UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Programador Web Avanzado

Clase N° 2: Javascript - Teoría

Profesor: Ing. Leandro Rodolfo Gil Carrano
Email: leangilutn@gmail.com

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript - Definición

Javascript es un lenguaje de programación que surgió con el objetivo inicial de programar ciertos comportamientos sobre las páginas web, respondiendo a la interacción del usuario y la realización de automatismos sencillos.

En los últimos años Javascript se está convirtiendo también en el lenguaje "integrador". **Lo encontramos en muchos ámbitos, ya no solo en Internet y la Web, también es nativo en sistemas operativos para ordenadores y dispositivos, del lado del servidor y del cliente**. Aquella visión de Javascript *"utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web"* se ha quedado muy pequeña.

Javascript - Definicion

En el contexto de un sitio web, con Javascript puedes hacer todo tipo de acciones e interacción. Antes se utilizaba para validar formularios, mostrar cajas de diálogo y poco más. Hoy es el motor de las aplicaciones más conocidas en el ámbito de Internet: Google, Facebook, Twitter, Outlook... absolutamente todas las aplicaciones que disfrutas en tu día a día en la Web tienen su núcleo realizado en toneladas de Javascript.
La Web 2.0 se basa en el uso de Javascript para implementar aplicaciones enriquecidas que son capaces de realizar todo tipo de efectos, interfaces de usuario y comunicación asíncrona con el servidor por medio de Ajax.

Javascript - Definicion

- Lenguaje del lado del cliente
  - Se ejecuta del lado del navegador
- Podemos crear efectos especiales en las páginas y definir interactividades con el usuario
- **API Javascript del HTML5:** podemos acceder a todo tipo de recursos adicionales, como la cámara, espacio para almacenamiento de datos, creación de gráficos basados en vectores y mapas de bits
- **Javascript != Java**

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Tipos de ejecucion

- El código javascript en muchos casos se embebe dentro del código HTML
- El código Javascript se coloca dentro de los tags **<script></script>**
- Hay dos formas de ejecutar código javascript:
  - Ejecución directa
  - Ejecución por eventos

Javascript – Ejecución directa

- Es el método de ejecutar scripts más básico. En este caso se incluyen las instrucciones dentro de la etiqueta <SCRIPT>, cuando el navegador lee la página y encuentra un script va interpretando las líneas de código y las va ejecutando una después de otra.

Javascript – Respuesta a un evento

- Los eventos son acciones que realiza el usuario. Los programas como Javascript están preparados para atrapar determinadas acciones realizadas, en este caso sobre la página, y realizar acciones como respuesta.
Por ejemplo la pulsación de un botón, el movimiento del ratón o la selección de texto de la página.

Javascript – NOSCRIPT

- En caso de no tener javascript activado en nuestro navegador, podemos realizar ciertas acciones al detectar esta condición. Por ejemplo indicar que estamos utilizando una versión del navegador sin dicho soporte.

```
<noscript>
Este navegador no comprende los scripts que se están ejecutando, debes actualizar
<br><br>
</noscript>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Incluir archivos externos

- Desde nuestro HTML podemos incluir codigo javascript externo.

Esto lo podemos hacer con el tag **<script>**, veamos el siguiente ejemplo:

```
<script src="/carrito_compra/assets/bootstrap/js/bootstrap.min.js"></script>
```

Sintaxis

Comentarios

```
<script>
//Este es un comentario de una línea
/*Este comentario se puede extender
por varias líneas.
Las que quieras*/
</script>
```

En caso de utilizar archivos JS minificados, se deben utilizar comentarios extendidos y **NO** de una línea.

Mayúsculas y Minúsculas

Javascript detecta las diferencias entre mayúsculas y minúsculas.

Por regla general, los nombres de las cosas en Javascript se escriben siempre en minúsculas, salvo que se utilice un nombre con más de una palabra, pues en ese caso se escribirán con mayúsculas las iniciales de las palabras siguientes a la primera.

Separación de instrucciones

Javascript tiene dos maneras de separar instrucciones:

- La primera es a través del carácter punto y coma (;)

```
llamadoFuncion();
```

- La segunda es a través de un salto de línea.

