FILE SYSTEM

A *file system* handles the persistent storage of data files, apps, and the files associated with the operating system itself. Therefore, the file system is one of the fundamental resources used by all processes.

1. IOS FILE SYSTEM

The iOS file system is geared toward apps running on their own. To keep the system simple, users of iOS devices do not have direct access to the file system and apps are expected to follow this convention.

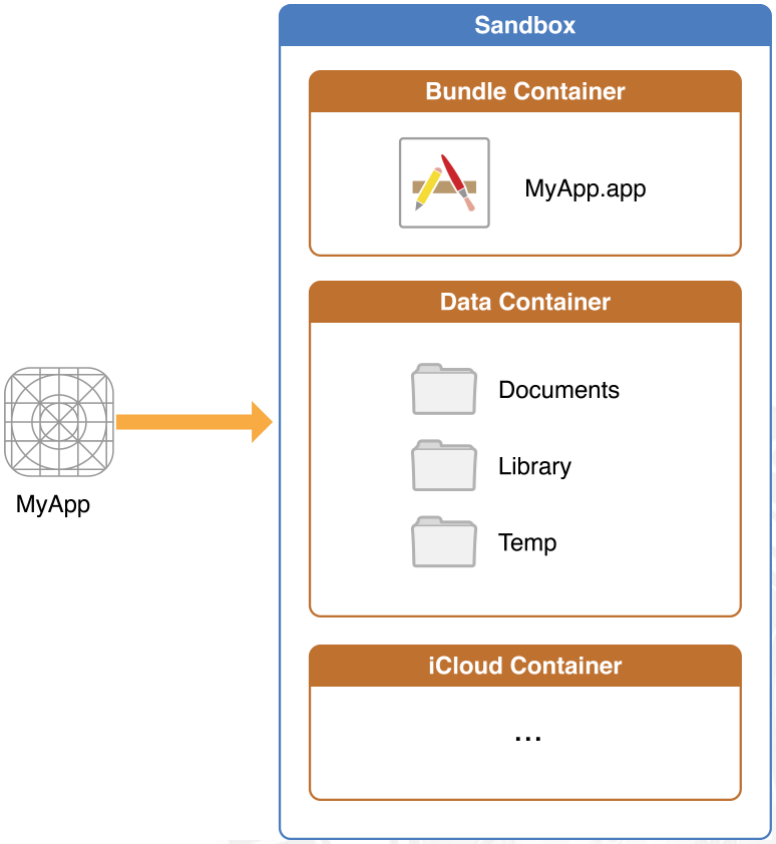
iOS Standard Directories: Where Files Reside

For security purposes, an iOS app's interactions with the file system are limited to the directories inside the app's sandbox directory. During installation of a new app, the installer creates a number of container directories for the app inside the sandbox directory. Each container directory has a specific role.

The bundle container directory holds the app's bundle, whereas the data container directory holds data for both the app and the user. The data container directory is further divided into a number of subdirectories that the app can use to sort and organize its data. The app may also request access to additional container directories—for example, the iCloud container—at runtime.

These container directories constitute the app's primary view of the file system. Figure shows a representation of the sandbox directory for an app.

Figure An iOS app operating within its own sandbox directory



The following Table lists some of the more important subdirectories inside the sandbox directory and describes their intended usage. This table also describes any additional access restrictions for each subdirectory and points out whether the directory's contents are backed up by iTunes and iCloud.

Table 1-1 Commonly used directories of an iOS app

Directory	Description
AppName.app	This is the app's bundle. This directory contains the app and all of its resources. You cannot write to this directory. To prevent tampering, the bundle directory is signed at installation time. Writing to this directory changes the signature and prevents your app from launching. You can, however, gain read-only access to any resources stored in the apps bundle. For more information, see the *Resource Programming Guide*

