**4.3 Tratamiento De Formularios**

Los Formularios Son Una Característica Del Estándar Html Que Permite A Los Autores Llevar Datos Del Exterior Al Interior De Nuestro Sistema. Esto Se Logra Mediante Componentes Que Transportan Diversos Tipos De Datos Para Múltiples Aplicaciones Donde Tu Imaginación Es El Límite

Los Formularios Están Delimitados Con La Etiqueta <Form>**…**</Form>, Que Permite Reunir Varios Elementos De Formulario, Como Botones Y Casillas De Texto Y Que Debe Poseer Los Siguientes Atributos:

* **Method** Indica Cómo Se Enviarán Las Respuestas “Post” Es El Valor Que Envía Los Datos Al Agente De Procesamiento Almacenándolos En El Cuerpo Del Formulario, En Tanto Que “Get” Envía Los Datos Agregándolos A La Dirección Url Y Separándolos De La Dirección Con Un Signo De Interrogación.
* **Action** Indica La Dirección A La Que Se Enviará La Información (Una Url O Dirección De Correo Electrónico: Mailto:Dirección\_De\_Correo@Equipo)
* **Target** Nos Permite Mostrar El Resultado En La Ventana Deseada:
  + Target = ‘\_Blank’ Fuerza Que El Documento Referenciado Por El Enlace Sea Mostrado En Una Nueva Ventana Del Navegador.
  + Target = ‘\_Self’ El Documento Referenciado Se Mostrara En La Misma Ventana

La Sintaxis Genérica Para La Etiqueta **Form:**

<Form Id='Nombre' Name='Nombre' Method='Get/Post' Action='Url'

Target = '\_Blank/\_Self' Enctype = 'X-Www-Form-Urlencoded'>

...Cuerpo Del Formulario

</Form>

La Etiqueta *Form* Actúa Como Una Especie De Contenedor Para Almacenar Elementos Que Permiten Al Usuario Seleccionar O Introducir Datos. Todos Los Datos Se Enviarán A La Dirección Url Indicada En El Atributo *Action* De La Etiqueta *Form*, Por El Método Indicado En El Atributo *Method*.

**4.4 Manejo De Objetos Del Servidor**

La  Programación  Orientada  A Objetos  Se Ha  Popularizado Enormemente En  Los  Últimos  Días, Y  Ya Resulta Difícil  Encontrar Lenguajes  De  Programación  Que No  Tengan Una Versión  O Que No  Estén Basados  En Esta Filosofía. Javascript No  Se Queda Al  Margen De  Esta Tendencia, Y  Aunque  No Permite Dos De Las Características Clásicas De Los Lenguajes Orientados A Objetos Ni La Herencia Ni El Polimorfismo, Sí Permite La Creación Y Manipulación De Objetos Sencil Os, Y La Definición De Métodos Y Propiedades Para Dichos Objetos. El Dom Es Una Jerarquía De Objetos Predefinidos Que Describen Los Elementos De La Página  *Web* Que Está Mostrando El Navegador, Así Como Otras Características Del  Proceso  De  Navegación Como Son  El  Historial, El  Tamaño De La Ventana  De  Navegación O El Contenido De La Barra De Estado Del Navegador. Un Objeto Es, En El Fondo, Un Conjunto De Variables Y Funciones Que Actúa Sobre Dichas Variables, Encapsuladas En Un Mismo Paquete. El Acceso A Las Funciones Y A Las Variables Se Realiza Mediante Una Interfaz Bien Definida Que Aísla Al Programador De  La Necesidad  De  Conocer Cómo Están  Implementadas  Internamente Dichas  Funciones. De Este Modo, La Programación  Orientada  A Objetos  Resulta Muy  Intuitiva, Y  Más  Próxima  Al  Conocimiento Humano. Como Vemos  En  El  Siguiente  Ejemplo De  Javascript,  Para Escribir  Un  Mensaje En  Un  Cuadro De Diálogo Utilizamos:

*Window.Alert(“¡Hola Mundo!”)*

Si Bien No Conocemos Como Funciona Internamente La Función Alert(), Sabemos Cómo Invocarla. La Abstracción Es Tal Que Nos Basta Con Saber Que Se Trata De Una Función Del Objeto Window. A Estas Funciones Se Las Llama Métodos, Y A Las Variables Propiedades.

