Ejercicios semana de 27 de abril al 8 de mayo

DEPARTAMENTO DE TECNOLOGÍA

CURSO 2º Bach

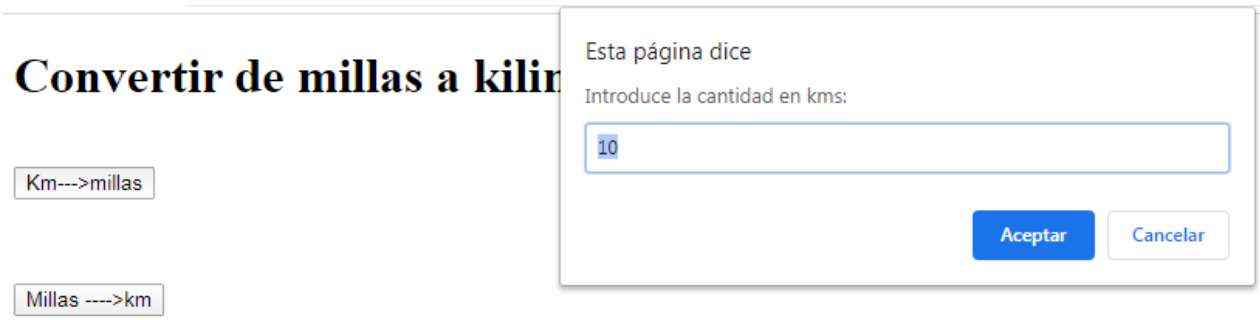
Materia Tecnología de la información y la comunicación:

GRUPO B2A	ALBERTO GONZALO GONZALO,	agonzalo@educa.jcyl.es
GRUPO B2B	Miguel SÁNCHEZ GALLEGO,	msanchezga@educa.jcyl.es
GRUPO B2C	ALBERTO GONZALO GONZALO,	agonzalo@educa.jcyl.es
GRUPO B2D	ALBERTO GONZALO GONZALO,	agonzalo@educa.jcyl.es
GRUPO B2E	ALBERTO GONZALO GONZALO,	agonzalo@educa.jcyl.es

1. Lee el texto y comprueba el funcionamiento de las funciones, crea al menos un ejemplo de cada caso.

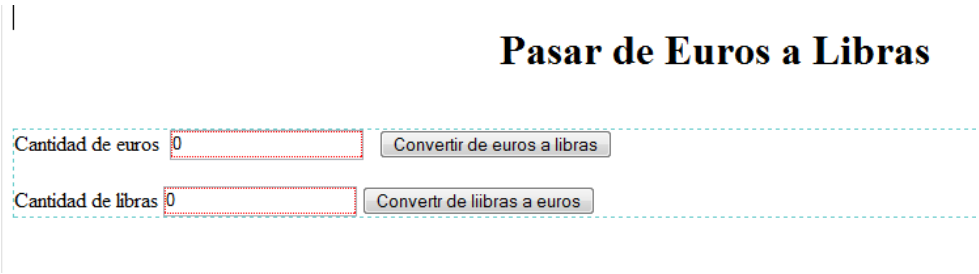
a) Funciones en el mismos script
b) Funciones en diferente script
c) Funciones con datos
d) Funciones con un botón
e) Otros elementos con las funciones.

2. **Crea un formulario y su función para pasar de millas a km y de km a millas.**



3.

4. **Crea un formulario y su función para pasar de euros a libras y de libras a euros.**



¿Qué es una función?

Una función es pequeño programa (subprograma) que realiza una o varias tareas determinadas.

- Conjunto de instrucciones englobadas en un mismo proceso: este proceso es una función y tiene un nombre.
- Pueden ser convocados por su nombre desde diferentes lugares. De modo que sirven para evitar la repetición del código

Sintaxis

Las funciones poseen la siguiente estructura:

Function nombre(parámetro_1, parámetro_2,....,parámetro_n) {
Código de tareas que realizará la función}

Se escribe la palabra function, luego el nombre de la función y entre paréntesis los parámetros, si no los tiene, igual se incorporan los paréntesis() .

