[Fecha]

JOSE ANTONIO URES

CampusFP

HITO INDIVIDUAL

LENGUAJE DE MARCAS

Contenido

[FASE 1: desarrollo del documento XML 2](#_Toc127902661)

[FASE 2: acceso y gestión de datos de un documento XML 5](#_Toc127902662)

[FASE 3: justificación del proyecto 6](#_Toc127902663)

[WEBGRAFIA 7](#_Toc127902664)

[FASE 1 7](#_Toc127902665)

# FASE 1: desarrollo del documento XML

El documento XML tiene la estructura que se muestra en la imagen:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE pedidos SYSTEM "./dtd.dtd">

<pedidos>

    <!--2021-->

    <!-- PRIMER TRIMESTRE -->

    <pedido>

        <numero>1</numero>

        <nombre>Juan</nombre>

        <apellido>Diaz</apellido>

        <telefono>611222333</telefono>

        <direccion>

            <calle>Calle mayor</calle>

            <ciudad>Toledo</ciudad>

            <codigo\_postal>45005</codigo\_postal>

            <provincia>Toledo</provincia>

        </direccion>

        <email>j.diaz@gmail.com</email>

        <fecha\_inclusion>

            <fecha>01/01/2021</fecha>

            <año>2021</año>

            <trimestre>1</trimestre>

        </fecha\_inclusion>

        <datosPedido>

            <id\_pedido>01</id\_pedido>

            <fecha\_compra>10/01/2021</fecha\_compra>

            <fecha\_entrega>15/01/2021</fecha\_entrega>

            <total\_factura>39.80€</total\_factura>

            <producto>

                <nombre\_producto>Camisa de rayas azules</nombre\_producto>

                <referencia>123456</referencia>

                <precio>19,90€</precio>

                <unidades>2</unidades>

            </producto>

        </datosPedido>

    </pedido>

Explicación:

**Primera línea**

Indica al programa que recibe un archivo de datos en XML, que se está escribiendo en XML y que debe corresponder el archivo a la especificación XML de la versión 1.0

**Segunda línea**

Indica mediante la etiqueta **DOCTYPE** la DTD que debe utilizar el procesador XML para validar el documento XML, puede estar incluida en el propio documento, ser un documento externo o combinarse ambas.

La palabra "**pedidos**", es este caso, es el nombre del tipo de documento XML, que debe coincidir con el nombre del elemento raíz del documento XML.

La palabra **SYSTEM** es un elemento que representa un identificador de sistema en un mensaje XML y después le indicamos la ruta donde se encuentra la DTD, en este caso es externa (./dtd.dtd).

La DTD tiene la estructura que se muestra en la siguiente imagen:

<!ELEMENT pedidos (pedido\*)>

    <!ELEMENT pedido (numero, nombre, apellido, telefono, direccion, email, fecha\_inclusion, datosPedido)>

        <!ELEMENT numero (#PCDATA)>

        <!ELEMENT nombre (#PCDATA)>

        <!ELEMENT apellido (#PCDATA)>

        <!ELEMENT telefono (#PCDATA)>

    <!ELEMENT direccion (calle, ciudad, codigo\_postal, provincia)>

        <!ELEMENT calle (#PCDATA)>

        <!ELEMENT ciudad (#PCDATA)>

        <!ELEMENT codigo\_postal (#PCDATA)>

        <!ELEMENT provincia (#PCDATA)>

        <!ELEMENT email (#PCDATA)>

    <!ELEMENT fecha\_inclusion (fecha,año,trimestre)>

        <!ELEMENT fecha (#PCDATA)>

        <!ELEMENT año (#PCDATA)>

        <!ELEMENT trimestre (#PCDATA)>

    <!ELEMENT datosPedido (id\_pedido, fecha\_compra, fecha\_entrega, total\_factura, producto)>

        <!ELEMENT id\_pedido (#PCDATA)>

        <!ELEMENT fecha\_compra (#PCDATA)>

        <!ELEMENT fecha\_entrega (#PCDATA)>

        <!ELEMENT total\_factura (#PCDATA)>

    <!ELEMENT producto (nombre\_producto, referencia, precio, unidades)>

        <!ELEMENT nombre\_producto (#PCDATA)>

        <!ELEMENT referencia (#PCDATA)>

        <!ELEMENT precio (#PCDATA)>

        <!ELEMENT unidades (#PCDATA)>

Explicación:

**Primera linea**

<!ELEMENT *nombre\_del\_elemento* (*modelo\_de\_contenido*)>

**ELEMENT** es una palabra reservada que indica que todo lo que se encuentre entre **<!** y **>** será la declaración de un elemento.

