# JavaScript: Objetos predefinidos

Francisco J. Martín Mateos
Carmen Graciani Diaz

Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

# Objetos en JavaScript

- JavaScript es un lenguaje orientado a objetos
- Un objeto es un tipo estructurado de dato que contiene propiedades y métodos
  - Las propiedades son valores asociados con el objeto
  - Los métodos son acciones que pueden ser evaluadas sobre los objetos

## Ejemplo de propiedades y métodos de un objeto

```
<HR>
<SCRIPT>
var txt = "Hola hola";
document.write(txt.length);
document.write(txt.toUpperCase());
</SCRIPT>
<HR>
```

# Objetos en JavaScript

- Objetos predefinidos en JavaScript
  - Cadenas de texto: `String`
  - Fechas: `Date`
  - Arreglos: `Array`
  - Lógicos: `Boolean`
  - Matemático: `Math`
  - Expresiones regulares: `RegExp`
- Modelo de objetos del documento: DOM

# Objeto String

- El objeto `String` se utiliza para manipular cadenas de texto
- Cualquier cadena de texto es un objeto `String` y no es necesario un proceso específico para crear un objeto `String` nuevo
- Propiedades
  - `length`: Longitud del objeto `String`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!";
document.write(txt.length);
</SCRIPT>
```

# Objeto String

- Método `toLowerCase()`
  - Sintaxis: `objetoString.toLowerCase()`
  - Resultado: Devuelve la cadena obtenida poniendo en minúscula el `objetoString`
- Método `toUpperCase()`
  - Sintaxis: `objetoString.toUpperCase()`
  - Resultado: Devuelve la cadena obtenida poniendo en mayúscula el `objetoString`
- Nota: No modifican el `objetoString` original

**Ejemplo**

```
<SCRIPT>
var txt = "Hola Hola"
document.write(txt.toLowerCase() + "<BR>");
document.write(txt.toUpperCase() + "<BR>");
</SCRIPT>
```

# Objeto String

- Método `concat()`
  - Sintaxis: `objetoString.concat(texto1,...,textoN)`
  - Argumentos: Varios cadenas `texto1`, ..., `textoN`
  - Resultado: Concatena `objetoString` con las cadenas de texto que se pasan como argumento, en el orden en que se proporcionan

## Ejemplo

```
<SCRIPT>
var txt1 = "hola "
var txt2 = "hola!";
document.write(txt1.concat(txt2));
</SCRIPT>
```

# Objeto String

- Método `charAt()`
  - Sintaxis: `objetoString.charAt(indice)`
  - Argumento: Un número natural `indice`
  - Resultado: Devuelve el carácter del `objetoString` que ocupa la posición dada por el `indice`
  - Nota: La indexación comienza por el `0`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.charAt(0));
document.write(txt.charAt(9));
</SCRIPT>
```

# Objeto String

- Método `indexOf()`
  - Sintaxis: `objetoString.indexOf(texto,indice)`
  - Argumentos: Una cadena `texto` y un número natural `indice`
  - Resultado: Devuelve la posición de la primera ocurrencia de la cadena `texto` en el `objetoString` a partir de la posición dada por el `indice`
  - Nota: El argumento `indice` es opcional, por defecto vale `0`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.indexOf("hola") + "<BR>");
document.write(txt.indexOf("hola",3) + "<BR>");
document.write(txt.indexOf("adios"));
</SCRIPT>
```

# Objeto String

- Método `lastIndexOf()`
  - Sintaxis: `objetoString.lastIndexOf(texto,indice)`
  - Argumentos: Una cadena `texto` y un número natural `indice`
  - Resultado: Devuelve la posición de la última ocurrencia de la cadena `texto` en el `objetoString`, hacia atrás desde la posición dada por el `indice`
  - Nota: El argumento `indice` es opcional, por defecto la búsqueda comienza en la última posición

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.lastIndexOf("hola") + "<BR>");
document.write(txt.lastIndexOf("hola",3) + "<BR>");
document.write(txt.lastIndexOf("adios"));
</SCRIPT>
```