Ejemplo:

function error(){ Window.alert("ERROR!!!!")
}

Formas de ejecutar una función:

Solo necesita invocar su nombre entre tags de script de la siguiente manera:
1. Luego de la función invocando su nombre:error()
2. En otra parte de la página
3. Desde un evento

Realice los siguientes ejemplos:

Desde el mismo script	Desde diferentes script

<script
type="text/javascript">
function alarma(){
alert(“ERRORRRR!")
}
alarma()
</script>
	<script type="text/javascript">
	function alarma()
	{ alert("ERRORRRR!")
	}
	</script>
	 veamos que pasa
	<script>
	alarma()
	</script>

Parámetros de las funciones

Las funciones tienen una entrada y una salida, que se pueden usar para recibir y devolver datos.
Los parámetros se usan para mandar valores a la función, con estos parámetros realizará acciones.
Entonces los parámetros:
Se usan para enviar valores a la función
Nota: Para incorporar un parámetro en la función, tenemos que poner el nombre de la variable que almacenará el dato que se ingresa, por otro lado la variable tendrá vida durante la ejecución de la función y dejará de existir cuando la función se termine de ejecutar. Los parámetros pueden recibir cualquier tipo de datos, textos, números, boléanos, u objetos.

Múltiples parámetros:
Se pueden ingresar más de un parámetro, separados por coma: ‘,’.

```
<script>
function alarma(nombre){
alert("Hola "+nombre)
}
</script>
</head>
<body>
<h1>veamos que pasa</h1>
<script>
v1 = prompt("Escriba su nombre","");
alarma(v1)
</script>
```

Retorno de la función:

Los parámetros de una función JAVASCRIPT se pasan por valor, o sea que si dentro de la función se cambia el valor de la variable, este cambio no se ve reflejado a menos que se utilice la palabra reservada return
La función realiza un conjunto de acciones y devuelve un valor.
Se usa la palabra return seguida por el valor que se desea retornar.
Cuando se la convoca se asigna a una variable.

```
<script type="text/javascript">

function suma(num) {
num++;
alert(num);
}
var a=1; d=suma(a); alert(d); alert(a)
</script>
```

```
<script type="text/javascript">
function suma(num) {
num++; alert(num); return num;
}
var a=1; d=suma(a); alert(d);
</script>
```

Múltiples retornos:
Dentro de una función se pueden usar más de un return
Lógicamente sólo habrá un valor que cumple con las condiciones de la función y ese es el valor que retornará.

Eventos

En este modelo, cada elemento o etiqueta XHTML define su propia lista de posibles eventos que se le pueden asignar. Un mismo tipo de evento (por ejemplo, pinchar el botón izquierdo del ratón) puede estar definido para varios elementos XHTML diferentes y un mismo elemento XHTML puede tener asociados varios eventos diferentes.

Eventos1.html

```
<html>
   <head>
      <meta charset="utf-8">
      <title>Detectar Mouse</title>
      <style>
         img{
            margin: 3% 5%;
         }
      </style>
   </head>
<body>

   <img src="mouseOver.jpg"
onmouseover="MostrarMouseOver()">
```

mouseEvents.js

```
function MostrarMouseOver(){
alert("Estoy pasando el mouse sobre la imagen");
}
function Mostrarclic(){

                alert("He hecho clic sobre la imagen");
}
```

<img src="click.png" onclick="Mostrarclic()"> <script src="mouseEvents.js"></script> </body> </html>	

Evento	Descripción	Elementos para los que está definido
onclick	Pinchar y soltar el ratón	Todos los elementos
ondblclick	Pinchar dos veces seguidas con el ratón	Todos los elementos
onfocus	Seleccionar un elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una tecla (sin soltar)	Elementos de formulario y <body>
onkeypress	Pulsar una tecla	Elementos de formulario y <body>
onload	La página se ha cargado completamente	<body>
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón *"sale"* del elemento (pasa por encima de otro elemento)	Todos los elementos
onmouseover	El ratón *"entra"* en el elemento (pasa por encima del elemento)	Todos los elementos

Formulario.

