

# How to Create a USB Liberty Library

## What is a “Liberty Library”?

A *Liberty Library* is a portable collection of audio, video and other files and content that can be quickly added to a Pirate Stick or Pirate Box on a USB storage device. It is called a Liberty Library (LL from now on) because with the help of people just like you, together we can save the information we treasure from the forces that want to see the destruction of the second “Library of Alexandria” and defeat the rampant censorship that seeks to keep people ignorant.

## Who is the Intended Audience of This Document?

Currently there is no Grandma “just” button to do this. It is much easier on the Pirate Stick than the Pirate Box. This preliminary document outlines the general concept of how to create a LL. Since the scope of potential content is so wide, the goal is to give tech savvy users and software developers an adequate description so they can create their own versions with their custom content. I intend to make the process of creating Liberty Libraries easier in future releases.

## What is IPFS?

Short for Inter-Planetary-File-System, a distributed and decentralized world-wide network of millions of computers that form the backbone of the Dweb (Decentralized Websites).

## Ok, so how do I create my own Liberty Library?

This document explains the structure for LLs and how they work. Once you understand that, you should be able to create that structure using tools you are familiar with. Briefly, the general architecture that provides the plug & play infrastructure supporting LLs is comprised of:

- software to detect when a USB device having a LL structure is inserted
- software to start another local IPFS node and configure it and the main IPFS node so they recognize each other as peers.
- a pirate menu entry so users can cleanly stop the LL IPFS server and allow the LL USB device to be safely removed without corrupting it
- **systemd** units to manage LL events such as those above

The complicated part is the second item in the list above. It is two python3 scripts, one started at boot time to monitor the *udev* system which launches the second to do the bulk of the heavy lifting.

The heavy lifting is such that all content in the LL is accessible from the main IPFS node transparently without exposing the LL content to the public IPFS swarm. To do that, changes to both IPFS server’s JSON config files must be made and the servers restarted.

The USB event detection code is a small python3 program (**uDevPython.py**) started at boot by the *systemd* unit **libLibeUSB.service**. When a LL USB device is detected, **uDevPython.py** runs **usbEvent.py** where the real magic & Atlas heavy lifting is performed.

The support software and infrastructure is installed and tested on all Pirate Stick and Pirate Box images. It is easier on the Pirate Stick due to how it runs a fully functional IPFS node on live USB media. When the IPFS Setup Wizard is run it will save the existing *.ipfs* folder by changing a symbolic link. The source for that link can be used for your LL and copied to the USB device. In the future I will retrofit this functionality back into the Pirate Box. By doing so it will also eliminate the need for the “Upgrade to External Storage” tool, although that tool also formats an external USB device and copies the IPFS repository found in *home/ipfs/.ipfs* to it. More about that in a future release.

## The USB Liberty Library Data Structure

- Content must be stored on a partition labeled “LIBERTY\_LIBRARY” in all caps
- The file system on the LL partition must be formatted as ext4
- A single, hidden folder named .ipfs contains the entirety of LL content
- A SQLite metadata file named llbry.db must be provided in the root of the LL partition

The .ipfs folder is identical to the .ipfs folder of a normal IPFS node repository. The llbry.db file is a SQLite database file containing searchable metadata used by the IPFS Search Tool found in the Advanced and Experimental Tools submenu of the main Pirate menu.

One way the SQLite database and IPFS content for the LL can be created is via the Media Grabber or Media Grabber Batch mode, the operation of which is described in a separate document in detail. It uses the popular youtube-dl downloader to scrape video and audio content from over 1000 sources such as vimeo, bitchute, lbry and youtube and add them to IPFS, and keeps track of the metadata associated with them (60+ fields of information such as title, upload date, author, media source etc) in the catalog.sqlite database which may be copied to the llbry.db file in the root of the LL USB device.

The SQLite database file serves as an index for the IPFS content. It contains metadata for each file in the LL IPFS repository stored on the LL USB stick. If you don't use the Media Grabber to create the llbry.db you will have to create it some other way, to track the LL content you add to the LL USB device. If you do that make sure it uses the same schema or your LL will not be searchable using the IPFS Search Tool.

The IPFS Search Tool was built to find content on my IPFS servers scraped with the Media Grabber tools. It can also be used to find content on USB LLs, but it relies on the SQLite database index file of metadata to do so. The database schema can be obtained from the catalog.sqlite SQLite file.

## Creating the .ipfs folder of content for LL

The easiest way to do this is to copy the IPFS repository (the .ipfs folder) to a USB device partition labeled “LIBERTY\_LIBRARY”. Whatever content you added to your IPFS node using the “ipfs add” command (on the command line) or imported and pinned in the IPFS Companion browser extension (referred to as the “IPFS Control Panel”) will become the content of your LL. NOTE: make sure you bookmark one or more items for test purposes later. Here in brief are the steps to create your LL after you have the content for it in your IPFS node:

1. Bookmark one or more items in your IPFS content that can be used to test the LL later.
2. Using gparted, format your LL USB device and create a partition with an ext4 filesystem.
3. Label the partition LIBERTY\_LIBRARY in all capitals.
4. Stop the main IPFS node with the **sudo systemctl stop ipfs** command.
5. Mount the newly created partition by removing the USB device and re-inserting it.
6. When “LIBERTY\_LIBRARY” appears on the desktop double click it to mount and open it.  
You can then close it after it is mounted. Note where it's mounted, usually /media/ipfs.
7. Copy the populated IPFS node repository to the LL partition you just mounted:

```
cp -r /home/ipfs/.ipfs /media.ipfs/LIBERTY_LIBRARY/.
```

8. Insure the ownership of the files on the LL device are correct:

**chown -R ipfs.ipfs /media/ipfs/LIBERTY\_LIBRARY**

9. Remove the LL USB device.

Your new LL is now ready, however do not insert it yet. Since it is an identical copy of your main IPFS repository, a conflict exists as they both have the same peerID and that needs to be resolved. To do that safely, rename your main IPFS node repository folder with:

**mv ~/.ipfs ~/.ipfs.save**

Now we can re-initialize the main IPFS node by going through the setup wizard, which will create a new peerID and a fresh, empty IPFS node. Run the Setup Wizard from the main Pirate Tools menu. When that is finished and your IPFS node has been restarted you can proceed to test your new LL in the next step.

10. This step requires the CID or hash for the items you bookmarked in step 1. Insert your LL USB device and dismiss the file manager popup. If your LL is working properly you should be able to open those bookmarks just as you did before, however now they will be coming from your new LL!

You can be certain of this if you take your IPFS node offline by disconnecting from Internet and opening a different bookmark that was not pinned on your IPFS node. We do this double check because the content you saved on your LL might also be served from another node on the Internet, not your LL. If you can still open the bookmarks you saved in step 1 and not those only available while online, congratulations, your LL is working!