# Objeto String

- Método `substring()`
  - Sintaxis: `objetoString.substring(inicio,final)`
  - Argumentos: Dos números naturales `inicio` y `final`
  - Resultado: Devuelve la subcadena del `objetoString` desde la posición dada por `inicio` hasta la posición dada por `final`
  - Notas:
    - El argumento `final` es opcional, su valor por defecto es el de la última posición en `objetoString`
    - La posición `inicio` puede ser mayor que la posición `final`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.substring(3) + "<BR>");
document.write(txt.substring(2,8) + "<BR>");
document.write(txt.substring(9,1));
</SCRIPT>
```

# Objeto String

- Método `substr()`
  - Sintaxis: `objetoString.substr(inicio,longitud)`
  - Argumentos: Dos numeros naturales `inicio` y `longitud`
  - Resultado: Devuelve la subcadena del `objetoString` desde la posición dada por `inicio` y con la `longitud` dada
  - Nota: El argumento `longitud` es opcional, su valor por defecto es lo que queda hasta llegar al final del `objetoString`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.substr(3) + "<BR>");
document.write(txt.substr(2,4));
</SCRIPT>
```

# Objeto String

- Método `search()`
  - Sintaxis: `objetoString.search(texto)`
  - Argumentos: Una cadena `texto`
  - Resultado: Devuelve la posición de la primera ocurrencia de la cadena `texto` en el `objetoString`
  - Nota: Es equivalente a `objetoString.indexOf(texto,0)`

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.search("hola") + "<BR>");
document.write(txt.search("adios"));
</SCRIPT>
```

# Objeto String

- Método `replace()`
  - Sintaxis: `objetoString.replace(texto1,texto2)`
  - Argumentos: Dos cadenas `texto1` y `texto2`
  - Resultado: Devuelve la cadena obtenida reemplazando la primera ocurrencia de `texto1` en el `objetoString` por la cadena `texto2`
  - Nota: Los argumentos `texto1` y `texto2` pueden tener distinta longitud

## Ejemplo

```
<SCRIPT>
var txt = "hola hola!"
document.write(txt.replace("hola","adios"));
</SCRIPT>
```

# Objeto String

- Método `split()`
  - Sintaxis: `objetoString.split(texto,total)`
  - Argumentos: Una cadena `texto` y un número natural `total`
  - Resultado: Devuelve un arreglo de cadenas obtenido rompiendo el `objetoString` en las ocurrencias de la subcadena `string` hasta obtener el `total` indicado de trozos
  - Nota: El argumento `total` es opcional, si no se indica se obtendrán todos los trozos posibles

## Ejemplo

```
<SCRIPT>
var txt = "Este es un ejemplo"
document.write(txt.split(" ",2) + "<BR>");
document.write(txt.split("",10) + "<BR>");
document.write(txt.split(" "));
</SCRIPT>
```

# Objeto Date

- El objeto `Date` se utiliza para trabajar con fechas
- Construimos un objeto `Date` nuevo de la siguiente forma: `new Date()`
  - Utilizado de esta forma, el objeto creado toma como valor la fecha actual: año, mes, día, hora, minuto y segundo
- También se puede construir un objeto `Date` indicando como argumento una cadena de texto con la información sobre la fecha

## Ejemplo

```
<SCRIPT>
fecha = new Date();
fecha = new Date("25 Nov 1971");
fecha = new Date("Nov 25 1971");
fecha = new Date("25 Nov 1971 00:30");
fecha = new Date("25 Nov 1971 00:30:24");
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getFullYear()`
  - Sintaxis: `objetoDate.getFullYear()`
  - Resultado: Devuelve el año completo del `objetoDate`
- Método `setFullYear()`
  - Sintaxis: `objetoDate.setFullYear(año,mes,dia)`
  - Argumentos: Un número `año` de cuatro cifras, un número `mes` del 0 al 11 y un número `dia` del 1 al 31
  - Resultado: Cambia el año, el mes y el día del mes del `objetoDate` y ajusta la fecha
  - Notas: Los argumentos `mes` y `dia` son opcionales

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getFullYear() + "<BR>");
fecha.setFullYear(1971,10,25);
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getMonth()`
  - Sintaxis: `objetoDate.getMonth()`
  - Resultado: Devuelve el mes (0-11) del `objetoDate`
- Método `setMonth()`
  - Sintaxis: `objetoDate.setMonth(mes,dia)`
  - Argumentos: Un número `mes` del 0 al 11 y un número `dia` del 1 al 31
  - Resultado: Cambia el mes y el dia del mes del `objetoDate` y ajusta la fecha
  - Notas: El argumento `dia` es opcional

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getMonth() + "<BR>");
fecha.setMonth(10);
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getDate()`
  - Sintaxis: `objetoDate.getDate()`
  - Resultado: Devuelve el día del mes (1-31) del `objetoDate`
- Método `setDate()`
  - Sintaxis: `objetoDate.setDate(dia)`
  - Argumento: Un número `dia` del 1 al 31
  - Resultado: Cambia el día del mes del `objetoDate` y ajusta la fecha

