**Класс File**

В отличие от большинства классов ввода/вывода, класс **File** работает не с потоками, а непосредственно с файлами. Данный класс позволяет получить информацию о файле: права доступа, время и дата создания, путь к каталогу. А также осуществлять навигацию по иерархиям подкаталогов.

Класс **java.io.File** может представлять имя определённого файла, а также имена группы файлов, находящихся в каталоге. Если класс представляет каталог, то его метод **list()**возвращает массив строк с именами всех файлов.

Для создания объектов класса **File** можно использовать один из следующих конструкторов.

* File(File dir, String name) - указывается объекта класса **File** (каталог) и имя файла
* File(String path) - указывается путь к файлу без указания имени файла
* File(String dirPath, Sring name) - указывается путь к файлу и имя файла
* File(URI uri) - указывается объекта **URI**, описывающий файл

**Методы класса File**

Класс **File** может использоваться для создания каталога или дерева каталогов. Также можно узнать свойства файлов (размер, дату последнего изменения, режим чтения/записи), определить к какому типу (файл или каталог) относится объект **File**, удалить файл. У класса очень много методов, перечислим некоторые.

* **getAbsolutePath()** - абсолютный путь файла, начиная с корня системы. В Android корневым элементом является символ слеша (/)
* **canRead()** - доступно для чтения
* **canWrite()** - доступно для записи
* **exists()** - файл существует или нет
* **getName()** - возвращает имя файла
* **getParent()** - возвращает имя родительского каталога
* **getPath()** - путь
* **lastModified()** - дата последнего изменения
* **isFile()** - объект является файлом, а не каталогом
* **isDirectory** - объект является каталогом
* **isAbsolute()** - возвращает *true*, если файл имеет абсолютный путь
* **renameTo(File newPath)** - переименовывает файл. В параметре указывается имя нового имени файла. Если переименование прошло неудачно, то возвращается *false*
* **delete()** - удаляет файл. Также можно удалить **пустой** каталог

**Каталог**

Каталог - это объект класса **File**, который содержит список других файлов и каталогов. После создания объекта класса **File**, являющего каталогом, его метод **isDirectory()** вернёт значение *true*. И тогда вы можете вызывать метод **list()**.

Для создания каталога можно использовать метод **mkdir()**, который вернёт *true* в успешном случае. Если указанный путь уже существует или каталог нельзя создать из-за отсутствия полного пути к нему, то вернётся *false*.

Метод **mkdirs()** создаёт сам каталог, так и всех его родителей.

**Список каталогов**

Если вы хотите получить содержимое каталога, то можно вызвать метод **list()** без аргументов. Вам вернётся полный список (массив) **имён** файлов и каталогов, содержащихся в данном каталоге. Есть ещё похожий метод **listFiles()**, который возвращает массив файлов (объектов, а не их имён), о котором поговорим отдельно.

String dirPath = "/";

File f = new File(dirPath);

File[] files = f.listFiles();

Если вам нужен только список файлов с расширением .java, то можно использовать специальный фильтр-класс, который описывает критерии отбора объектов **File** с помощью интерфейса **FilenameFilter**.

String[] list(FilenameFilter filter)

Интерфейс **FilenameFilter** определяет единственный метод **accept()**, вызываемый по одному разу с каждым файлом в списке.

boolean accept(File dir, String filename)

Метод возвращает *true* для файлов каталога, указанного в **dir**, которые должны быть включены в список, и *false* - для файлов, которые следует исключить.

**Метод listFiles()**

Метод имеет несколько перегруженных версий. Он возвращает список файлов в виде массива объектов класса **File**. Одна из версий метода также использует интерфейс **FilenameFilter**. Другая версия использует интерфейс **FileFilter** и возвращает те файлы, путевые имена которых соответствует интерфейсу.

Пример использования методов класса **File** показан в уроке [Файловый менеджер](http://developer.alexanderklimov.ru/android/fileexplorer.php) и в других примерах.