**Objeto  Window**: Es  El  Objeto  Principal. Define La Ventana Sobre La Que  Estamos  Trabajando  E Incluye Los Objetos Referentes A La Barra De Tareas, El Documento O La Secuencia De Direcciones De La Última Sesión. Aún Cuando El Objeto Se Llame Window, Disponemos Siempre De Una Referencia A Él  Llamada Window, Y  Todas  Las  Propiedades  Y  Métodos  Que L Amamos  Sin Utilizar Ninguna Referencia, En Realidad Se Hacen Utilizando Esa Referencia Window. Este Hecho  Es Común A Todos Los Objetos Del Modelo.  Los Métodos Más Comunes Del Objeto Son Los Siguientes:

1.   **Open**(Url, Nombre, Propiedades).Permite Crear  Y  Abrir  Una Nueva  Ventana. Si  Queremos Tener Acceso A Ella Desde La Ventana Donde La Creamos Deberemos Asignarle Una Variable, Si  No  Simplemente Invocamos  El  Método: El  Navegador Automáticamente Sabrá Que Pertenece Al Objeto Window. El Parámetro Url Es Una Cadena Que Contendrá La Dirección De  La Ventana Que Estamos  Abriendo: Si  Está En  Blanco, La Ventana  Se Abrirá Con  Una Página En Blanco. El Nombre Será El Que Queramos Que Se Utilice Como Parámetro De Un Target Y Las Propiedades Son Una Lista Separada Por Comas De Algunos De Los Siguientes Elementos:

* Toolbar[=Yes|No]
* Location[=Yes|No]
* Directories[=Yes|No]
* Status[=Yes|No]
* Menubar[=Yes|No]
* Scrollbars[=Yes|No]
* Resizable[=Yes|No]
* Width=Pixels
* Height=Pixels

2.   **Close**() . Cierra La Ventana, Esta Mostrará Al Usuario Una Ventana De Confirmación Para Que Decida Él  Si  Quiere O  No  Cerrarla,  Esto  Se  Hace  Por  Motivos  De  Seguridad.

3.   **Alert**(Mensaje).Muestra Una Ventana De Diálogo Con El Mensaje Especificado.

4.     **Confirm**(Mensaje).Muestra Una Ventana  De  Diálogo Con  El  Mensaje Especificado Y  Dos Botones. Si El Usuario Pulsa Ok, El Método Devuelve True. Si, En Cambio, Pulsa Cancelar, Devolverá False.

5.   **Prompt**(Mensaje, Sugerencia). Muestra Una  Ventana De  Diálogo  Con El  Mensaje Especificado Y Un Campo De Texto En El Que El Usuario Pueda Teclear, Cuyo Valor Inicial Será Igual A Sugerencia. Si El Usuario Pulsa Ok, El Método Devuelve La Cadena Introducida En El Campo De Texto. Si, Por El Contrario, Pulsa Cancelar, Devolverá El Valor Null. Dado Que Este Valor Se Considera Igual A False, Podemos Comprobarlo Directamente En Una Condición Para Ver Que Ha Hecho El Usuario.

Por Ejemplo, El Siguiente Código Muestra Un Saludo Sólo Si El Usuario Ha Pulsado El Botón De Aceptar:

*Var Contestacion = Prompt(“¿Cómo Te Llamas?”, “”);*

*If (Contestacion)*

*Alert(“Hola, ” + Contestacion);*

6.   **Status**.Define La Cadena De Caracteres  Que Saldrá En La Barra De  Estado  En  Un Momento Dado.

7.     **Defaultstatus**. Define  La Cadena De Caracteres  Que Saldrá Por  Defecto En  La Barra De Estado. Cuando No  La Especificamos, Defaultstatus  Será Igual  Al  Último Valor Que Tomó Status.

8.     **Settimeout**(“Función”,Tiempo).  Llama      A     Función  Cuando          Hayan  Pasado  Tiempo Milisegundos. Imprescindible A La Hora De Realizar Cualquier Rutina Que Deba Ejecutarse A Cierta Velocidad.

**Objeto  Frame**: Para Acceder  Al  Código  O A Objetos  Como Window  O Document De  Otros  Frames, Javascript Utiliza  El  Objeto  Frame  Con  Las  Etiquetas  Frame  Y  Frameset, Este Objeto Permite Acceder A Los  Frames  Que Hemos  Declarado En Él  Por Medio Del  Arreglo  Frames. Es  Decir, Si  En Nuestro Documento Estuviera La Siguiente Línea:

*<Frame Name=”Principal” Src=”Mipagina.Html”>*

Podríamos Acceder A Su Objeto Window Por Medio De La Referencia *Window.Frames[“Principal”]*, A Su Vez, Desde El  Documento “Hijo”, Es  Decir, Desde  El  Documento Que Está Encerrado En  Un  Marco, Podemos  Acceder Al  Padre Por  Medio  De  La Referencia Parent. También  Podemos  Acceder Al Documento Que Esté Arriba Del Todo En Esta Jerarquía Por Medio De Top. Por Ejemplo:

*Window == Window.Top*

Esta Igualdad Comprobará Si Nuestro Documento Está En La Ventana Principal O En Un Frame.