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getDate() + "<BR>");
fecha.setDate(24);
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getDay()`
  - Sintaxis: `objetoDate.getDay()`
  - Resultado: Devuelve el día de la semana (0-6) del `objetoDate`
  - Nota: La semana comienza en Domingo

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getDay() + "<BR>");
</SCRIPT>
```

# Objeto Date

- Método `getHours()`
  - Sintaxis: `objetoDate.getHours()`
  - Resultado: Devuelve la hora (0-23) del `objetoDate`
- Método `setHours()`
  - Sintaxis: `objetoDate.setHours(hora,minuto,segundo)`
  - Argumentos: Un número `hora` del 0 al 23, un número `minuto` del 0 al 59 y un número `segundo` del 0 al 59
  - Resultado: Cambia la hora, el minuto y el segundo del `objetoDate` y ajusta la fecha
  - Nota: Los argumentos `minuto` y `segundo` son opcionales

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getHours() + "<BR>");
fecha.setHours(22,20,10);
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getMinutes()`
  - Sintaxis: `objetoDate.getMinutes()`
  - Resultado: Devuelve el minuto (0-59) del `objetoDate`
- Método `setMinutes()`
  - Sintaxis: `objetoDate.setMinutes(minuto,segundo)`
  - Argumentos: Un número `minuto` del 0 al 59 y un número `segundo` del 0 al 59
  - Resultado: Cambia el minuto y el segundo del `objetoDate` y ajusta la fecha
  - Nota: El argumento `segundo` es opcional

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getMinutes() + "<BR>");
fecha.setMinutes(20,10);
document.write(fecha);
</SCRIPT>
```

# Objeto Date

- Método `getSeconds()`
  - Sintaxis: `objetoDate.getSeconds()`
  - Resultado: Devuelve el segundo (0-59) del `objetoDate`
- Método `setSeconds()`
  - Sintaxis: `objetoDate.setSeconds(segundo)`
  - Argumento: Un número `segundo` del 0 al 59
  - Resultado: Cambia el segundo del `objetoDate` y ajusta la fecha

## Ejemplo

```
<SCRIPT>
var fecha = new Date();
document.write(fecha.getSeconds() + "<BR>");
fecha.setSeconds(10);
document.write(fecha);
</SCRIPT>
```

# Objeto Array

- El objeto `Array` se utiliza para almacenar un conjunto de valores en una misma variable (arreglos)
- Construcción de objetos `Array`
  - Sin indicar el tamaño: `new Array()`
  - Indicando el tamaño: `new Array(tamaño)`
  - Indicando los valores: `new Array(val1,...,valN)`

## Ejemplo

```
<SCRIPT>
var coches = new Array();
var coches = new Array(2);
Array0 = "Ford";
Array1 = "Seat";
var coches = new Array("Ford","Seat");
document.write(coches);
</SCRIPT>
```

# Objeto Array

- Propiedades
  - `length`: Número de elementos del array
  - Nota: Sirve para conocer el tamaño de un array y para modificarlo

## Ejemplo

```
<SCRIPT>
var coches = new Array("Ford","Seat");
document.write(coches + "<BR>");
document.write(coches.length + "<BR>");
coches.length = 3;
document.write(coches + "<BR>");
coches.length = 1;
document.write(coches);
</SCRIPT>
```

# Objeto Array

- Método `concat()`
  - Sintaxis: `objetoArray.concat(array1,...,arrayN)`
  - Argumento: Varios arrays `array1`, ..., `arrayN`
  - Resultado: Concatena `objetoArray` con todos los arrays que se pasan como argumento, en el orden en que se proporcionan

## Ejemplo

```
<SCRIPT>
var coches1 = new Array("Ford","Seat");
var coches2 = new Array("Fiat","BMW");
document.write(coches1.concat(coches2));
</SCRIPT>
```

# Objeto Array

- Método `pop()`
  - Sintaxis: `objetoArray.pop()`
  - Resultado: Elimina y devuelve el último elemento del `objetoArray`
- Método `push()`
  - Sintaxis: `objetoArray.push(elt1,...,eltN)`
  - Argumento: Varios elementos `elt1`, ..., `eltN`
  - Resultado: Añade al final del `objetoArray` los elementos que se pasan como argumento, en el orden en que se proporcionan

## Ejemplo

```
<SCRIPT>
var coches = new Array("Ford","Seat","Fiat");
document.write(coches.pop() + "<BR>");
coches.push("BMW","Toyota");
document.write(coches);
</SCRIPT>
```

