El futuro digital es de todos

MinTIC

UNIVERSIDAD DEL NORTE

Vigilada Mineducación

CICLO III:

Desarrollo de software

MisionTIC2022

El futuro digital es de todos

MinTIC

UNIVERSIDAD DEL NORTE

Vigilada Mineducación

Sesión 7:

Desarrollo Software

Tecnologías y lenguajes para el desarrollo del Front-end

MisiónTIC2022

Objetivos de la sesión

Al finalizar esta sesión estarás en capacidad de:

1. Construcción básica de páginas web dinámicas con JavaScript.
2. Creación de scripts con Javascript.
3. Manejo de eventos con javascript.

El futuro digital es de todos | MinTIC

UNIVERSIDAD DEL NORTE
Vigilada Mineducación

Desarrollo Web – FrontEnd
JavaScript: Objetos

- Un objeto es un conjunto de propiedades y funciones (llamadas métodos).
- **Ejemplo:** Una persona tiene unas características o propiedades: cabello largo, altura, color de ojos, peso,... Además de propiedades una persona también tiene un comportamiento, ejecuta ciertas acciones tales como comer, dormir,... El conjunto de propiedades y acciones/funciones definen un objeto.
- En JavaScript existen una serie de objetos ya definidos, con sus propiedades y funciones (métodos). Uno de los objetos más importantes es el objeto Window, que contiene a su vez otros objetos, estructurados en forma de árbol.

Mision TIC 2022

Desarrollo Web – FrontEnd
JavaScript: Objetos

- Para utilizar las propiedades y métodos de un objeto se debe colocar el nombre del objeto seguido de un punto y la propiedad o el método a continuación.

- Ejemplo:

  window.status=“mensaje”;

- Los métodos son funciones que se pueden ejecutar y las propiedades son simplemente variables que se pueden asignar o leer.

Desarrollo Web – FrontEnd
JavaScript: Objetos

Objeto window:

- La propiedad status del objeto window indica el mensaje que aparece en la barra de estado.
- El método open abre una ventana nueva con las propiedades que se le definan. La sintaxis es: ventana=open(“URL”,”nombre”,”opciones”) donde opciones se refiere una lista separada por comas de las siguientes propiedades: toolbar, location, directories, status, menubar, scrollbars, resizable, width y height.
- **Ejemplo**: ventana=open(“URL”,”nombre”,”toolbar=no,menubar=no,status=yes,width=200,height=200,resizable=yes);

Desarrollo Web – FrontEnd
JavaScript: Objetos

Otros métodos del objeto window:

- alert
- Prompt
- Confirm
- Close
- SetPrint()
- SetInterval()
- SetTimeout()

```
<script>
var edad=prompt("Indica tu edad");
while(!confirm("Tu edad es: "+edad)) {
    edad=prompt("Indica tu edad");
}
if(edad < 18){
    alert("Eres menor de edad, aún no puedes ir a la disco");
}
else{
    alert("Eres mayor de edad, ya puedes ir a la disco");
}
setTimeout("cerrar()",5000);
document.write("Esta página se autodestruirá en 5segundos (solo si
tienes explorer)");
function cerrar(){
 window.close();
}
</script>
<br><a href="javascript:window.print()">Imprimir esta página</a>
```

Desarrollo Web – FrontEnd
JavaScript: Objetos

Objeto navigator:

Este objeto permite obtener datos del navegador utilizado.

- appName: facilita el nombre del navegador utilizado.
- Languaje: facilita el lenguaje del navegador utilizado.

Ejemplo:

```
alert("Navegador: "+navigator.appName);
alert(“Idioma : "+navigator.language);
```

Estas propiedades son solo de lectura.

Desarrollo Web – FrontEnd
JavaScript: Objetos

Objeto location:

Contiene información acerca de la página que se está visitando en el momento.

Propiedades:

- Href: dirección completa.
- Protocol: protocolo utilizado (http, ftp,…).
- Host: nombre del servidor y puerto utilizados.
- Hostname: nombre del servidor.

Estas dos últimas no sirven para archivos locales.