**Objeto Document.** Este Objeto Representa El Documento Html En El Que Estamos. Se Accede A Él Por Medio De La Referencia Document. Su Mayor Importancia Viene Por El Número De  Arreglos Que Posee, Que Referencian A Objetos Image, Form O Link, Los Cuales Permiten Acceder A Las Imágenes, Formularios Y Enlaces Del Documento, Respectivamente.  Entre Sus Métodos Y Propiedades Más Importantes Se Encuentran Los Siguientes:

1.   **Lastmodified**. Contiene La Fecha Y Hora En Que Se Modificó El Documento Por Última Vez Y Se Suele Usar En  Conjunción  Con Write Para Añadir  Al  Final  Del  Documento Estas Características.

2.     **Bicolor**. Modifica El  Color  De  Fondo Del  Documento.  El  Color  Deberá  Estar En  El  Formato Usado En Html. Es Decir, Puede Ser Red O Ff0000.

3.   **Forms**[]. Arreglo Que Contiene Los Formularios Del Documento. El Primero Será El Número 0, El Segundo El 1, Etc.

4.   **Images**[]. Arreglo Que Contiene Las Imágenes Del Documento. Se Ordenan Igual Que En El Anterior Caso, Aunque  También Permiten  Ser Accedidas  Con  El  Nombre Como Índice.  Es Decir, A Una Imagen Definida Como <Img Src=”..” Name=”Miimagen”> Se Puede Acceder Con Document.Inages[“Miimagen”].

5.     **Links**[]. Arreglo Que Contiene Los Enlaces Del Documento. Se Ordenan Igual Que En Los Dos Anteriores, Aunque No Se Suele Utilizar En El Código Javascript. Su Razón De Ser Es Que, Al Ser Los Enlaces Objetos, Permiten Incluir Código Javascript En Ellos Por Medio De La Pseudo-Url Javascript:Codigo.

6.   **Write**(Cadena) Writeln(Cadena). Escribe  El  Código  Html  Indicado En Cadena En Nuestro Documento Html. Writeln Hace Lo Mismo, Pero Incluyendo Al Final Un Retorno De Carro.

7.   **Open**(). Abre Un  Nuevo Documento Para Escribir. Un  Documento Debe  Estar Abierto Para Poder Escribir En Él. Sin Embargo, No Se Utiliza Mucho Ya Que Los Dos Métodos Anteriores Abren Automáticamente El Documento Si No Lo Está Ya.

8.   **Close**(). Cierra El Documento, Impidiendo Escribir De Nuevo En Él.

**Objeto Image**. Este Objeto Representa A Una Imagen. Se Puede Acceder A Las Diversas Imágenes Del Documento Por Medio Del Arreglo De Referencias Document.Images[]. Sus Propiedades Más Importantes Son:

1.  Src. Contiene El Archivo De La Imagen.

2.  Complete. Valor Lógico Que  Será True  Si  La Imagen  Ha  Terminado  Ya De  Cargarse.

También  Dispone De Diversas  Propiedades  De Sólo  Lectura Que  Se  Corresponden  Con  Los Atributos De La Etiqueta <Img> Como, Por Ejemplo, Width O Border.

**Objeto Form**. Cada Forma En Un Documento Crea Un Objeto Form, Porque Un Documento Puede Contener Más De Una Forma. Los Objetos Forma Son Almacenados En Un Arreglo Llamado Forms. La Primer Forma En La Página Es Forms[0] Y Así Sucesivamente. Los Métodos Y Propiedades Más Importantes Del Objeto Form Son Los Siguientes:

1.   **Submit**(). Envía El Formulario.

2.   **Reset**(). Devuelve Los Valores De Un Formulario A Su Estado Inicial.

