HCLTech | Supercharging Progress™

Custom USB access solution for Android devices

Table of contents

Introduction

As Android versions advance, heightened security measures are pivotal, impacting access to Universal Serial Bus (USB) external devices. Security enforcement, especially on Android TV android Auto and Android Things (IoT) platforms, imposes restrictions on USB file system access, particularly for devices not predefined in Android's OEM list or custom USB devices. The Storage Access Framework (SAF) is instrumental in enhancing security by channelling file operations through content URIs instead of direct file paths. However, this introduces challenges for applications necessitating low-level file operations and restricts direct file modifications in external storage directories. To address these constraints, the proposed USB Custom Framework emerges as a solution. This framework offers diverse Application Programming Interfaces (APIs) for file access, read, write, etc., enabling Android apps to interact with USB devices seamlessly, regardless of OEM or custom specifications, bridging the gap created by Android's evolving security measures across various platforms.

Problem

Android platforms have become increasingly popular. One common requirement for Android applications is to access files stored on external USB devices.

The restriction is part of Android's ongoing efforts to enhance user privacy and security by providing a more controlled and user-friendly file access mechanism.

As different Android devices have variations in their file system paths and USB access will vary based on Android versions and manufacturer/OEM, for instance, the below are some potential issues w.r.t Android platform

- Storage Access Framework (SAF) was not officially supported on Android TV android Things (IoT) and Android Auto platforms
- Manufacturer/OEM USB file system path restrictions
- No direct platform APIs in the Android platform

Solution

The USB custom framework has a tiny web server and a network service utilised to enable file handling from USB storage. This framework offers diverse APIs to access files stored on external USB devices, allowing Android apps to interact with USB devices.

The framework has four major components:

API layer: APIs to access files stored on external USB devices

Webserver: Network service to enable file accessing from USB storage

Storage module: Accessing and providing HTTP URLs to the clients via API

Device interface: Device interaction and communication channelling for various Android platform devices (ex: TV/IoT/Auto).

The combination of the above components provides a unique and easy way of overcoming the Android platform restrictions and ensuring seamless integration with USB storage across various environments like Android TV android Auto and Android Things (IoT).

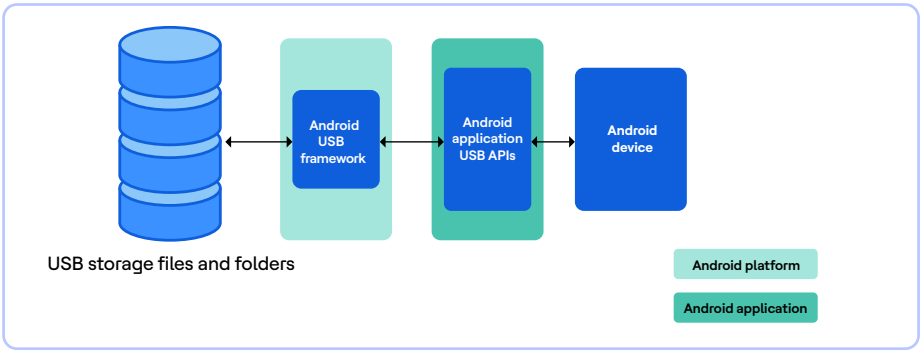
Some APIs of USB custom framework:

Void start()
Void start (int timeout, int port)
Void stop ()
Boolean wasStarted ()
Boolean isAlive ()
String getFile ()
String getExternalStoragePath ()
Boolean setExternalStoragePath (String path)
Static String getMimeTypeForFile (String URL)

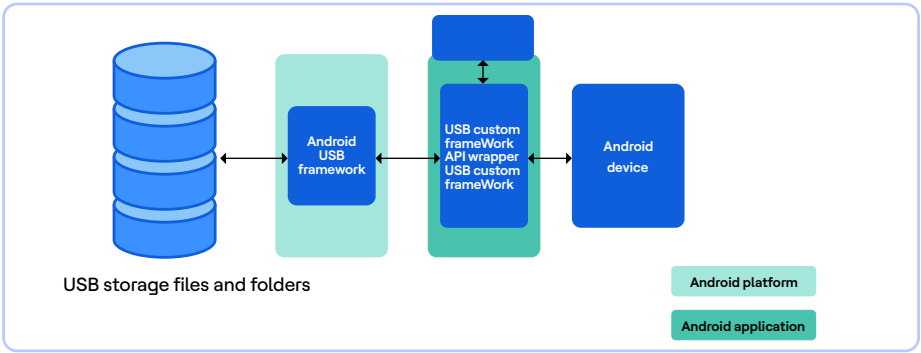
Adopting the USB custom framework:

To access USB contents using the USB custom framework solution in Android, follow these steps:

- Include the library in the Android project
- Customize the required parameters of the framework as per the framework requirements
- Access the required APIs as per the application need



Standard USB framework solution for Android devices



USB custom framework solution for Android devices [Architecture]

Play store links with an application developed by us using the USB custom framework solution-

Google Play Store:

https://play.google.com/store/apps/details?id=com.shiksha.TVPlayer

Amazon Play Store:

https://www.amazon.com/gp/product/B0CLYHFZHR

The table below explains about current limitations of Android file access and the merits of USB custom framework for various features.

Aspect	Android file access (Limitations)	USB custom framework with Android
Scoped storage	Limited access to designated directories due to scoped storage.	Can serve files from any accessible location, including root directories, bypassing scoped storage limitations.
User consent	Requires explicit user consent for USB file access within the app.	Users access files through the custom framework APIs, eliminating the need for explicit consent within the app.
Limited root directories access	Restricted by default to the root directories of external storage.	It provides access to root directories or specific directories based on your configuration.
File URI deprecation	Deprecates ` file://` URIs in favour of the SAF	Serves files via framework APIs, avoiding reliance on ` file://` URIs.
USB disconnect handling	Standard Android file access may not handle USB disconnection gracefully and the platform expects to handle it in Android applications.	Handles gracefully by using appropriate API.
Storage space management	Not handled by default. Basic storage management practices need to be implemented.	Allows you to implement file management logic for better storage space management.

Use case

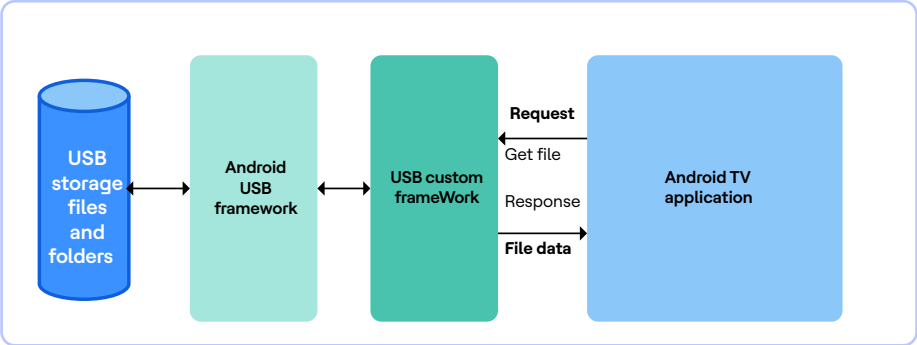
Using a local server to expose USB files to HTML Player:

Problem

Traditional methods of sharing files over a network can be cumbersome, especially when dealing with USB devices. USB drives are usually designed

for direct file storage and access rather than network sharing. When users want to share files stored on a USB drive, they often resort to physically transferring the drive to another computer or using third-party software, which can be inconvenient and time-consuming. Additionally, traditional file-sharing methods usually require a complex setup, making them unsuitable for quick file-sharing in a local network environment.

Solution



USB custom framework, can be leveraged to facilitate the easy sharing of USB files over a local host network. By integrating a USB custom framework into an Android application, users can create a local web server on the host machine that serves files from the USB drive. This allows users on the same network to access and download files from the USB drive via a web browser without the need for additional software or physical transfers.

This solution provides a simple and convenient way to share files stored on a USB drive over a local network using a USB custom framework. It eliminates the need for physical transfers or complex setups, making file sharing more accessible and efficient.