```
llamadoFuncion()
```

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Variables

Variables

Una variable es un espacio en memoria donde se almacena un dato, un espacio donde podemos guardar cualquier tipo de información que necesitemos para realizar las acciones de nuestros programas.

Los nombres de las variables han de construirse con caracteres alfanuméricos y el carácter subrayado (_).
Los nombres tienen que comenzar por un carácter alfabético o el subrayado. **No podemos utilizar caracteres raros como el signo +, un espacio.**

Las variables no pueden utilizar nombres reservados, por ejemplo if, for, while, etc

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Declaración de Variables

Declarar variables consiste en definir y de paso informar al sistema de que vas a utilizar una variable.

Javascript cuenta con la palabra "**var**" que utilizaremos cuando queramos declarar una o varias variables. Como es lógico, se utiliza esa palabra para definir la variable antes de utilizarla. Ejemplo:

```
var operando1;
var operando2;

var operando1 = 23;
var operando2 = 33;

var operando1,operando2;
```

Variables ejemplo

```
<script language="javascript">

    //creo una variable
    var x;
    //x actualmente no tiene ningún valor
    x = 25;
    //ahora x tiene el valor numérico 25

    //creo una segunda variable
    var y = 230;
    //he creado una variable y asignado un valor en un solo paso!!

    //las variables guardan datos con los que puedo realizar operaciones
    var suma;
    suma = x + y;
    //he creado la variable suma y he asignado la suma de x e y
    alert(suma);
    //he mostrado el valor de la variable suma

</script>
```

Variables ejemplo

```
<script language="javascript">

    //creo una variable
    var x;
    //x actualmente no tiene ningún valor
    x = 25;
    //ahora x tiene el valor numérico 25

    //creo una segunda variable
    var y = 230;
    //he creado una variable y asignado un valor en un solo paso!!

    //las variables guardan datos con los que puedo realizar operaciones
    var suma;
    suma = x + y;
    //he creado la variable suma y he asignado la suma de x e y
    alert(suma);
    //he mostrado el valor de la variable suma

</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Ámbito de las variables

Se le llama ámbito de las variables al lugar donde estas están disponibles. Por lo general, cuando declaramos una variable hacemos que esté disponible en el lugar donde se ha declarado.

Variables globales: son las que están declaradas en el ámbito más amplio posible.

```
<script>
var variableGlobal
</script>
```

Variables locales: sólo podremos acceder a ellas dentro del lugar donde se ha declarado

```
<script>
function miFuncion (){
    var variableLocal
}
</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Variables – Tipos de datos

Las variables no están forzadas a guardar un tipo de datos en concreto y por lo tanto no especificamos ningún tipo de datos para una variable cuando la estamos declarando. Podemos introducir cualquier información en una variable de cualquier tipo, incluso podemos ir cambiando el contenido de una variable de un tipo a otro sin ningún problema. Vamos a ver esto con un ejemplo.

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Operadores

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Operadores de cadenas

Las cadenas de caracteres, o variables de texto, también tienen sus propios operadores para realizar acciones típicas sobre cadenas. Aunque javascript sólo tiene un operador para cadenas se pueden realizar otras acciones con una serie de funciones predefinidas en el lenguaje que veremos más adelante.

Para concatenar 2 cadenas de texto debemos usar el operador **“+”**

```
<script>
cadenaConcatenada = cadena1 + cadena2 /
</script>
```

Operadores lógicos

Estos operadores sirven para realizar operaciones lógicas, que son aquellas que dan como resultado un verdadero o un falso, y se utilizan para tomar decisiones en nuestros scripts.

```
<script>

prueba && prueba2 //Operador and

prueba || prueba2 //Operador or

</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Typeof

Para comprobar el tipo de un dato se puede utilizar otro operador que está disponible a partir de javascript 1.1, el **operador typeof**, que devuelve una cadena de texto que describe el tipo del operador que estamos comprobando.

```
<script>

document.write("<br>El tipo de boleano es: " + typeof boleano)

</script>
El tipo de boleano es: boolean
```

Estructuras

If

```
<script>

if (expresión) {
    //acciones a realizar en caso positivo
    //...
} else {
    //acciones a realizar en caso negativo
    //...
}

</script>
```

If

```
<script>

//Operador ternario
Variable = (condición) ? valor1 : valor2

</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

For

