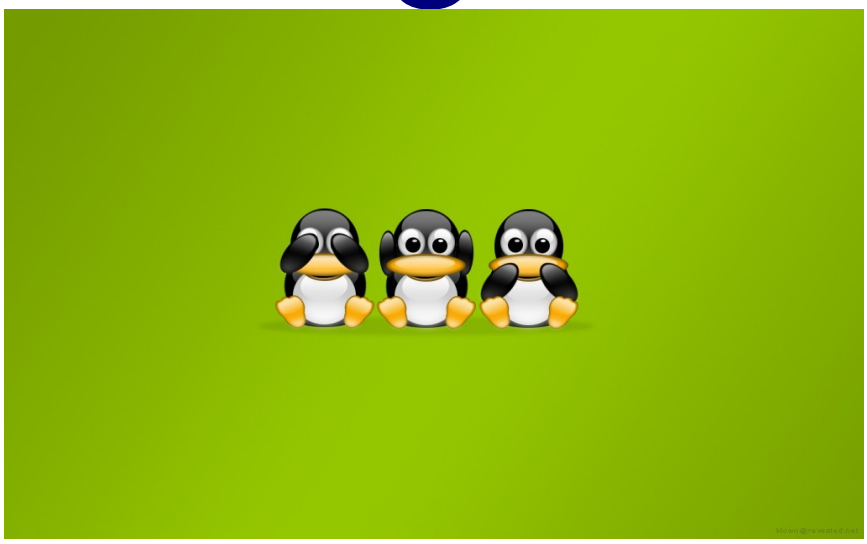
JavaScript Objectes Integrats



Índex

OBJECTES

Un **objecte** és una representació detallada, concreta i particular de *quelcom*. Aquesta representació determina la seva identitat, el seu estat i el seu comportament en un moment donat.

La identitat d'un objecte li permet ser distingit d'entre altres i això es dóna gràcies a l'adreça de memòria, són diferents si ocupen diferents adreces de memòria.

L'estat d'un objecte és el conjunt de valors concrets que el caracteritzen en un moment donat, com pes, color, preu, etc...

El comportament defineix un conjunt de funcions que l'objecte és capaç de portar a terme. Aquestes funcions poden estar relacionades entre sí, modificar l'estat de l'objecte o fer crides a funcionalitats d'altres objectes, entre altres coses...

Una classe es defineix com la generalització d'un objecte particular. És a dir, una classe representa a una família d'objectes concrets.

Una instància d'una classe és un objecte en particular.

Es pot tenir, per exemple, una classe anomenada **Gos** amb determinades propietats i comportaments. Una vegada creada aquesta classe es pot fer una instància de la mateixa anomenada *"Fido"* que tindrà certes propietats específiques i els comportaments de la classe.

```
definir classe Gos:
    propiedad sexe en {mascle, femella}
    propiedad color en {blanc, marró, negre, mixte}
    propiedad cansanci en (res, poc, moderat, molt, exhauste)
         ... altres propietats ...
    mètode passeig() retorna cansanci
    mètode quiet() retorna res
    mètode mort() retorna res
    mètode menjar(tipus_menjar) retorna res
         ... altres comportaments ...

crear Fido:
 mètodo cercar() retorna branca
 propietat vacunes col.lecció
 Fido.sexe = mascle
 Fido.color = marró

crear Princessa:
 Princessa.sexe = femella
 Princessa.color = blanc
```

Un objecte té **propietats** i **mètodes**.

PROPIETATS

Les propietats són els valors associats a un objecte.

A l'exemple següent s'utilitza la propietat *length* de l'objecte String per retornar el nombre de caràcters que hi ha en una cadena:

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

  var text="Hola Món!"
  document.write(text.length)

</script>
```

El resultat seria: 9

MÈTODES

Els mètodes són les accions que es poden fer amb un objecte.

En el següent exemple s'utilitza el mètode *toUpperCase()* de l'objecte String per mostrar una cadena en lletres majúscules.

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

   var text="Hola Món!"
  document.write(text.toUpperCase())

</script>
```

El resultat seria: HOLA MÓN!

L'OBJECTE STRING

L'objecte **String** s'utilita per a treballar amb cadenes de caràcters. A continuació es poden veure alguna de les propietats i mètodes d'aquest objecte:

Propietat	Descripció
length	Retorna la longitut de la cadena

Mètode	Funció
anchor(name)	Crea una “ancla”
big()	Mostra la cadena de caràcters com una font gran
blink()	La cadena es representa de forma intermitent
bold()	Mostra la cadena en negreta
charAt(n)	Mostra el caràcter situat a la posició n de la cadena. Tots els arrays i strings en JavaScript comencen a la posició zero
charCodeAt(n)	Mostra el codi del caràcter situat a la posició n de la cadena. Tots els arrays i strings en JavaScript comencen a la posició zero
fixed()	Mostra la cadena en una font no proporcional
fontcolor(color)	Representa la cadena en un color determinat. El color pot ser cualsevol valor de color (RGB) vàlid
fontsize(n)	Mostra la cadena en un tamany de font determinat per n, que pot tenir un valor de 1 a 7
indexOf(smallstring,start)	Retorna la posició d'una part de la cadena. La cerca comença a la posició indicada
italics()	Representa la cadena en cursiva
lastIndexOf(chr,start)	Dóna com a resultat la última posició d'un caràcter. La cerca comença a la posició indicada
link(url)	Converteix la cadena en un hipervincle. El destí s'especifica amb una URL
small()	Representa la cadena en una font petita
split(char)	Retorna un array, usant com a separador de cada element de l'array el caràcter char.
strike()	Representa la cadena subratllada
sub()	Representa la cadena amb format de subíndex
substr(start,length)	Mostra una cadena parcial que comença a la posició start i una longitud length. El primer caràcter de la cadena està a la posició 0
substring(x,y)	Mostra una cadena parcial que comença a la posició X i acaba a la posición Y. El primer caràcter de la cadena està a la posició 0
sup()	Representa la cadena amb format de superíndex
toLowerCase()	Mostra la cadena en minúscules

Mètode	Funció
toUpperCase()	Mostra la cadena en majúscules
toString()	Aquest mètode el tenen tots els objectes i s'utilitza per a conventir-los en cadenes

L'OBJECTE MATH

L'objecte **Math** permet realitzar operacions matemàtiques comuns. No es necessita definir l'objecte Math abans d'utilitzar-lo.

VALORS MATEMÀTICS

JavaScript proporciona 8 valors matemàtics (constants) que es poden referenciar de l'objecte Math:

```
Math.E
Math.PI
Math.SQRT2        arrel quadrada de 2
Math.SQRT1_2      arrel quadrada de 1/2
Math.LN2          logaritme natural de 2
Math.LN10         logaritme natural de 10
Math.LOG2E        logaritme en base 2 d'E
Math.LOG10E       logaritme en base 10 d'E
```

MÈTODES MATEMÀTICS

Mètode	Funció
abs(n)	Retorna el valor absolut de n
acos(n)	Retorna el arcocosinus de n
asin(n)	Retorna el arcosinus de n
atan(n)	Retorna l'arcotangent de n
ceil(n)	Retorna el menor valor enter que és igual o major que n
cos(n)	Retorna el cosinus de n
exp(n)	Retorna e elevat a n, on n és l'argument i e la constant de Euler
floor(n)	Retorna el major valor enter igual o menor que n
log(n)	Retorna el logaritme natural de n amb base e
max(x,y)	Retorna x o y, en funció de quin dels dos és major
min(x,y)	Retorna x o y, en funció de quin dels dos és menor
pow(x,y)	Retorna x, que juntament amb y determinen la potència del radical (xy)
random()	Retorna un número aleatori entre 1 i 0
round(n)	Retorna el valor n que ha estat arrodonit en el següent valor enter
sin(n)	Retorna el sinus de n
sqrt(n)	Retorna l'arrel quadrada de n
tan(n)	Retorna la tangent de n

El següent exemple utilitza el mètode round() per arrodonir un número al següent nombre enter:

```
document.write(Math.round(4.7))
```

El resultat seria: 5

El següent exemple utilitza el mètode random() per retornar un número aleatori entre 0 i 1:

```
document.write(Math.random())
```

Un possible resultat seria: 0.47163643055483806

El següent exemple utilitza els mètodes floor() i random() per retornar un número aleatori entre 0 i 10:

```
document.write(Math.floor(Math.random()*11))
```

Un possible resultat seria: 7

L'OBJECTE ARRAY

L'objecte **Array** s'utilitza per emmagatzemar una sèrie de valors en una sola variable.

Les propietats i mètodes més comuns són:

Propietat	Descripció
length	Retorna el número d'elements de l'array

Mètode	Funció
concat()	Agrupa dos o més arrays i retorna el resultat
join()	Agrupa tots els elements en una cadena de caracters. Els elements de la cadena estan separats per comes
pop()	Treu i retorna l'últim element d'un array
push()	Afegeix un o més elements al final d'un array i retorna la nova longitud
reverse()	Retorna la matriu en la seqüència invertida
sort()	Retorna la versió ordenada del vector

Per definir un nou array:

```
var nouArray=new Array()
```

Hi ha 2 maneres per informar els valors d'un array:

a.1)
```
var cotxes=new Array()
cotxes[0]="Saab"
cotxes[1]="Volvo"
cotxes[2]="BMW"
```

a.2) Si es vol especificar el nombre total d'elements que es poden afegir a l'array:

```
var cotxes=new Array(3)
cotxes[0]="Saab"
cotxes[1]="Volvo"
cotxes[2]="BMW"
```

b) `var cotxes=new Array("Saab","Volvo","BMW")`

ACCEDINT A UN ARRAY

Es pot accedir a un element particular d'un array indicant el nom de l'array i l'índex de la posició que es vol accedir. S'ha de recordar que el número de l'índex comença pel valor 0.

