Android 内部存储

一、简介

android有一套自己的安全模型，当应用程序(.apk)在安装时系统就会分配给他一个userid，当该应用要去访问其他资源比如文件的时候，就 *需要userid匹配* 。默认情况下，任何应用创建的文件，sharedpreferences，数据库都应该是私有的-位于**/data/data/包名/files**，其他程序无法访问，只能被本应用程序所访问。除非在创建时指定了Context.MODE_WORLD_READABLE或者Context.MODE_WORLD_WRITEABLE ，只有这样其他程序才能正确访问。Context.MODE_PRIVATE：为默认操作模式，代表该文件是私有数据，只能被应用本身访问，在该模式下，写入的内容会覆盖原文件的内容，如果想把新写入的内容追加到原文件中。可以使用Context.MODE_APPEND。

应用程序的私有文件即为形如 **data/data/com.snajdan.demo** 的文件路径，这也是内部存储空间的路径，可以理解为创建于应用内部存储空间的文件。

二、内部存储空间重要函数

openFileOutput()

新建的文件位于在 **data/data/com.snajdan.demo/files**目录下

```
 /**
  * Open a private file associated with this Context's application package
  * for writing.  Creates the file if it doesn't already exist.
  *
  * @param name The name of the file to open; can not contain path
  *             separators.
  * @param mode Operating mode.  Use 0 or {@link #MODE_PRIVATE} for the
  * default operation, {@link #MODE_APPEND} to append to an existing file,
  * {@link #MODE_WORLD_READABLE} and {@link #MODE_WORLD_WRITEABLE} to control
  * permissions.
  *
  * @return FileOutputStream Resulting output stream.
  *
  * @see #MODE_APPEND
  * @see #MODE_PRIVATE
  * @see #MODE_WORLD_READABLE
  * @see #MODE_WORLD_WRITEABLE
  * @see #openFileInput
  * @see #fileList
  * @see #deleteFile
  * @see java.io.FileOutputStream#FileOutputStream(String)
  */
 public abstract FileOutputStream openFileOutput(String name, int mode)
     throws FileNotFoundException;
```

File getCacheDir ()

Returns the path of the directory holding application cache files.

返回的目录路径是 **/data/data/com.snajdan.demo/cache**

该目录主要用于存放缓存文件，当系统的内存存储空间紧张时，该目录下的文件会被删除掉。关于这些文件究竟会在存储空间剩余多少的情况，没有严格的标准保障。

注意：你不应该依赖系统来清理这些缓存文件，你应该对这些缓存文件占用的最大存储空间设定个最大值，比如是1M，当实际占用空间超过这个值时，你应该对这些缓存文件做相应的清理工作

```
/**
    * Returns the absolute path to the application specific cache directory
    * on the filesystem. These files will be ones that get deleted first when the
    * device runs low on storage.
    * There is no guarantee when these files will be deleted.
    *
```

```
     * <strong>Note: you should not <em>rely</em> on the system deleting these
     * files for you; you should always have a reasonable maximum, such as 1 MB,
     * for the amount of space you consume with cache files, and prune those
     * files when exceeding that space.</strong>
     *
     * @return Returns the path of the directory holding application cache files.
     *
     * @see #openFileOutput
     * @see #getFileStreamPath
     * @see #getDir
     */
    public abstract File getCacheDir();
```

File getDir ()

该函数主要用于得到一个文件夹的句柄，并通过该句柄创建和访问外文件夹。

注意：参数int mode是指文件夹的访问权限而并不包括其子文件夹和文件的访问权限

```
    /**
     * Retrieve, creating if needed, a new directory in which the application
     * can place its own custom data files.  You can use the returned File
     * object to create and access files in this directory.  Note that files
     * created through a File object will only be accessible by your own
     * application; you can only set the mode of the entire directory, not
     * of individual files.
     *
     * @param name Name of the directory to retrieve.  This is a directory
     * that is created as part of your application data.
     * @param mode Operating mode.  Use 0 or {@link #MODE_PRIVATE} for the
     * default operation, {@link #MODE_WORLD_READABLE} and
     * {@link #MODE_WORLD_WRITEABLE} to control permissions.
     *
     * @return Returns a File object for the requested directory.  The directory
     * will have been created if it does not already exist.
     *
     * @see #openFileOutput(String, int)
     */
    public abstract File getDir(String name, int mode);
```

getDir函数的返回路径比较特殊，如下代码：

```
File file = this.getDir("download", Context.MODE_PRIVATE);
path = file.getAbsolutePath();
```

所返回的路径是：/data/data/com.snajdan.demo/app_downlaod,即通过调用getDir(name,mode)函数，会创建一个位于私有文件目录下app_name的文件目录

File getFileStreamPath ()

该函数返回的是之前用openFileOutput所创建的文件，路径都在**data/data/com.snajdan.demo/files**下面

```
public abstract File getFileStreamPath(String name);
```

File getFilesDir ()

该函数返回的是通过openFileOutput所创建的文件目录，即**data/data/com.snajdan.demo/files**

```
/**
     * Returns the absolute path to the directory on the filesystem where
     * files created with {@link #openFileOutput} are stored.
     *
     * @return Returns the path of the directory holding application files.
     *
     * @see #openFileOutput
```

```
     * @see #getFileStreamPath
     * @see #getDir
     */
    public abstract File getFilesDir();
```

Environment中还包含有三个内部存储函数：

```
public static File getDataDirectory ()
```

用File返回数据文件的根目录，返回的文件的路径为“/data”。该目录下的文件是只读。应用程序无法对该目录下的文件进行写操作。

```
public static File getDownloadCacheDirectory ()
```

用File返回缓存文件的根目录，返回的文件的路径为“/cache”。对于第三方应用程序。该目录下的文件是只读。第三方应用程序无法对该目录下的文件进行写操作。

```
public static File getRootDirectory ()
```

用File返回Android系统文件的根目录，返回的文件的路径为“/system”。该目录下的文件是只读。应用程序无法对该目录下的文件进行写操作。

三、总结

内部存储位于系统中很特殊的一个位置，如果你想将文件存储于内部存储中，那么文件默认只能被你的应用访问到，且一个应用所创建的所有文件都在和应用包名相同的目录下。也就是说应用创建于内部存储的文件，与这个应用是关联起来的。当一个应用卸载之后，内部存储中的这些文件也被删除。从技术上来讲如果你在创建内部存储文件的时候将文件属性设置成可读，其他app能够访问自己应用的数据，前提是他知道你这个应用的包名，如果一个文件的属性是私有（private），那么即使知道包名其他应用也无法访问。 内部存储空间十分有限，因而显得可贵，另外，它也是系统本身和系统应用程序主要的数据存储所在地，一旦内部存储空间耗尽，手机也就无法使用了。所以对于内部存储空间，我们要尽量避免使用。Shared Preferences和SQLite数据库都是存储在内部存储空间上的。内部存储一般用Context来获取和操作。