```
<script>
for (inicialización; condición; actualización) {
    //sentencias a ejecutar en cada iteración
}
</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

While

```
<script>
while (condición){
    //sentencias a ejecutar
}
</script>
```

Funciones

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Funciones

Primero se escribe la palabra **function**, reservada para este uso. Seguidamente se escribe el nombre de la función, que como los nombres de variables puede tener números, letras y algún carácter adicional como en guión bajo.

```
<script>
function nombrefuncion (){
    instrucciones de la función
    ...
}
</script>
```

Funciones – parámetros

Al hilo del uso de parámetros en nuestros programas Javascript, tenemos que saber que los parámetros de las funciones se pasan por valor. Esto quiere decir que estamos pasando valores y no variables. En la práctica, aunque modifiquemos un parámetro en una función, la variable original que habíamos pasado no cambiará su valor.

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Arrays

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Creación

El primer paso para utilizar un array es crearlo. Para ello utilizamos un objeto Javascript ya implementado en el navegador.

```
<script>
//Array vacio
var miArray = new Array()
//Creacion de array con numero de compartimentos
var miArray = new Array(10)
//Setear datos
miArray[0] = 290
miArray[1] = 97
miArray[2] = 127
//Declaracion e inicializacion
var arrayRapido = [12,45,"array inicializado en su declaración"]
</script>
```

Longitud

Para obtener la longitud de un array utilizaremos el método **length**

```
<script>
document.write("Longitud del array: " + miArray.length)
</script>
```

Longitud

Para obtener la longitud de un array utilizaremos el método **length**

```
<script>
document.write("Longitud del array: " + miArray.length)
</script>
```

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Clases y Objetos

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Funciones nativas

eval(string)
Esta función recibe una cadena de caracteres y la ejecuta como si fuera una sentencia de Javascript.

parseInt(cadena,base)
Recibe una cadena y una base. Devuelve un valor numérico resultante de convertir la cadena en un número en la base indicada.

parseFloat(cadena)
Convierte la cadena en un número y lo devuelve.

escape(carácter)
Devuelve un el carácter que recibe por parámetro en una codificación ISO Latin 1.

unescape(carácter)
Hace exatamente lo opuesto a la función escape.

isNaN(número)
Devuelve un boleano dependiendo de lo que recibe por parámetro. Si no es un número devuelve un true, si es un numero devuelve false.

Objetos

Para instanciar un objeto lo hacemos con el método **new**.
Para acceder a un método de un objeto lo hacemos con **“.”**

```
<script>
//Instancia un objeto
var miObjeto = new miClase()
//Acceder a un metodo del objeto
miObjeto.miMetodo(parametro1,parametro2)
</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Clase string

Propiedades

Length

La clase String sólo tiene una propiedad: length, que guarda el número de caracteres del String.

Métodos

charAt(indice)

Devuelve el carácter que hay en la posición indicada como índice. Las posiciones de un string empiezan en 0.

indexOf(carácter,desde)

Devuelve la posición de la primera vez que aparece el carácter indicado por parámetro en un string. Si no encuentra el carácter en el string devuelve -1. El segundo parámetro es opcional y sirve para indicar a partir de que posición se desea que empiece la búsqueda.

lastIndexOf(carácter,desde)

Busca la posición de un carácter exáctamente igual a como lo hace la función indexOf pero desde el final en lugar del principio. El segundo parámetro indica el número de caracteres desde donde se busca, igual que en indexOf.

http://www.desarrolloweb.com/articulos/726.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Clase date

```
<script>
//Instanciar el objeto
miFecha = new Date(año,mes,dia,hora,minutos,segundos)
</script>
```

Métodos

getDate()
Devuelve el día del mes.

getDay()
Devuelve el día de la semana.

getHours()
Retorna la hora.

getMinutes()
Devuelve los minutos.

getMonth()
Devuelve el mes (atención al mes que empieza por 0).

getSeconds()
Devuelve los segundos.

http://www.desarrolloweb.com/articulos/744.php

Creación de clases

Para crear nuestros propios objetos debemos crear una clase, que recordamos que es algo así como la definición de un objeto con sus propiedades y métodos. Para crear la clase en Javascript debemos escribir una función especial, que se encargará de construir el objeto en memoria e inicializarlo. Esta función se le llama **constructor.**

```
<script>
function MiClase (valor_inicializacion){
    //Inicializo las propiedades y métodos
    //this hace referencia a su propia clase
       this.miPropiedad = valor_inicializacion
    this.miMetodo = nombre_de_una_funcion_definida
}
</script>
```

El nombre del constructor comienza con mayúscula

Creación de clases

Para colocar un método en una clase debemos asignar la función que queremos que sea el método al objeto que se está creando.