¿Qué es un formulario?

Los formularios en HTML sirven al propósito de recolectar información proporcionada por los visitantes del sitio, la cual es luego enviada nuevamente al servidor.
Para su correcto funcionamiento es importante que el formulario provisto en HTML sea acompañado de un código del lado servidor, al que denominaremos "**agente procesador",** que se encargará de recibir y procesar la información como el autor vea conveniente.

Este procesamiento puede consistir en, por ejemplo, el almacenamiento de la información o su envío mediante correo electrónico.

El siguiente código muestra la estructura de un formulario, con sus etiquetas de apertura y cierre encerrando un conjunto de controles.

```
<form>
  [Conjunto de controles]
</form>
```

Elementos de entrada INPUT

El elemento `input` representa un campo de datos tipado que normalmente se asocia con un control que permite a los usuarios editar su valor. Este elemento es capaz de proveer muchos tipos diferentes de campos, de acuerdo al valor que presente en el atributo `type`.

- input type=text entrada de texto

```
<p>Nombre:<input type="text" name="nombre" ></p>
    <p>Email:<input type="text" name="email"></p>
```

- Contraseña: <input type="password" name ="contraseña">
- input type=password oculta los caracteres.

- **Required** La versión Html5 ha añadido este nuevo atributo, required que nos permite comprobar que el campo ha sido rellenado antes incluso de pulsar ese botón de envío, sin necesidad de más complicaciones o código extra.
- **Placeholder** Mediante esta propiedad del Html5 podemos definir el texto que queremos que aparezca dentro del campo del formulario a modo de ayuda para las visitas. Por ejemplo, se suele colocar un texto explicativo acerca de la información que se debe rellenar en ese campo.

```
<p>Nombre:<input type="text" name="nombre" required></p>
    <p>Email:<input type="text" name="email"
    placeholder="Introduzca su email"></p>
    <p>Contraseña: <input type="password" name ="contraseña"></p>
<input type="submit" value="Enviar el formulario">
```

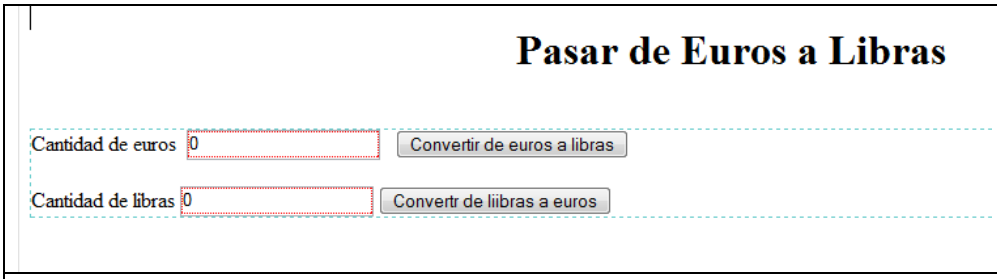
- input type=date fecha
- input type=color seleccionamos un color.
- input type=checkbox casilla de verificación
- input type=radio botón de opción
- input type=file fichero adjunto
- input type=submit **Botón de envío del formulario**
- input type=image muestra imagen de envío

