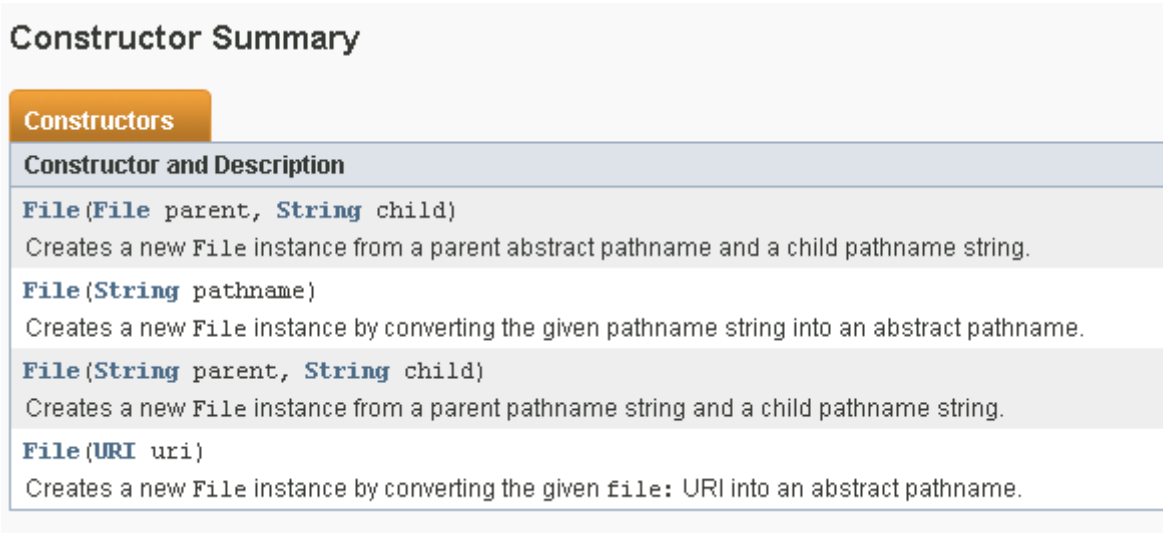
Clase File

La **clase File** representa un fichero o directorio de nuestro sistema de archivos. También nos es útil ya que puede pasarse como parámetro para las principales clases que manejan ficheros, como **FileReader, FileInputStream, FileWriter, etc.**

Podemos pasarle un String con la ruta del fichero, también hay otros constructores. Estos son:



Constructor Summary

Constructors

Constructor and Description
`File(File parent, String child)` Creates a new `File` instance from a parent abstract pathname and a child pathname string.
`File(String pathname)` Creates a new `File` instance by converting the given pathname string into an abstract pathname.
`File(String parent, String child)` Creates a new `File` instance from a parent pathname string and a child pathname string.
`File(URI uri)` Creates a new `File` instance by converting the given `file:` URI into an abstract pathname.

Esta clase está en el paquete **java.io.**

A diferencia de otras clases que manejan ficheros, en la clase **File** no es necesario controlar excepciones en la mayoría de los métodos. Si hay alguna excepción a manejar lo indicaremos.

Estos son los métodos más conocidos:

NOMBRE	DESCRIPCIÓN	PARÁMETROS	DATO DEVUELTO
exists	Indica si existe o no el fichero.	Ninguno.	boolean
isDirectory	Indica si el objeto File es un directorio.	Ninguno.	boolean
isFile	Indica si el objeto File es un fichero.	Ninguno.	boolean
isHidden	Indica si el objeto File está oculto.	Ninguno.	boolean
getAbsolutePath	Devuelve una cadena con la ruta absoluta del fichero o directorio.	Ninguno.	String
canRead	Indica si se puede leer.	Ninguno.	boolean
canWrite	Indica si se puede escribir.	Ninguno.	boolean
canExecute	Indica si se puede ejecutar.	Ninguno.	boolean
getName	Devuelve una cadena con el nombre del fichero o directorio.	Ninguno.	String
getParent	Devuelve una cadena con el directorio padre.	Ninguno.	String
listFiles	Devuelve un array de File con los directorios y ficheros hijos.	Ninguno.	Array de File