```
<script>
function AlumnoUniversitario(nombre, edad){
    this.nombre = nombre
    this.edad = edad
    this.numMatricula = null
    //matriculate es una funciona declarada a parte
    this.matriculate = matriculate
    this.imprimete = imprimete
}
function matriculate(){
    //Declaracion del metodo
}
</script>
```

UTN.BA

UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM

DOM (Document Object Model o modelo de objetos del navegador) que nos sirve para acceder a cualquiera de los componentes que hay dentro de una página. Por medio del DOM podremos controlar al navegador en general y a los distintos elementos que se encuentran en la página.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Jerarquía

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Window

Todos los objetos comienzan en un objeto que se llama window. Este objeto ofrece una serie de métodos y propiedades para controlar la ventana del navegador. Con ellos podemos controlar el aspecto de la ventana, la barra de estado, abrir ventanas secundarias y otras cosas que veremos más adelante cuando expliquemos con detalle el objeto.

Además de ofrecer control, el objeto window da acceso a otros objetos como el documento (La página web que se está visualizando), el historial de páginas visitadas o los distintos frames de la ventana. De modo que para acceder a cualquier otro objeto de la jerarquía deberíamos empezar por el objeto window. Tanto es así que javascript entiende perfectamente que la jerarquía empieza en window aunque no lo señalemos.

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Window

```
<script>
window.document.bgColor = "red"
window.document.write("El texto a escribir")
</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Se trata de las propiedades que son arrays, por ejemplo la propiedad images es un array con todas las imágenes de la página web. También encontramos arrays para guardar los enlaces de la página, los applets, los formularios y las anclas.

Cuando una página se carga, el navegador construye en memoria la jerarquía de objetos. De manera adicional, construye también estos arrays de objetos.

```
<script>
for (i=0;i<document.images.length;i++){
    document.write(document.images[i].src)
    document.write("<br>")
}
</script>
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Para ver en detalle el objeto window:
Propiedades
http://www.desarrolloweb.com/articulos/826.php
Métodos:
http://www.desarrolloweb.com/articulos/827.php

Para ver en detalle el objeto document:
Propiedades
http://www.desarrolloweb.com/articulos/846.php
Métodos
http://www.desarrolloweb.com/articulos/861.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Objeto window:

Propiedades:

- **document:** Objeto que contiene el la página web que se está mostrando.
- **Frame:** Un objeto frame de una página web. Se accede por su nombre.
- **history:** Objeto historial de páginas visitadas.
- **location:** La URL del documento que se está visualizando. Podemos cambiar el valor de esta propiedad para movernos a otra página.
- **name:** Nombre de la ventana. Lo asignamos cuando abrimos una nueva ventana.

Para ver mas:

http://www.desarrolloweb.com/articulos/826.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Objeto window:

Métodos:

- **alert(texto):** Presenta una ventana de alerta donde se puede leer el texto que recibe por parámetro
- **blur():** Quitar el foco de la ventana actual
- **close():** Cierra la ventana.
- **forward():** Ir una página adelante en el historial de páginas visitadas.
- **open():** Abre una ventana secundaria del navegador.

Para ver mas:
http://www.desarrolloweb.com/articulos/827.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Objeto document:

Propiedades:

- **bgColor:** El color de fondo del documento.
- **cookie:** Accede a una cookie del navegador
- **domain:** Nombre del dominio del servidor de la página.
- **fgColor:** El color del texto. Para ver los cambios hay que recargar la página.
- **ids:** Para acceder a estilos CSS.
- **title:** El titulo de la página.

Para ver mas:

http://www.desarrolloweb.com/articulos/846.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

DOM - Document

Objeto document:

Métodos:

- **close():** Cierra el flujo del documento.
- **open():** Abre el flujo del documento.
- **write():** Escribe dentro de la página web. Podemos escribir etiquetas HTML y texto normal.
- **writeln():** Escribe igual que el método write(), aunque coloca un salto de línea al final.

Para ver mas:
http://www.desarrolloweb.com/articulos/861.php

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

ES 6

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Definir una constante

En javascript ES 6 podemos definir una constante con la palabra reservada const