# Objeto Array

- Método `shift()`
  - Sintaxis: `objetoArray.shift()`
  - Resultado: Elimina y devuelve el primer elemento del `objetoArray`
- Método `unshift()`
  - Sintaxis: `objetoArray.unshift(elt1,...,eltN)`
  - Argumento: Varios elementos `elt1`, ..., `eltN`
  - Resultado: Añade al principio del `objetoArray` los elementos que se pasan como argumento, en el orden en que se proporcionan

## Ejemplo

```
<SCRIPT>
var coches = new Array("Ford","Seat","Fiat");
document.write(coches.shift() + "<BR>");
coches.unshift("BMW","Toyota");
document.write(coches);
</SCRIPT>
```

# Objeto Array

- Método `reverse()`
  - Sintaxis: `objetoArray.reverse()`
  - Resultado: Invierte el orden de los elementos en el `objetoArray`
- Nota: Modifica el array original

## Ejemplo

```
<SCRIPT>
var coches = new Array("Seat","Toyota","Ford","BMW");
coches.reverse();
document.write(coches + "<BR>");
</SCRIPT>
```

# Objeto Array

- Método `sort()`
  - Sintaxis: `objetoArray.sort()`
  - Resultado: Ordena alfabéticamente los elementos en el `objetoArray`
- Nota: Modifica el array original

## Ejemplo

```
<SCRIPT>
var coches = new Array("Seat","Toyota","Ford","BMW");
coches.sort();
document.write(coches);
</SCRIPT>
```

# Objeto Array

- Método `join()`
  - Sintaxis: `objetoArray.join(separador)`
  - Argumento: Una cadena `separador`
  - Resultado: Devuelve una cadena de texto con todos los elementos del `objetoArray`, en el orden en que están, insertando el `separador` entre ellos
  - Nota: El argumento `separador` es opcional, su valor por defecto es ","

## Ejemplo

```
<SCRIPT>
var coches = new Array("Seat","Toyota","Ford","BMW");
document.write(coches.join(";"));
</SCRIPT>
```

# Objeto Array

- Método `slice()`
  - Sintaxis: `objetoArray.slice(inicio,final)`
  - Argumento: Dos números naturales `inicio` y `final`
  - Resultado: Devuelve los elementos del `objetoArray` desde la posición dada por `inicio` hasta la posición anterior a la dada por `final`
  - Nota: El argumento `final` es opcional, su valor por defecto es la longitud del `objetoArray`

## Ejemplo

```
<SCRIPT>
var coches = new Array("Seat","Toyota","Ford","BMW");
document.write(coches.slice(2) + "<BR>");
document.write(coches.slice(1,4));
</SCRIPT>
```

# Objeto Math

- El objeto `Math` se utiliza para realizar operaciones matemáticas
- Propiedades
  - `Math.PI`: El número $\Pi$
  - `Math.E`: El número $\epsilon$
  - `Math.SQRT2`: La raíz cuadrada de 2
  - `Math.SQRT1_2`: La raíz cuadrada de 1/2
  - `Math.LN2`: El logaritmo de 2
  - `Math.LN10`: El logaritmo de 10

# Objeto Math

- Método `floor()`
  - Sintaxis: `Math.floor(x)`
  - Argumento: Un número `x`
  - Resultado: Devuelve el entero inmediatamente anterior a `x`
- Método `ceil()`
  - Sintaxis: `Math.ceil(x)`
  - Argumento: Un número `x`
  - Resultado: Devuelve el entero inmediatamente posterior a `x`
- Método `round()`
  - Sintaxis: `Math.round(x)`
  - Argumento: Un número `x`
  - Resultado: Devuelve el entero más cercano a `x`

# Objeto Math

- Método `abs()`
  - Sintaxis: `Math.abs(x)`
  - Argumento: Un número `x`
  - Resultado: Devuelve el valor absoluto de `x`
- Método `max()`
  - Sintaxis: `Math.max(x,y)`
  - Argumento: Dos números `x` e `y`
  - Resultado: Devuelve el máximo entre `x` e `y`
- Método `min()`
  - Sintaxis: `Math.min(x,y)`
  - Argumento: Dos números `x` e `y`
  - Resultado: Devuelve el minimo entre `x` e `y`

# Objeto Math

- Método `pow()`
  - Sintaxis: `Math.pow(x,y)`
  - Argumento: Dos números `x` e `y`
  - Resultado: Devuelve el valor $x^y$
- Método `sqrt()`
  - Sintaxis: `Math.sqrt(x)`
  - Argumento: Un número `x`
  - Resultado: Devuelve la raíz cuadrada de `x`
- Método `random()`
  - Sintaxis: `Math.random()`
  - Resultado: Devuelve un número pseudoaleatorio entre 0 y 1