- Search: almacena los argumentos de llamada a la página, lo que acompaña a '?' (cuando existe) en la barra de direcciones.

Métodos:

- reload(): recarga la página.
- replace(url): carga la página que se indique entre los paréntesis.

  Por ejemplo: location.replace(file:///D:/ejemplos/status.html)

Desarrollo Web – FrontEnd
JavaScript: Objetos

Objeto document:

Hace referencia al contenido del documento HTML. Es el nodo raíz del DOM.

Propiedades:

- bgColor: color de fondo.
- fgColor: color del texto.
- vlink: color de los enlaces visitados.
- alink: color de los enlaces activos.

Uno de los métodos más utilizados es write().

Desarrollo Web – FrontEnd
JavaScript: Clases

- Clases Predefinidas:
  - Clase **String**: Cada vez que se asigna una cadena de caracteres a una variable, se crea un objeto de la clase String.
  - Clase **Math**: Se usa para efectuar cálculos matemáticos.
  - Clase **Date**: Para el manejo de fechas y horas.
- Clases del Browser o Navegador:
  - Tienen que ver con la navegación.
- Clases del Documento HTML:
  - Están asociadas con cualquier elemento de una página Web (link, ancla, formulario, etc.).
- Clases definidas por el usuario.

Desarrollo Web – FrontEnd
JavaScript: Modelo de Eventos

- Señal que se genera cuando el usuario interactúa con el cliente Web.
- Ejemplos:
  - Hacer ‘clic’ en un botón.
  - Presionar tecla en una caja de texto.
- Los eventos suceden a tres niveles:
  - A nivel del documento HTML.
  - A nivel de un formulario individual.
  - A nivel de un elemento de un formulario.
- El evento es gestionado por una sección de código en *JavaScript* (Gestor de Eventos).
- Declaración de Gestores de Eventos: similar a los atributos en HTML.
- Los eventos en javascript permiten al programador más flexibilidad a la hora de validar datos y restringir ‘movimientos’ de usuario.

Desarrollo Web – FrontEnd JavaScript: Gestores de Eventos (Event Handlers)

Evento	Ocurre cuando	Gestor	Etiquetas
Click	El usuario hace 'clic'	onClick	Button, document, Checkbox, Link, Radio, Reset, Submit
load	Al terminar de cargar una página	onLoad	Image, Layer, window
Focus	Al coger el foco un control	onFocus	Button, Checkbox, Password, Radio, Reset, Select
Resize	Al modificar el tamaño de la ventana	onResize	window
Blur	Un elemento pierde el foco	onBlur	Button, Checkbox, Layer, Radio, Select, Submit, Text…
Change	El valor de un campo de formulario cambia	onChange	FileUpload, Select, Text, Textarea
KeyPress	El usuario pulsa una tecla	onKeyPress	document, Image, Link, Textarea

Desarrollo Web – FrontEnd JavaScript: Gestores de Eventos (Event Handlers)

Evento	Ocurre cuando	Gestor	Etiquetas
Move	Se mueve una venta o un marco	onMove	window
Select	Se selecciona texto o área de texto	onSelect	Text, Textarea
Unload	El usuario cierra la página	onUnload	window
Reset	El usuario limpia el formulario	onReset	Form
MouseDown	El usuario pulsa el botón del ratón	onMouseDown	Button, document, Link
Mouseover	El usuario mueve el ratón sobre un link	onMouseOver	Button, document, Link
Submit	Se envía un formulario	onSubmit	Form

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

- El código manejador se encuentra en el interior del código HTML.
- Sintaxis

  <‘Etiqueta’ ‘manejador’=‘codigo a ejecutar’>

- Ejemplo

```
<input type="button" value=" Pulsar boton “
onClick="window.alert('Hola mundo!')";>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onClick

```
<html>
  <head>
    <title>Ejemplo onClick</title>
  </head>
  <body>
    <center>
      <input type="button" value=" Pulsar boton
    para saludo... " onClick="window.alert('Hola
    mundo!')";>
    </center>
   </body>
</html>
```

onLoad