Features supported

Accessing USB files through the framework offers several benefits, making it a practical and efficient solution for Android applications. In addition to providing access to USB custom devices, it has some more key advantages, including:

1. **Easy file access:** Creating endpoints that serve files directly from the USB storage simplifies the process of accessing and serving files, eliminating the need for complex file-handling procedures.
2. **Security:** By creating access control mechanisms and authentication to restrict who can access the USB files, adding a layer of protection to the file-sharing solution.
3. **Streaming support:** Framework architecture allows for efficient file streaming, enabling smooth media playback directly from USB storage.
4. **Multi-device support:** This solution can be applied across different versions and devices of Android like TV, Auto and IoT devices etc.

Conclusion

The USB custom framework emerges as a solution, facilitating efficient file access from USB storage for Android applications. Specifically addressing the challenges posed by security restrictions on USB file system access in Android TV android Auto and Android Things (IoT) platforms, this paper offers a comprehensive overview of the USB custom framework. This approach empowers developers to create robust, user-friendly applications tailored for Android platforms, ensuring seamless integration with USB storage across various contexts, including Android Auto and Android Things (IoT) environments.

This solution can also be extended to support features like streaming enhancements and access to USB contents using standard browsers, as well as necessary security measures.

References

List of references to cite all sources used in this whitepaper.

- https://developer.android.com/training/tv/start
- https://developer.android.com/training/data-storage/shared/documents-files

Author information

Avrama Chandrasekhar

Chandra Sekhar holds a Master's degree in Electronics and has over 15 years of experience in Android platform engineering. His expertise spans the Android framework and system services, system applications, custom launchers and window-management design (including multi-window and desktop modes), Android security and permissions, Android enterprise with work-profile deployments and end-to-end AOSP device integration across diverse form factors.

His advanced focus areas include system services and Binder IPC (designing, extending and debugging framework services), as well as deep customization of the Android system UI (status bar, navigation bar, launcher, notifications and quick settings). Additionally, he has expertise in Android window management using WMS/ATMS and task/display stacks. He also has substantial experience with the AOSP build system and modularization (Soong, Make, product and device configuration, system image builds), as well as platform testing using CTS, XTS, CTS Verifier and automated Tradefed/atest workflows for comprehensive system validation.

Vijayakrishna S

Vijaya Krishna joined HCLTech in 2009 and has over 15 years of experience in the consumer electronics domain. He holds a bachelor's degree in Electrical and Electronics Engineering and a master's degree in Power Electronics from JNTU Hyderabad. He has worked on various projects, including the development of DTV software for Linux and Android. He also worked on the development of the Multi-Functional Printer software.

Has good knowledge of Android customization of mobiles, tablets and various consumer electronics devices in platform, framework and application layers. He has addressed numerous problems related to DTV middleware, application performance and user data access/control of consumer devices. At HCLTech, he is part of the ERS CU Digital Engineering team, working on various consumer products. He is passionate about the ease of usage and performance improvement of consumer products.

At HCLTech, he is part of the ERS CU Digital Engineering team, working on various consumer products. He is passionate about the ease of usage and performance improvement of consumer products.

Author information

Dayanand Sivasubramaniyan

Dayanand S joined HCLTech in 2004 and has over 20 years of experience in the consumer electronics,aerospace and semiconductor domains.
He holds a bachelor's degree in Electronics and Communication Engineering and a master's degree in Software Engineering (BITS)
He has worked on various projects, including DTV and mobile software development for Linux and Android. He has good experience in program management of TransferJetX Product, Aero Cabin experience program, Shiksha Program, etc.

At HCLTech, he is part of the ERS CU-EMBENGG team working across various consumer products. He is passionate about the ease of usage and better UX for consumer products.

Venkat Nemali

Venkat Nemali joined HCLTech in 2010 and has over 25+ years of experience in the consumer electronics domain and Industrial Automation. He holds a Master's degree in Electronics Instrumentation from NIT Warangal. He has worked on various projects related to STB, DTV and other consumer devices, spanning Linux/ RTOS/Android across the layers from platform to middleware and application and has addressed numerous problems and also experienced working on distributed control systems automation for power and petrochemical industries.

His key roles would include handling requirement specifications, providing technical guidance to the team to drive system-level issues and performance improvements and aligning with delivery teams for technical support on new proposals/business expansions. At HCLTech, he is part of the ERS CU Digital Engineering team, working on various consumer products.