	The contents of this directory are not backed up by iTunes or iCloud. However, iTunes does perform an initial sync of any apps purchased from the App Store.
Documents/	Use this directory to store user-generated content. The contents of this directory can be made available to the user through file sharing; therefore, this directory should only contain files that you may wish to expose to the user. The contents of this directory are backed up by iTunes and iCloud.
Documents/Inbox	Use this directory to access files that your app was asked to open by outside entities. Specifically, the Mail program places email attachments associated with your app in this directory. Document interaction controllers may also place files in it. Your app can read and delete files in this directory but cannot create new files or write to existing files. If the user tries to edit a file in this directory, your app must silently move it out of the directory before making any changes. The contents of this directory are backed up by iTunes and iCloud.
Library/	This is the top-level directory for any files that are not user data files. You typically put files in one of several standard subdirectories. iOS apps commonly use the Application Support and Caches subdirectories; however, you can create custom subdirectories. Use the Library subdirectories for any files you don't want exposed to the user. Your app should not use these directories for user data files. The contents of the Library directory (with the exception of the Caches subdirectory) are backed up by iTunes and iCloud.

	For additional information about the Library directory and its commonly used subdirectories, see The Library Directory Stores App-Specific Files.
tmp/	Use this directory to write temporary files that do not need to persist between launches of your app. Your app should remove files from this directory when they are no longer needed; however, the system may purge this directory when your app is not running. The contents of this directory are not backed up by iTunes or iCloud.

An iOS app may create additional directories in the Documents, Library, and tmp directories. You might do this to better organize the files in those locations

2. ANDROID FILE SYSTEM.

(i) Flash Memory Android File System

1. exFAT

Originally created by Microsoft for flash memory, the exFAT file system is not a part of the standard Linux kernel. However, it still provides support for Android devices in some cases. It stands for Extended File Allocation Table.

2.F2FS

Users of Samsung smartphones are bound to have come across this type of file system if they have been using the smartphone for a while. F2FS stands for Flash-Friendly File System, which is an Open Source Linux file system. This was introduced by Samsung 4 years ago, in 2012.

3. JFFS2

It stands for the Journal Flash File System version 2. This is the default flash file system for the Android Open Source Project kernels. This version of Android File System has been around since the Android Ice Cream Sandwich OS was released. JFFS2 has since replaced the JFFS.

4. YAFFS2

It stands for Yet Another Flash File System version 2. Funny as the name might sound like, it is actually a serious business! It has not been a part of the AOSP for a while now and is rarely found in Android smartphones. However, it does tend to make a few appearances every now and then.

(ii) Media-based Android File System

1. EXT2/EXT3/EXT4

Ext, which stands for the EXTended file systems, are the standards for the Linux file system. The latest out of these is the EXT4, which has now been replacing the YAFFS2 and the JFFS2 file systems on Android smartphones.

2. MSDOS

Microsoft Disk Operating System is known to be one of the oldest names in the world of Operating Systems, and it helps FAT 12, FAT 16 and FAT 32 file systems to run.

3. VFAT

An extension to the aforementioned FAT 12, FAT 16 and FAT 32 file systems, the VFAT is a kernel module seen alongside the MSDOS module. External SD cards that help expand the storage space are formatted using VFAT.

(iii) Pseudo File Systems

1. CGroup

Cgroup stands for Control Group. It is a pseudo file system which allows access and meaning to various kernel parameters. Cgroups are very important for the Android File System as the Android OS makes use of these control groups for user accounting and CPU Control.

2. Rootfs

Rootfs acts as the mount point, and it is a minimal file system. It is located at the mount point "/".

3. Procfs

Usually found mounted at the /proc directory. The procfs file system has files which showcase the live kernel data. Sometimes this file system also reflects a number of kernel

data structures. These number directories are reflective of the process IDs for all the currently running tasks.

4. Sysfs

Usually mounted on the /sys directory. The sysfs file system helps the kernel identify the devices. Upon identifying a new device, the kernel builds an object in sys/module/ directory. There are various other elements stored inside the /sys/ folder which helps the kernel communicate with various Android File Systems.

5. Tmpfs

A temporary file system, tmpfs is usually mounted on /dev directory. Data on this is lost when the device is rebooted.