big>
* Se pueden anidar

Tablas



TEMA 05 – DIVS SEMÁNTICOS



* Iguales a una etiqueta div genérica
* Nombres simple- para dar sentido a su contenido

Header: nombre del sitio y logotipo

Footer: detalles legales, copyright, etc.

Nav: menu principal (puede ir dentro del header)

Main

* Contenido importante o principal
* No descendiente de un elemento que no sea el body

Section

* Contenido relacionado entre sí

Normal- contiene una o varias etiquetas aricle

Article

* Información que tiene estructura común dentor de su section
* Todas las etiquetas article dentro de una etiqueta section tienen la misma estructura

Aside

* Contenido secundario (anuncios, enlaces externos, cita del texto…)
* Normalmente en los lados
* Recoge contenido secundario con respecto a su etiqueta madre (body, section o article)

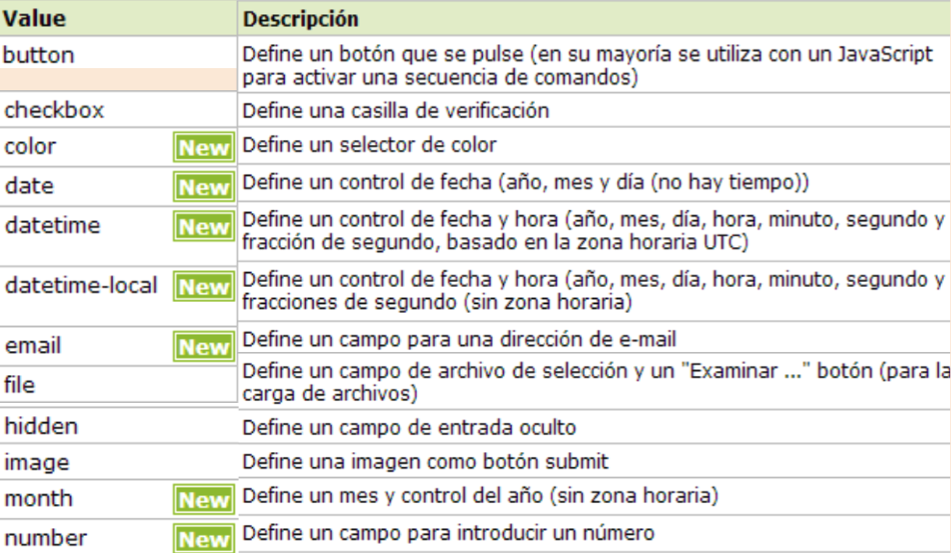
TEMA 06 – FORMULARIOS

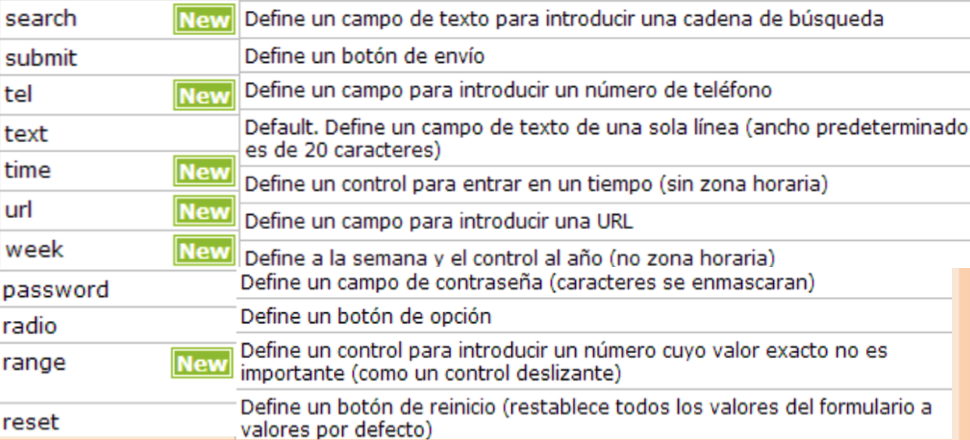
Formularios

* Para recoger datos de los usuarios (pedido, encuesta, preguntas…)
* Una vez enviados, se arranca un programa que recibirá los datos y hará el trata- correspondiente
* Se necesitan algunas nociones de programación (PHP o JS)
* Formato por etiquetas, campos de texto, menús y botones
* HTML5: validaciones nativas y nuevas etiquetas