- input type=reset borra el formulario.
- input type=button botón genérico

- **1.- name**: identifica un control dentro de un formulario. Este atributo nos va a resultar necesario para poder rescatar la información.
- **2.- value**: cada control tiene un valor inicial y un valor actual. Normalmente, el valor inicial de un control puede especificarse con el atributo value. El valor actual del control se hace en primer lugar igual al valor inicial. A partir de ese momento, el valor actual del control puede ser modificado mediante la acción del usuario. El valor inicial de un control no cambia. Así, cuando se carga el formulario, el valor actual de cada control se restablece a su valor inicial. Si el control no tiene un valor inicial, el efecto de una reinicialización o carga del formulario sobre ese control es indefinido. Indefinido significa que no sabemos exactamente lo que va a pasar. Para que no ocurra esto, es preferible establecer siempre un valor inicial.
- **3.- id:** el valor de este atributo permite relacionar un control con una etiqueta. Por ejemplo, si un control tiene por id el valor id="email", esto significa que ese control está relacionado con la etiqueta (label) cuyo atributo for es for="email".
- Otra útil propiedad que podemos colocar dentro del código de la etiquet input es **placeholder**="bla bla bla". Mediante esta propiedad del Html5 podemos definir el texto que queremos que aparezca dentro del campo del formulario a modo de ayuda para las visitas.
- La etiqueta **<fieldset>** agrupa campos del formulario y la etiqueta <legend> asigna un nombre a cada grupo.
- **Required** La versión Html5 ha añadido este nuevo atributo, required que nos permite comprobar que el campo ha sido rellenado antes incluso de pulsar ese botón de envío, sin necesidad de más complicaciones o código extra.
- El elemento `textarea` representa un campo para la entrada de texto multilínea. El control asociado a este campo es una caja de texto que permite a los usuarios editar múltiples líneas de texto regular. Los controles `textarea` son útiles para recolectar o editar líneas largas de texto como mensajes, contenido de archivos, listas, reseñas, artículos, etc.

http://librosweb.es/libro/xhtml/capitulo_8.html

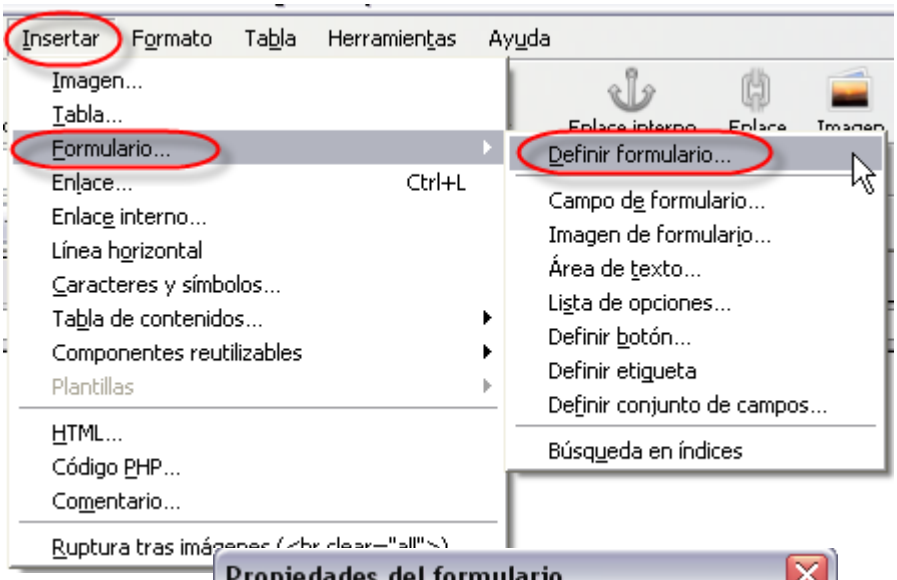
Vamos a crear el siguiente formulario.