```
<html>
  <head>
    <title>Ejemplo onLoad</title>
  </head>
  <body  onLoad="boton.value='hola!'">
     <center>
      <input type="button" name="boton" value=""
    onLoad = " value= 'hola mundo!' ">
    </center>
   </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onFocus

```
<html>
  <head>
     <title>Ejemplo onFocus</title>
  </head>
  <body>
    <center>
      <input type="text" value=" Al coger foco muestra mensaje... " onFocus="window.alert('Hola mundo!')";>
    </center>
   </body>
</html>
```

onResize

```
<html>
    <head>
       <title>Ejemplo onLoad</title>
    </head>
    <body onResize="alert('Hola mundo');">
    </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onBlur

```
<html>
    <head>
        <title>Ejemplo onLoad</title>
    </head>
    <body>
         <center>
                        <input type="text" size=30
    name="texto" value="Al perder foco muestra
    mensaje" onBlur = " alert('Hola mundo!');">
          </center>
    </body>
</html>
```

onChange

```
<html>
    <head>
          <title>Ejemplo onLoad</title>
    </head>
    <body>
          <center>
          <input type="text" size=30 name="texto"
    value="Al modificar texto mensaje" onChange = "
    alert('Hola mundo!');">
          </center>
    </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onKeyPress

```
<html>
    <head>
        <title>Ejemplo onLoad</title>
    </head>
    <body>
        <center>
        <input type="text" size=30
    name="texto" value="Al pulsar tecla
    mensaje" onKeyPress = " alert('Hola
    mundo!');">
        </center>
    </body>
</html>
```

onMove

```
<html>
    <head>
        <title>Ejemplo onLoad</title>
    </head>
    <body>
        <center>
            <input type="text" size=30
    name="texto" value="Al mover  muestra mensaje"
    onMove = " alert('Hola mundo!');">
        </center>
    </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onSelect

```
<html>
    <head>
         <title>Ejemplo onLoad</title>
    </head>
    <body>
         <center>
         <input type="text" size=30
    name="texto" value="Al seleccionar
    texto muestra mensaje" onSelect = "
    alert('Hola mundo!');">
         </center>
    </body>
</html>
```

onUnload

```
<html>
    <head>
          <script language="JavaScript">
         function Salida() {
            if (confirm('¿Estás seguro de
            que quieres abandonar este
            script?'))
                 return true;
            else {
                 return false;}
         }
         </script>
</head>
    <body bgcolor="white" onUnload="Salida()">
    </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript: Gestores de Eventos

onReset

```
<html>
    <head>
          <title>Ejemplo onLoad</title>
    </head>
    <body>
          <center>
          <input type="text" size=30
    name="texto" value="Al borrar muestra
    mensaje" onReset = " alert('Hola
    mundo!');">
          <input type="reset" size=30
    name="boton" value="Borrar..." onClick
    = "texto.value = '';" >
          </center>
    </body>
</html>
```

onMouseDown

```
<html>
    <head>
          <title>Ejemplo onLoad</title>
    </head>
    <body>
          <center>
          <input type="text" size=30
    name="texto" value="Pulsar boton
    raton aqui para mensaje" onMouseDown
    = " alert('Hola mundo!');">
          </center>
    </body>
</html>
```

Desarrollo Web – FrontEnd
JavaScript Asíncrono

- La asincronía es la base fundamental de Javascript, debido a que es un lenguaje de programación que solo puede ejecutar un subproceso o hilo (single thread).
- Javascript fue diseñado para ser ejecutado en navegadores, trabajar con peticiones sobre la red y procesar las interacciones de usuario, al tiempo que se mantiene una interfaz fluida. Por esta razón Javascript utiliza un modelo asíncrono y no bloqueante, con un loop de eventos implementado con un único thread para sus interfaces de entrada/salida:
    - La petición devuelve inmediatamente para evitar el bloqueo.
    - Se envía una notificación una vez que la operación se ha completado. Es entonces cuando la función es procesada, la respuesta se encola para ser ejecutada en algún momento en nuestra aplicación.