El següent codi:

```
document.write(mycars[0])
```

dóna com a resultat: Saab

MODIFICANT VALORS EN UN ARRAY EXISTENT

Per modificar el valor d'un element dins de l'array s'ha d'indicar el nou valor a guardar especificant l'índex de la posició on es vol emmagatzemar:

```
mycars[0]="Opel"
```

Per tant, si ara s'executa: `document.write(mycars[0])`
el resultat serà: Opel

L'OBJECTE DATE

L'objecte **Date** s'utilitza per a treballar amb dates i temps. Per definir un objecte de tipus Date es fa:

```
var unaData = new Date()
```

Aquesta nova variable **unaData** s'inicialitza amb la data i hora actual.

MANIPULACIÓ DE DATES

Es pot treballar fàcilment amb dates mitjançant els mètodes de l'objecte Date:

En el següent exemple es fixa una data determinada (14 de gener de 2014):

```
var unaData=new Date()
unaData.setFullYear(2010,0,14)
```

En el següent exemple es sumen 5 dies a la data actual:

```
var unaData=new Date()
unaData.setDate(unaData.getDate()+5)
```

COMPARANT DATES

L'objecte Date s'utilitza molt sovint per fer la comparació de dues dates.

Al següent exemple es compara la data d'avui amb la del 14 de gener del 2010:

```
var unaData=new Date()
unaData.setFullYear(2010,0,14)

var avui = new Date()

if (unaData>avui)
  alert("Avui és abans del 14 de Gener del 2010")
else
  alert("Avui és després del 14 de Gener del 2010")
```

Mètode	Funció
getDate()	Retorna el dia del mes. El valor és un número enter entre 1 i 31
getDay()	Retorna el dia de la setmana. El valor és un número enter entre 0 i 6, on 0 és diumege, 1 dilluns etc
getHours()	Retorna la hora amb un valor enter entre 0 i 23
getMinutes()	Retorna els minuts amb un valor enter entre 0 i 59
getMonth()	Retorna el mes del objeto date actual como un valor entero entre 0 y 11 (0 es enero, 1 febrero, etc.)
getSeconds()	Retorna los segundos del valor date actual como un valor entero de 0 a 59
getTime()	Retorna la hora amb un valor enter que representa els mil·lisegons que han transcorregut des de les 00:00:00 hores del 1 de gener de 1970
gelTimezoneOffset()	Retorna la diferència entre la hora local i la hora GTM. El valor és un enter que representa la diferència en minuts
getYear()	Retorna l'any amb un valor enter de dos dígits que representa l'any menys 1900
parse(datestring)	Retorna el número de mil·lisegons transcorreguts en l'espai de temps que va des de les 00:00:00 hores del 1 de gener de 1970 i la data especificada en datestring. La cadena datestring ha de tenir el format: dia, DD mes AAAA HH:MM:SS TZN o mes DD, AAAA.
setDate(dateValue)	Estableix el dia de mes. DateValue és un valor entero entre 1 i 31
setHours(hoursValue)	Determina la hora. HoursValue és un valor enter entre 0 i 23
setMinutes(minutesValue)	Estableix els minuts. La dada minutesValue és un valor enter entre 0 i 59
setMonth(monthValue)	Determina el mes. La dada monthValue és un valor enter entre 0 i 11, on 0 és gener, 1 febrer, etc
setSeconds(secondsValue)	Estableix els segons. El paràmetre secondsValue és un valor enter entre 0 i 59
setTime(timeValue)	Estableix el valor de l'objete date actual. El paràmetre timeValue és un valor enter que representa els mil·lisegons transcorreguts entre les 00:00:00 hores del dia 1 de gener de 1970 i la data desitjada
setYear(yearValue)	Determina l'any. El paràmetre yearValue és un valor enter major que 1900
oTMSString()	Retorna el valor de l'objecte date actual en GTM com un string. La cadena de caràcters ha de tenir el format: dia, DD mes AAAA HH:MM:SS GTM.
toLocaleString()	Retorna el valor de l'objecte date actual com un string.
UTC(yearValue, monthValue, dateValue, hoursValue, minutesValue, secondsValue)	Retorna la quantitat de mil·lisegons transcorregutss des de les 00:00:00 hores del 1 de gener de 1970. El paràmetre yearValue és un valor enter major que 1900, monthValue és un valor enter entre 0 i 11, dateValue és un enter entre 1 i 31, hoursValue és un enter entre 0 i 23, finalment minutesValue i secondsValue, opcionals, són valors enters entre 0 i 59.