```
const PI = 3.141593;
alert(PI > 3.0);
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Let

Podemos utilizar let en lugar de var a la hora de declarar una variable, cuando queremos que esta solo sea accedida de manera local en determinado ámbito. Por ejemplo:

```
//ES5
(function() {
  console.log(x); // x no está definida aún.
    if(true) {
      var x = "hola mundo";
    }
    console.log(x);
    // Imprime "hola mundo", porque "var" hace que sea global
    // a la función;
})();

//ES6
(function() {
  if(true) {
      let x = "hola mundo";
    }
    console.log(x);
    //Da error, porque "x" ha sido definida dentro del "if"
})();
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Función Arrow

```
var miFuncion = function(num) {
    return num + num;
}
//ES6
var miFuncion = (num) => num + num;
```

```
var data = [{...}, {...}, {...}, ...];

data.forEach(elem => {

  console.log(elem);

});
```

En ambos ejemplos se puede que se sustituye el uso de la palabra reservada function dando simplicidad en el código.

(parametro1,parametro2,…,parámetro n) =>{Definición de la función}

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Clases

Ahora JavaScript tendrá clases, muy parecidas las funciones constructoras de objetos que realizábamos en el estándar anterior, pero ahora bajo el paradigma de clases, con todo lo que eso conlleva, como por ejemplo, herencia.

```
class LibroTecnico extends Libro {
  constructor(tematica, paginas) {
      super(tematica, paginas);
        this.capitulos = [];
        this.precio = "";
        // ...
    }
    metodo() {
      // ...
    }
}
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Clases (this)

Antiguamente teníamos que cachearlo en otra variable ya que solo hace referencia al contexto en el que nos encontremos. Por ejemplo, en el siguiente código si no hacemos var that = this dentro de la función document.addEventListener, this haría referencia a la función que pasamos por Callback y no podríamos llamar a foo()

```
//ES3
var obj = {
  foo : function() {...},
    bar : function() {
      var that = this;
        document.addEventListener("click", function(e) {
          that.foo();
        });
    }
}
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Clases (this)

Con ECMAScript5 la cosa cambió un poco, y gracias al método bind podíamos indicarle que this hace referencia a un contexto y no a otro.

```
//ES5
var obj = {
  foo : function() {...},
    bar : function() {
        document.addEventListener("click", function(e) {
          this.foo();
        }.bind(this));
    }
}
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Clases (this)

Ahora con ES6 y la función Arrow => la cosa es todavía más visual y sencilla.

```
//ES6
var obj = {
  foo : function() {...},
    bar : function() {
      document.addEventListener("click", (e) => this.foo());
    }
}
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Template Strings

Con ES6 podemos interpolar Strings de una forma más sencilla que como estábamos haciendo hasta ahora. Fíjate en este ejemplo:

```
//ES6
let nombre1 = "JavaScript";
let nombre2 = "awesome";
console.log(`Sólo quiero decir que ${nombre1} is ${nombre2}`);
// Solo quiero decir que JavaScript is awesome
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Valores por defecto

Otra novedad es asignar valores por defecto a las variables que se pasan por parámetros en las funciones. Antes teníamos que comprobar si la variable ya tenía un valor. Ahora con ES6 se la podemos asignar según creemos la función.

```
//ES5
function(valor) {
  valor = valor || "foo";
}

//ES6
function(valor = "foo") {...};
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Javascript – Módulos

Ahora JavaScript se empieza a parecer a lenguajes como Python o Ruby. Llamamos a las funciones desde los propios Scripts, sin tener que importarlos en el HTML, si usamos JavaScript en el navegador.

```
//File: lib/person.js
module "person" {
    export function hello(nombre) {
        return nombre;
    }
}
```

Javascript – Módulos

Y para importar en otro fichero:

```
//File: app.js
import { hello } from "person";
var app = {
   foo: function() {
        hello("Carlos");
    }
}
export app;
```

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Sitios de referencia

Podres consultar mas información en:

- http://www.desarrolloweb.com/manuales/20/