Desarrollo Web – FrontEnd
JavaScript Asíncrono

Los mecanismos que utiliza JavaScript para controlar la asincronía son:

- Callbacks.
- Promises.
- Async / Await.

Desarrollo Web – FrontEnd
JavaScript Asíncrono - Callbacks

Los callbacks son la base para que Javascript funcione de forma asíncrona.

Un callback (llamada de vuelta) es una función que recibe como argumento otra función y la ejecuta.

Ejemplo:

```
setTimeout(function(){
  console.log("Hola Mundo con retraso!");
}, 1000)
```

setTimeout es una función asíncrona que programa la ejecución de un callback una vez ha transcurrido, como mínimo, una determinada cantidad de tiempo.

Desarrollo Web – FrontEnd
JavaScript Asíncrono - Callbacks

Ejemplo:

```
<input type="text" id="a" />
<input type="text" id="b" />
<button id="operar">
    Sumar
</button>
function Sumar(a, b, callback){
    return callback(a+b);
}

document.querySelector("#operar").addEventListener('click', function(){
    var a = parseInt(document.querySelector("#a").value),
        b = parseInt(document.querySelector("#b").value);

    Sumar(a, b, function(r){
        console.log('El resultado es ' + r);
    })
})
```

El futuro digital es de todos | MinTIC

UNIVERSIDAD DEL NORTE
Vigilada Mineducación

Desarrollo Web – FrontEnd
JavaScript Asíncrono - Promises

- Una promesa (promise) es un objeto que representa el resultado de una operación asíncrona, cuenta con 3 posibles estados:
    - Pendiente.
    - Resuelta.
    - Rechazada.
- Se pueden encadenar (then), siendo el resultado de una promesa, los datos de entrada de otra posible función.
- Las promesas permiten mantener un código más legible y mantenible que las callbacks, también poseen un mecanismo para detectar errores (catch), el cual es posible usar en cualquier parte del flujo de instrucciones.

MisiónTIC2022

Desarrollo Web – FrontEnd
JavaScript Asíncrono - Promises

Ejemplo:

```
var promesa1 = function() {
          return new Promise(function(resolver, cancelar) {
               setTimeout(function() {
                  console.log("pasan 4 segundos");
                  resolver();
               }, 4000);
          });
       }
        var promesa2 = function() {
          return new Promise(function(resolver, cancelar) {
             setTimeout(function() {
                  console.log("pasan 1 segundos");
                  resolver();
              }, 1000);
           })
        }
       promesa1().then(promesa2).then(function() {
            console.log("termino");
       });
```

El futuro digital es de todos | MinTIC

UNIVERSIDAD DEL NORTE
Vigilada Mineducación

Desarrollo Web – FrontEnd
JavaScript Asíncrono-Async/Await

- Las funciones asíncronas (async / await) se crearon para simplificar el uso de las promesas.
- async declara una función como asíncrona e indica que una promesa será devuelta automáticamente.
- Se pueden declarar como async funciones con nombre o anónimas.
- La palabra await debe ser utilizada siempre dentro de una función que haya sido declarada como async, ésta esperará de forma asíncrona y no bloqueante a que una promesa se resuelva o rechace.

MisionTIC2022

Desarrollo Web – FrontEnd
JavaScript Asíncrono-Async/Await

```
function despuesde2segundos() {
  return new Promise(resolve => {
    setTimeout(() => {
      resolve('resuelto');
    }, 2000);
  });
}
async function asyncCall() {
  console.log(‘ejecutando’);
  const result = await despuésde2segundos();
  console.log(result);
  // expected output: “resuelto“
}
asyncCall();
```