NOMBRE	DESCRIPCIÓN	PARÁMETROS	DATO DEVUELTO
list	Devuelve un array de String con los directorios hijos. Solo funciona con directorios.	Ninguno.	Array de String
mkdir	Permite crear el directorio en la ruta indicada. Solo se creara si no existe.	Ninguno.	boolean
mkdirs	Permite crear el directorio en la ruta indicada, también crea los directorios intermedios. Solo se creara si no existe.	Ninguno.	boolean
createNewFile	Permite crear el fichero en la ruta indicada. Solo se creará si no existe. Debemos controlar la excepcion con IOException.	Ninguno.	boolean

Os dejo un ejemplo, usando los métodos:

```
1  import java.io.File;
2  import java.io.IOException;
3  public class EjemploFileApp {
4
5      public static void main(String[] args) throws IOException {
6
7          //Creamos objetos File
8          File fichero=new File("D:\\fich_binario.ddr");
9          File fichero2=new File("D:\\fichero.txt");
10         File directorio=new File("D:\\prueba");
11         File directorio2=new File("D:\\directorio");
12
13                 //Creo los ficheros y directorios
14                 fichero.createNewFile();
15                 fichero2.createNewFile();
16                 directorio.mkdir();
17                 directorio2.mkdir();
18
19                 //Indica si existen los archivos
20         System.out.println("Existencia: ");
21         System.out.println("Fichero "+fichero.exists());
22         System.out.println("Directorio "+directorio.exists());
23
24         System.out.println("");
25
26         //Indica si son directorios
27         System.out.println("¿Son directorios?: ");
28         System.out.println("Fichero "+fichero.isDirectory());
29         System.out.println("Directorio "+directorio.isDirectory());
30
31         System.out.println("");
32
           //Indica si son ficheros
           System.out.println("¿Son ficheros?: ");
           System.out.println("Fichero "+fichero.isFile());
           System.out.println("Directorio "+directorio.isFile());

           System.out.println("");
```

```
33
34          //Indica la ruta absoluta del fichero o directorio
35          System.out.println("Ruta absoluta: ");
36          System.out.println("Fichero "+fichero.getAbsolutePath());
37          System.out.println("Directorio "+directorio.getAbsolutePath());
38
39          System.out.println("");
40
41          //Indica si se puede leer
42          System.out.println("¿Se pueden leer?:");
43          System.out.println("Fichero "+fichero.canRead());
44          System.out.println("Directorio "+directorio.canRead());
45
46          System.out.println("");
47
48          //Indica si se puede escribir
49          System.out.println("¿Se pueden escribir?:");
50          System.out.println("Fichero "+fichero.canWrite());
51          System.out.println("Directorio "+directorio.canWrite());
52
53          System.out.println("");
54
55          //Indica si se puede ejecutar
56          System.out.println("¿Se pueden ejecutar?:");
57          System.out.println("Fichero "+fichero.canExecute());
58          System.out.println("Directorio "+directorio.canExecute());
59
60          System.out.println("");
61
62          //Indica el nombre sin rutas
63          System.out.println("Nombres sin rutas: ");
64          System.out.println("Fichero "+fichero.getName());
65          System.out.println("Directorio "+directorio.getName());
66
67          System.out.println("");
68
69          //Indica el nombre del directorio padre
70          System.out.println("Nombre del directorio padre: ");
71          System.out.println("Fichero "+fichero.getParent());
72          System.out.println("Directorio "+directorio.getParent());
73
74          System.out.println("");
75
76          //Guarda en un array de File los directorios hijos, solo con directorios
77          System.out.println("Nombre de los objetos File dentro de un array");
78          File lista[]=directorio.listFiles();
79          for(int i=0;i<lista.length;i++){
80              System.out.println(lista[i]);
81          }
82
83          System.out.println("");
84
85          //Guarda en un array de String los directorios hijos, solo con directorios
86          System.out.println("Nombre de los objetos String dentro de un array");
87          String listaString[]=directorio.list();
88          for(int i=0;i<listaString.length;i++){
89              System.out.println(listaString[i]);
90          }
91
92          System.out.println("");
93
94
```

```
83      }
84  }
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
```

La clase File tiene un par de constantes muy útiles, que nos servirán si usamos nuestra aplicación en varios sistemas operativos. Nos permitirá mostrar el separador entre ficheros entre rutas de cada sistema operativo, por ejemplo, para Windows, se separan las rutas con; y los directorios con \.
Los métodos son **pathSeparator** y **separator.**

Se invocan con el nombre de la clase, por ejemplo, **File.separator.**