3.   **Elements**[]. Contiene Todos  Y  Cada  Uno De Los  Elementos  De  Los  Que Consta El Formulario, Es  Decir, Los  Botones, Cajas  De  Textos, Listas  Desplegables, Etc.  Que Componen Un  Formulario. Cada  Elemento Puede  Ser  Un  Objeto  Distinto, Por  Lo Que Deberemos Averiguar De Qué Tipo Son Por Medio De La Propiedad Común Type.

**Objetos Text**. Cuatro Elementos  Html  Distintos  (Text, Textarea, Password Y  Hidden) Son, Objetos Muy Parecidos Entre Sí. Tienen Tres Propiedades Verdaderamente Importantes:

1.   **Name**. Nombre Del Elemento, Indicado En El Atributo Name De Su Etiqueta Html. Este Atributo Está Presente En Todos Los Objetos Que Son Elementos De Un Formulario.

2.   **Type**. Tipo Del Elemento. Al Igual Que El Anterior, Esta Propiedad Está Presente En Todos Los Objetos Que Representan Elementos De Un Formulario. En Este Caso Valdrá Siempre “Text”.

3.   **Value**. Contiene El Valor, Es Decir, El Texto Tecleado Por El Usuario.

**Objetos Checkbox Y Radio**. En Estos Objetos El Estado Del Elemento Es De Tipo Lógico: Puede Estar Seleccionado O No. Este Valor Lógico Se Contiene En La Propiedad Checked

**Objetos Select Y  Option**. El  Objeto  Select Es  El  Más  Complicado. Esto Es  Debido  A Que Contiene En Su Interior Un Arreglo De Objetos Option. Dispone De Dos Propiedades:

1.   **Selectedindex**. Empezando  A Contar  A  Partir  De  Cero, Indica Qué  Opción Está Seleccionada. Si Hay Varias Seleccionadas, Indica La Opción Con El Índice Más Bajo.

2.   **Options[].**  Arreglo  Que Contiene  Los  Objetos  Option  Correspondientes  A Todas  Y  Cada Una De Las Opciones.  El Objeto Option, Por Otro Lado, Dispone De Otras Dos Propiedades:

1.   **Selected**. Valor Lógico Que Será Verdadero Si La Opción Está Seleccionada.

2.   **Text**. Texto Que Acompaña A La Opción.  Otros  Objetos. El  Modelo De  Objetos  Del  Documento Define  Varios  Objetos, Por  Así Decirlo, “Menores”, Que No Tienen Relación Con Nada Físico De La Página En La Que Estamos. Es Decir, No Guardan Relación Con Las Etiquetas Html Que Estén En Ellas Escritas. Son Los Siguientes:

**Objeto  History**. Se Accede A  Este  Objeto Por  Medio  De  La Referencia Document.History  Y Contiene  Todas  Las  Direcciones  Que Se Han Visitado  En  La Sesión  Actual. Aunque No Permite Acceder A Él, Dispone De Varios Métodos Para Sustituir El Documento Actual Por Alguno Que El Usuario Ya Haya Visitado:

1.   **Back**(). Volver A La Página Anterior, La Sintaxis Para Utilizarlo Es La Siguiente: Window.History.Back();

2.   **Forward**(). Ir A La Página Siguiente.

3.   **Go**(Donde). Ir  A Donde  Se Indique, Siendo Donde  Un  Número Tal  Que Go(1)=Forward()  Y Go(-1)=Back().

**Objeto  Location.**  Se Accede A Este Objeto Por  Medio De  La Referencia Document.Location Y Contiene Información Sobre La Dirección De La Página Actual En Varias Propiedades.

1.     **Href**.  Permite El  Acceso  A La Dirección De La Página  Actual. Si  Lo  Cambiamos Cambiaremos De Página.

2.   **Protocol**. Protocolo De La Página Actual. Habitualmente Http.

3.   **Host**. Máquina Donde Se Alberga La Página Actual.

4.   **Pathame**. Camino De La Página Actual

5.   **Search**. Esta Propiedad Permite Acceder A La Última Parte De La Dirección, A Partir De La Interrogación,  Que  Son Los  Parámetros. Puede  Ser Útil  Para Pasar Parámetros  De  Una Página A Otra.

**Objeto  Screen**.  Se Puede Acceder A Este Objeto  Por  Medio De La Referencia  Screen.  Nos Permitirá Conocer La Configuración  De La Pantalla Del  Usuario, Todos  Sus  Atributos  Son  De  Sólo Lectura.

1.   **Height**. Altura De La Resolución De La Pantalla.

2.   **Width**. Anchura De La Resolución De La Pantalla.

3.   **Pixeldepth**. Número De Bits Por Pixel.