El futuro digital es de todos | MinTIC

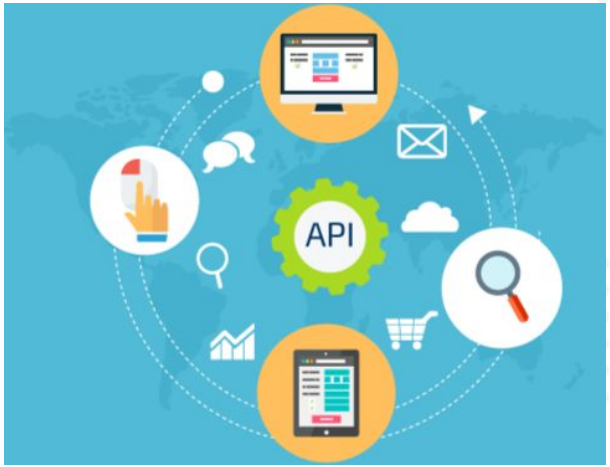
UNIVERSIDAD DEL NORTE
Vigilada Mineducación

Desarrollo Web – FrontEnd

JavaScript - API Web

Una API (Application Programming Interface) es la interfaz que permite intercambiar información entre dos componentes de software independientes. Una API hace es la intermediaria entre las funciones internas y las externas del software, creando un intercambio de información que a menudo pasa desapercibido ante el usuario final por la sencillez con la que se ejecuta.

Las API conectan diferentes partes de una plataforma de software con el objetivo de asegurar que la información llegue al lugar correcto.



MisionTIC2022

Desarrollo Web – FrontEnd
JavaScript - API Web

APIs en JavaScript del lado cliente:

JavaScript del lado cliente, tiene varias APIs a su disposición — cabe aclarar que estas no son parte del lenguaje como tal, sino que están desarrolladas sobre el núcleo de este lenguaje de programación. Generalmente, se dividen en dos categorías:

APIs de navegador: Hacen parte del navegador web y pueden desplegar datos de este. Por ejemplo, la API de Geolocalización que facilita algunos desarrollos sencillos de JavaScript para obtener datos de ubicación con los que puede trazar la ubicación del usuario en un mapa de Google.

APIs de terceros: No hacen parte por defecto del navegador, y por lo general es necesario obtener el código e información desde algún lugar de la Web. Por ejemplo, la API de Twitter permite hacer cosas como mostrar los últimos tweets en un sitio web.

El futuro digital es de todos | MinTIC

UNIVERSIDAD DEL NORTE
Vigilada Mineducación

Desarrollo Web – FrontEnd

JavaScript - API Web

APIs de navegador más usadas:

- **APIs para manipular documentos**: API DOM (Document Object Model), que permite manipular HTML y CSS.
- **APIs que obtienen datos del servidor:** Usadas para actualizar pequeñas secciones de una página web. Las APIs hacen esto posible gracias a que incluyen XMLHttpRequest y la Fetch API.
- **APIs para dibujar y manipular gráficos**: Las más populares son Canvas y WebGL.
- **APIS de audio y vídeo**: HTMLMediaElement, la Web Audio API, y WebRTC.
- **APIs de dispositivos**: APIs para manipular y recuperar información de dispositivos modernos de hardware de forma que sean útiles para aplicaciones web. Ejemplo: API de geolocalización.
- **APIS de almacenamiento en el lado del cliente:** Web Storage API.

Mision TIC 2022

Desarrollo Web – FrontEnd
JavaScript - API Web

APIs de terceros más comunes:

- **API de Twitter:** permite mostrar últimos tweets en sitio web del cliente.
- **API de Google Maps**: permite hacer todo tipo de cosas con mapas en las páginas web del cliente.
- **APIs de Facebook:** permite usar partes de las propiedades de facebook para mejorar la aplicación del cliente.
- **YouTube API:** permite integrar videos de Youtube en sitio web del cliente, buscar en Youtube, construir listas de reproducción, etc.
- **Twilio API:** suministra un framework que permite crear la funcionalidad de llamadas y videollamadas, enviar SMS o MMS y más.

El futuro digital es de todos

MinTIC

UNIVERSIDAD DEL NORTE

Vigilada Mineducación

Ejercicios de práctica

Mision TIC 2022

El futuro digital es de todos

MinTIC

UNIVERSIDAD DEL NORTE

Vigilada Mineducación

¡GRACIAS

POR SER PARTE DE ESTA EXPERIENCIA DE APRENDIZAJE!

Mision TIC 2022