**nombre\_del\_elemento**: se podrá asignar libremente y hará referencia al elemento raíz, debiendo seguir la sintaxis de XML:

* Pueden contener letras, números y tres signos de puntuación: guion, guion bajo, punto y dos puntos (este último se reserva para indicar espacios de nombre).
* No pueden empezar por un numero ni signo de puntuación (excepto el guion bajo).
* No pueden empezar por las letras “xml” ni contener espacios en blanco.

**(modelo\_de\_contenido)**: define qué otros elementos pueden contener, en qué cantidad y en qué orden.

De las tres, la parte más importante es el modelo de contenido, así que vamos a verlo más detenidamente.

En el modelo de contenido lo que se define es la naturaleza de los hijos que acepta un elemento.

**#PCDATA** es la abreviatura de *parsed character data*, es decir, texto plano interpretado o analizado como tal.

**Segunda línea**

El nombre del elemento será “pedido” y tendrá los hijos que están entre paréntesis.

En las siguientes líneas se definirán los elementos hijos especificando el nombre y su contenido, en este caso, todos los elementos hijos tendrá como contenido PCDATA.

# FASE 2: acceso y gestión de datos de un documento XML

## Explicación del código: tabla de clientes y pedidos tienen la misma estructura.

* Abre un **SCRIPT** con la etiqueta **<script>**
* Mediante un condicional **IF** según el navegador crea un objeto nativo o un objeto ActiveX para navegadores obsoletos.
* Abre documento mediante el método **.OPEN** que admite tres parámetros, usamos **GET** para obtener información, la **URL** donde se encuentra el archivo, **FALSE** que es la forma de ejecutarse. Puede síncrona o asíncrona, en este caso FALSE se asigna a una petición asíncrona.
* El método **.SEND()** envía el objeto al servidor.
* Recupera el documento con estructura DOM a partir de un XML.
* Mediante el método **document.write** imprimimos texto en pantalla, en este caso abre una tabla (se cerrará al final) con una cabecera mediante las etiquetas <table> y <th></th>
* El método **.GETELEMENTSBYTAGNAME** busca elementos con la etiqueta dada y devuelve una colección con ellos.
* Mediante el método **document.write** muestra por pantalla la información recogida con la estructura de una tabla mediante las etiquetas <tr></tr> y <td></td>
* Cierra la tabla </table>
* Cierra el script </script>

## Explicación del código tabla factura:

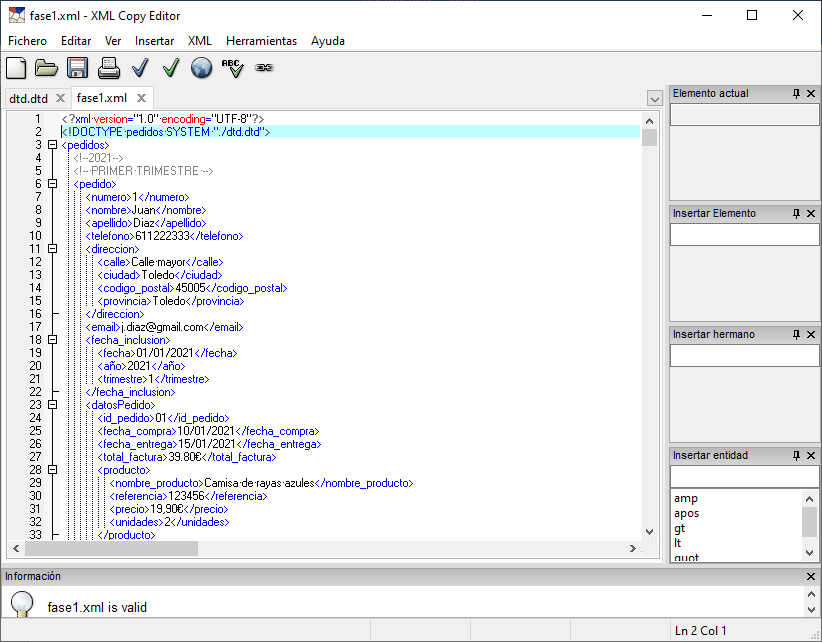
* Abre un **SCRIPT** con la etiqueta **<script>**
* Instancia un objeto.
* Abre el documento mediante el método **.OPEN** que admite tres parámetros, usamos **GET** para obtener información, la **URL** donde se encuentra el archivo, **TRUE** que es la forma de ejecutarse. Puede síncrona o asíncrona, en este caso **TRUE** se asigna a una petición síncrona.
* Con el método **addEventListener**(“load”, function(){}) ejecutara el evento load con una función.
* Instancia un objeto XMLHttpRequest().
* Abre el documento igual que anteriormente.
* El método onreadystatechange = function() almacena el nombre de la función que se ejecutará cuando el objeto XMLHttpRequest cambie de estado.
* Mediante un condicional IF ejecutará el código si this.readyState === 4 (requirimiento del servidor completado) y el método this.estatus === 200 (estado OK)
* Recuperará el documento con estructura DOM a partir de un archivo XML.
* Guardará el valor del elemento indicado en una constante.
* Accederá al elemento indicado del documento y guardará su valor en una constante.
* Iguala el elemento ubicado dentro del DOM a la longitud del elemento raíz y saldrá del condicional.
* Envía el objeto al servidor.
* Vuelve a almacena el nombre de la función que se ejecutará cuando el objeto XMLHttpRequest cambie de estado.
* Y repite el mismo condicional, pero en este caso, recorre las etiquetas del formulario y no el valor asignado en la etiqueta label y recorre las etiquetas del elemento indicado.
* Vuelve a entrar en otro condicional y comprueba si el valor introducido en la etiqueta label es menor que 1 o mayor que la longitud del elemento raíz – 1, mostrará todos los campos en blanco y mostrará un mensaje de error. En caso contario recorrerá todas las etiquetas indicadas del elemento raíz y las imprimirá por pantalla.