Etiquetas

* Form
  + <form> … </form>
  + Para crear el formulario
  + Encierra todos los elementos del formulario
  + Atributos:
    - id=”x”: representa el id o nombre del formulario para el CSS o scripts
    - action=”url”: indica la URL donde se enviarán los datos y que se encargará de procesarlos
      * Esta aplicación también se encarga de generar la respuesta que muestra el navegador
    - method=”x”: especifica el método HTTP que se empleará para enviar los datos
      * get: por defecto
        + No adjuntos
        + 500 bytes máximo
        + Se ve en la barra de direcciones del navegador
        + Para formularios que no modifican la información (ej: formulario de búsqueda)
      * post:
    - enctype=”application/x-www-form-urlencoded” o “multipart/form-data”
      * Tipo de codificación empleada al enviar el formulario al servidor
      * Solo se indica explícita- en los formularios que permiten adjuntar archivos
    - accept=”tipo\_de\_contenido\_ MIME”: lista separada por comas de todos los tipos de archivos aceptados por el servidor desde el formulario
  + La mayoría utilizar solo id, action y method
* Input
  + <input />
  + La mayoría de controles del formulario
  + Definición y lista de atributos muy extensa
  + Atributos
    - ‘src’: permite crear botones con imágenes (ruta de la imagen)
    - ‘step’ (HTML5): permite establecer la cantidad de valores posibles dentro de un rango
    - ‘value’ indica el valor iniciar del control
    - ‘autofocus’ (HTML5): fuerza el foco una vez carga la página
      * En JS igual con la función focus()
    - ‘required’ (HTML5): obligatorio
    - ‘placeholder’ (HTML5): texto por defecto hasta tener el foco
    - ‘autocomplete’ (HTML5): activa o desactiva (on/off) el auto completado en base a los textos introducidos anterior- por el usuario
      * Puede usarse como atributo para un campo o para el form
    - ‘alt’: cuando el tipo es una imagen y no se puede cargar en la interfaz de usuario, se muestra este texto alternativo
    - ‘checked’: para indicar si el elemento está seleccionado por defecto en los controles checkbox y radiobutton
    - ‘disabled’: booleano que indica que está desactivado (no admite entrada de datos) y no se envía al servidor
    - ‘readonly’: booleano que indica que el valor es solo lectura
    - ‘formenctype’ (HTML5): indica el modo de envío
      * ‘application/x-www-form-urlencoded’
      * ‘multipart/form-data’
      * /text/plain’
    - ‘formaction’ (HTML5): dirección URL que se utiliza al activar el formulario
    - 'height' (HTML5): altura del elemento tipo image
    - ‘width’ (HTML5): anchura del elemento tipo image
    - ‘size’: tamaño inicial del control
      * En text/password: cantidad de caracteres, en resto píxeles
    - ‘max’ (HTML5): valor máximo que se puede entrar
    - ‘min’ (HTML5): valor mínimo que se puede entrar
    - ‘maxlength: longitud máxima para text y passoword
    - ‘formnovalidate’ (HTML5): valor booleano que indica que el formulario no se valida antes de enviarse
    - ‘multiple’ (HTML5): valor booleano que indica que se permite la selección de valores múltiples
    - ‘name’: nombre que identifica al campo de datos (importante para identificación por el servidor)
    - ‘pattern’ (HTML5): expresión regular que se utiliza para validar entradas de datos
      * Permite hacer validaciones muy complejas y utilizando correcta- puede ahorrar muchas líneas de código
  + Elementos (<input type=”valor” />)

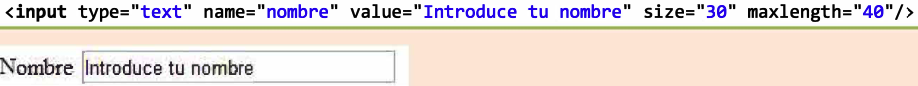




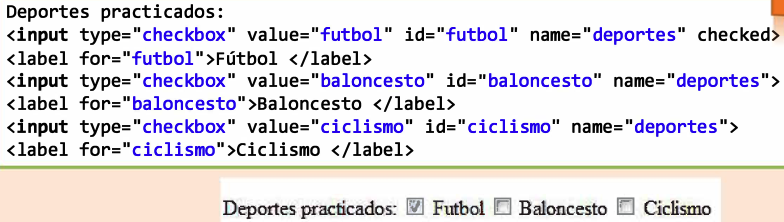
* + Todos:

<https://mardeasa.es/descargas/formularios/Formularios-camposHTML5/Formularios-camposHTML5.html>

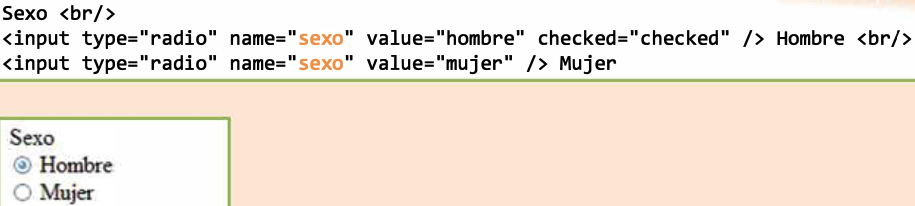
* + Text
    - Pedir datos de entrada relativamente cortos
      * ‘name’: identificador donde se guardará el valor del usuario y como referencia en scripts (más recomendado id para esto)
      * ‘value’: valor predeterminado en el campo de entrada
      * ‘size’: longitud visible
      * ‘maxlength’: longitud máxima de caracteres



* + Password:
    - Igual que text, pero aparece oculto
    - No codificado, solo oculto en pantalla
  + Checkbox
    - Opciones individuales
    - Cada uno es independiente del resto
    - Se utilizan cuando se pueden activar varias opciones relacionadas, pero no excluyentes
    - Aunque resulta redundante que ID y valor sea iguales, es obligatorio porque los atributos en XHTML no pueden tener valores vacíos



* + Radio
    - Mutuamente excluyentes
    - ’name’ para relacionarlos



* + Submit
    - Envíar el formulario al servidor
    - Si no hay atributo ‘value’, el navegador muestra el texto predefinido ‘Enviar consulta’



* + Image
    - Aspecto del botón cambiado por una imagen
    - Permite personalizar la estética de los botones
    - Ralentiza la carga



* + Reset
    - Antes era muy popular
    - Borra todos los datos
    - ‘value’ por defecto: Restablecer



* + Button
    - Uso genérico
    - Se le pueden asociar distintos eventos por script
    - Uso frecuente: comprobar coherencia y validez de los datos para versiones anteriores a HTML5



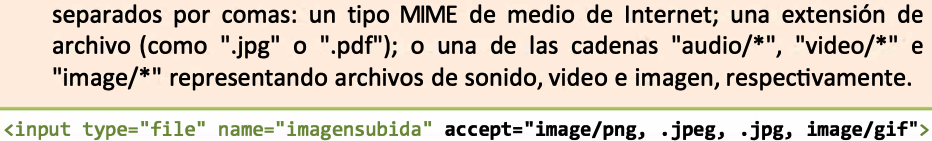
* + Hidden
    - No se muestra en pantalla
    - A veces es preciso enviar información sin que el usuario tenga que introducirla (núm sesión sin que sea visible por el usuario)



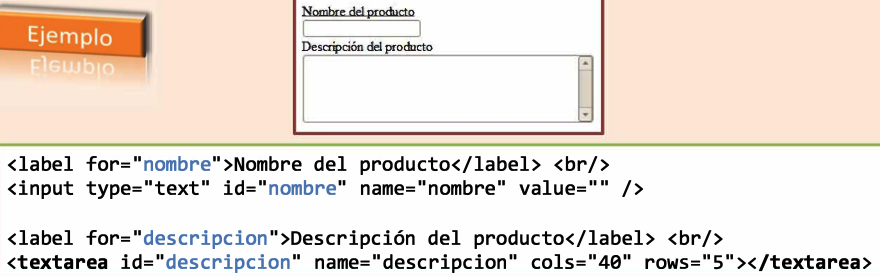
* + File
    - Obligatorio ‘enctype’ en el <form>: multiform/form-data:



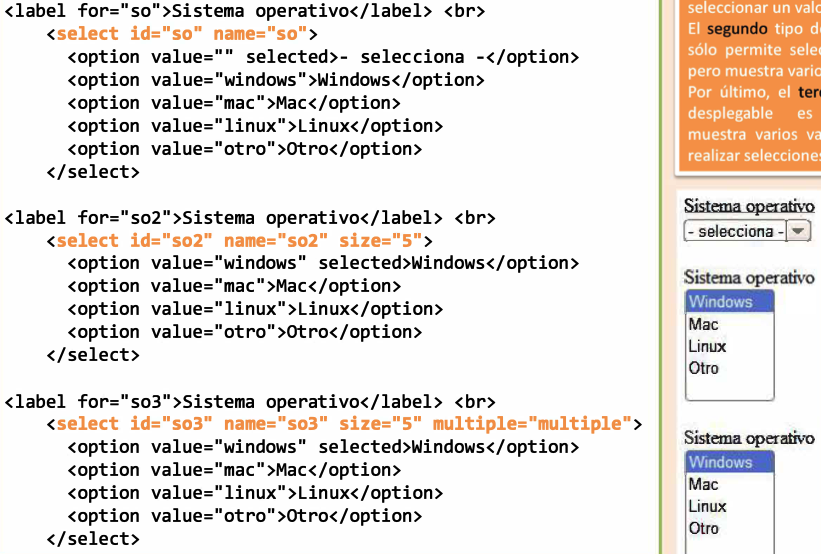
* + - Accept para filtrar los tipos de archivos:



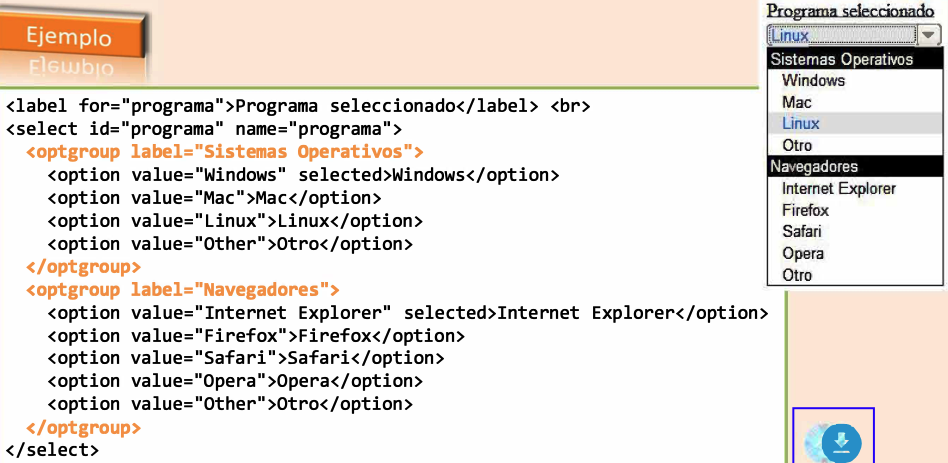
* Textarea
  + <textarea></textarea>
  + Caja de filas y columnas en las cuales se introduce texto
  + Campo de texto multilínea
  + Textos largos (descripciones, etc.)
  + No límite caracteres
  + Atributos:
    - ‘cols’: anchura en columnas visibles en número de caracteres
    - ‘rows’: = en altura



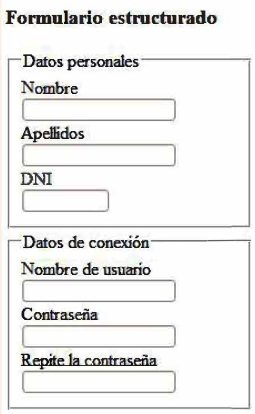
* Button
* Select
  + <select> … </select>
  + Con los <option> … </option> en su interior
  + Optcionalmente <optgroup> … </optgroup>
  + Todo ‘select’ al menos un ‘option’
  + ‘optgroup’ agrupa opciones
  + Atributos del ‘option’:
    - ‘selected’: preseleccionada inicial-
    - ‘value’: valor inicial de la opción a visalizarse
      * Si no se especifica, es el texto entre las etiquetas
      * Obligatorio, ya que es el dato que se envía al servidor
    - ‘label’: permite especificar un rótulo más corto para la opción



* + Atributos del ‘outgrup’: label



* Fieldset
  + <fieldset> … </fieldset>: agrupa campos del formulario
  + <legend> --- </legend>: nombre a cada grupo



* Label
  + <label></label>
  + Para establecer el título de cada campo del formulario
  + El HTML queda mejor estructurado y mejora su accesibilidad
    - Al pulsar el label, el puntero se posiciona para escribir su input



* Datalist
* Keygen
* Output