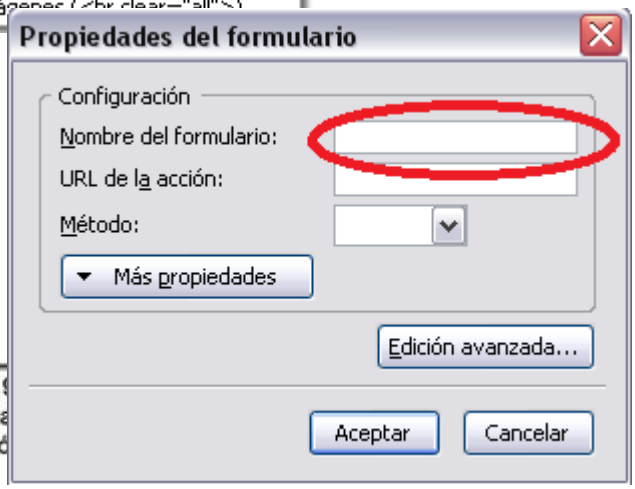

```
<body>
<h1 style="text-align: center;">Pasar de Euros a Libras</h1>
<br>
<form method="get" action="form" name="convertir_eu_li"> Cantidad de
euros  <input name="euros" value="0">   <input
onclick="li()" name="boton_euros" value="Convertir de euros a libras"
type="button"><br>
<br>
Cantidad de libras <input onclick="li()" name="libras" value="0"> <input
onclick="eu()" name="boton_libras" value="Convertr de liibras a euros"
type="button"><br>
</form>
</body>
```

CREACIÓN DE FORMULARIOS

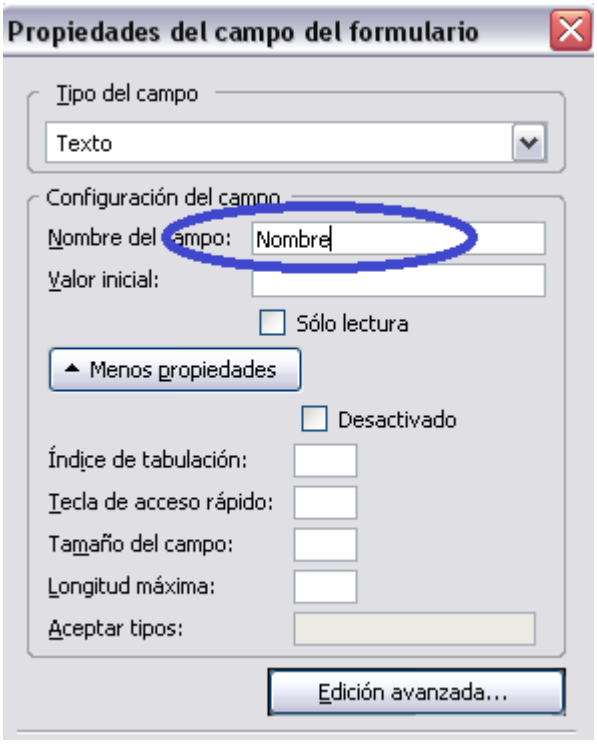
Para crear formularios con KompoZer, abriremos el programa e iremos a la opción **Insertar** --> **Formulario** --> **Definir formulario**.



Entonces aparece la ventana **Propiedades del**



formulario



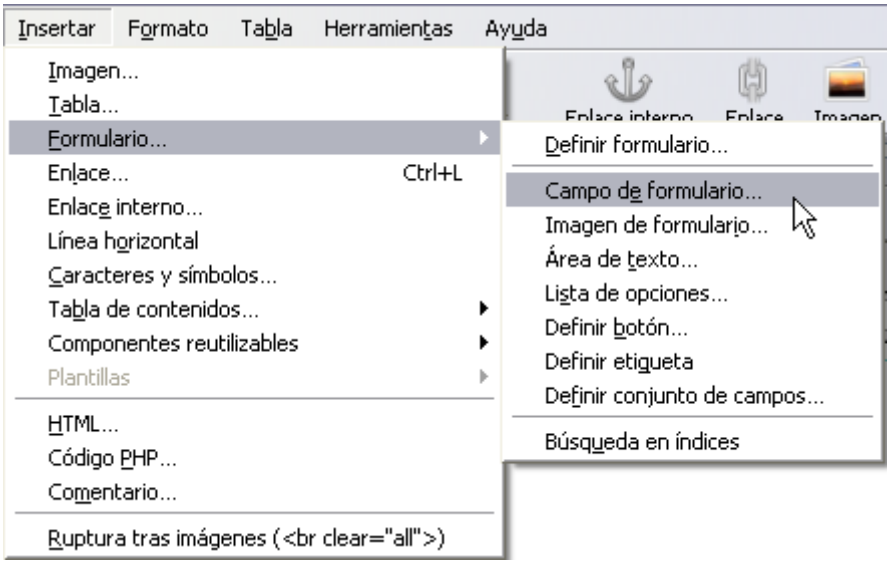
CAMPO DE TEXTO:

Vamos a insertar un campo en el que el visitante ponga su nombre, para ello:

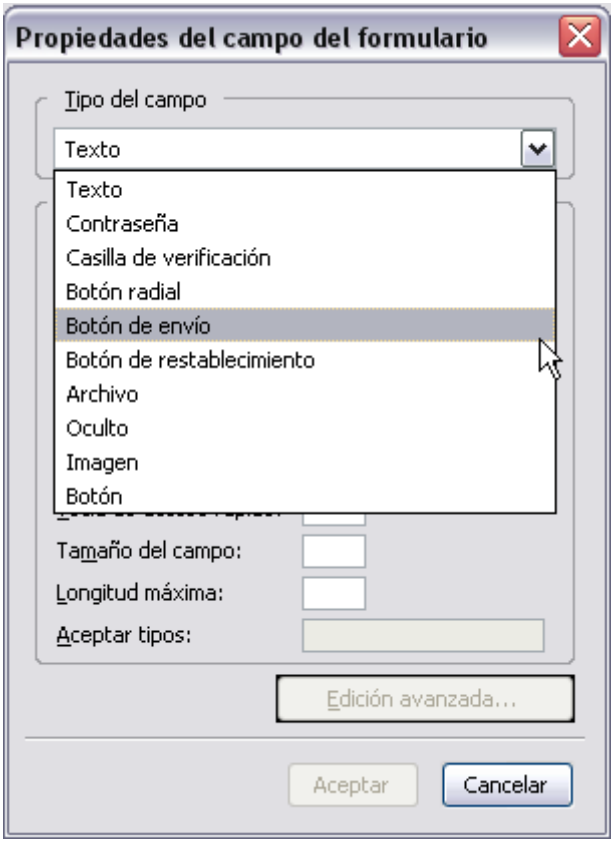
- Primero hacemos clic con el ratón dentro de ese área punteada en celeste y ponemos la palabra **Nombre**.

- A continuación, volvemos a la barra de herramientas **Insertar** --> **Formulario** --> **Campo de Formulario.** Rellenaremos la ventana que nos salga con los siguientes datos: elegiremos un **campo de texto**, lo nombraremos a efectos de poder reconocerlo posteriormente en el formulario y pulsaremos **aceptar**.
Nuestro formulario irá adquiriendo esta apariencia.

Nombre:



BOTONES DE ENVÍO Y RESTABLECIMIENTO:



Finalmente solo quedará insertar los botones que harán posible que el visitante envíe el formulario o limpie la información del mismo por si se arrepiente de enviarlo o por si se equivoca en la consignación de algún dato. Para insertar los botones haremos lo siguiente: **Insertar** --> **Formulario** --> **Campo de Formulario**; seleccionaremos la opción **botón de envío** y botón de restablecimiento posteriormente, de forma que completemos el formulario. A cada uno de los botones pondremos sus propiedades.

Obtener y escribir datos de un formulario.

Lo normal cuando el usuario rellena el formulario es que nosotros queramos acceder a los datos que él ha rellenado.
Podemos acceder a los datos del texto que introduce el usuario en los campos de texto <input type="text" ...>, <input type="password" ...>y <textarea>...</textarea> mediante la propiedad value aplicada a estos elementos..

Tengamos por ejemplo el siguiente formulario:

```
<form name="rellenar" >
    <p><input name="ver" type="button" value="ver" />
</form>
```

Leer un dato:

```
Dato = document.rellenar.ver.value
```

Escribir un dato:

```
document.rellenar.ver.value = “dato que escribimos”
```