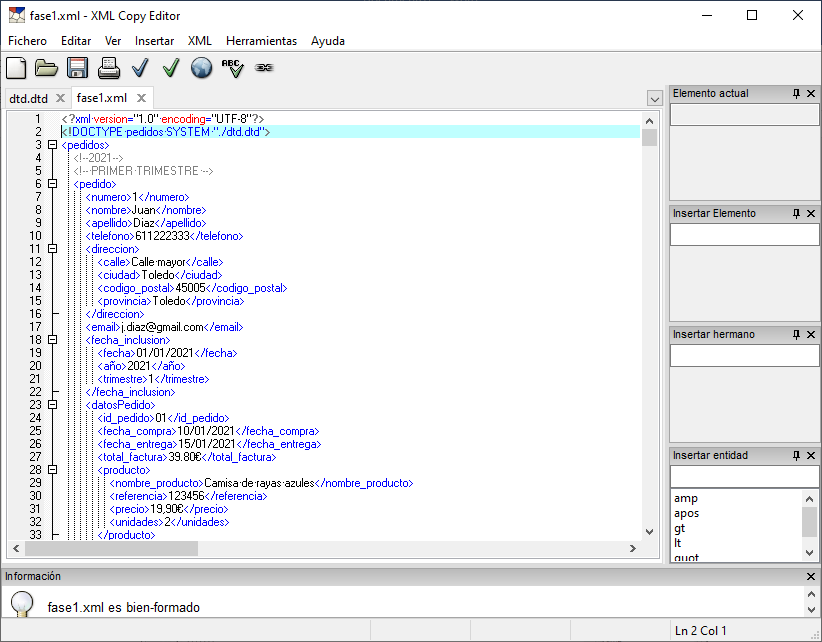
## Explicación del código tabla trimestres

* Abre un **SCRIPT** con la etiqueta **<script>**
* Entra en un condicional y crea un objeto dependiendo si es un navegador moderno u obsoleto.
* Abre el documento mediante el método **.OPEN** que admite tres parámetros, usamos **GET** para obtener información, la **URL** donde se encuentra el archivo, **FALSE** que es la forma de ejecutarse. Puede síncrona o asíncrona, en este caso **FALSE** se asigna a una petición asíncrona.
* Envía el objeto al servidor.
* Recupera el documento con estructura DOM a partir de un XML.
* Imprime por pantalla un texto especifico en la página.
* Busca elementos con la etiqueta dada y devuelve una colección con ellos.
* Imprime por pantalla un texto especifico en la página, en este caso será la cabecera de la tabla mediante la etiqueta <th></th>.
* Entra en un bucle que inicializa i=0, mientras i sea menor recorre todos los elementos indicados y en cada vuelta se incrementa i en 1.
* Con el método .**getElementsByTagName()** accede a las etiquetas mediante su nombre, crea un array con todas las etiquetas con el mismo nombre.
* Y entra en otro condicional donde imprimirá los elementos indicados si el elemento trimestre es menor que 2 mediante la funcion parseInt que analizara la cadena para determinar si contiene un valor entero y saldrá del condicional.
* Volverá a entrar en otro condicional y hará lo mismo si trimestre es mayor que 7 y saldrá
* Cerrará la tabla.

# FASE 3: justificación del proyecto

Validación del documento XML





Desde la aplicación “**XML Copy Editor**” abrimos el documento XML y vemos que esta bien formado y es válido.

# WEBGRAFIA

## FASE 1

<http://www.codexexempla.org/articulos/2008/leer_dtd_1.php>

## FASE 2

<https://es.javascript.info/xmlhttprequest>