iMazing CLI - Documentation

Last updated: **January 5th, 2025**

Author: **DigiDNA - Jérôme Bédat**

Table of contents:

1. Installation

1.1. macOS

1. Download and install iMazing: https://imazing.com/download
2. Add iMazing CLI to your path environment variable by editing either `~/.zshrc` or `~/.bashrc` depending on the shell you are using. Add the following line to your shell config:

```
export PATH=/Applications/iMazing.app/Contents/MacOS/:$PAT
H
```

3. To activate iMazing CLI, run:

```
iMazing --license <your iMazing CLI license code>
```

1.2. Windows

1. Download and install iMazing: https://imazing.com/download
2. Launch the Windows “Command Prompt” (`cmd`)
3. To add iMazing CLI to your path environment variable, run:

```
setx path "%path%;C:\Program Files\DigiDNA\iMazing"
```

4. To activate iMazing CLI, run:

```
iMazing-CLI --license <your iMazing CLI license code>
```

Run iMazing `--help` (or iMazing-CLI `--help` on Windows) to list commands available.

Run iMazing `--remove-license` to deactivate your license and free up a slot.

2. Requirements and other important info

2.1. Windows Apple Components & Drivers

Starting from version 2.17, iMazing Installer can download all necessary Apple components from the Microsoft Store to install them. iMazing Installer will place all components in the following folder:

`C:\ProgramData\DigiDNA\iMazing\MobileDevice`

To troubleshoot Apple components installation, a log file is created at this path:

`C:\ProgramData\DigiDNA\iMazing\MobileDevice\log.txt`

Custom Apple Components Installation

iMazing requires **Apple Mobile Device Support**. For Wi-Fi connectivity, **Bonjour** is also required. The two `.msi` installers are packaged in the `iTunesSetup.exe` or `iTunes64Setup.exe` Installer packages and can be extracted with 7zip or WinRAR. It is important to keep these components up to date in order to maintain compatibility with new iOS versions and devices, and avoid compatibility issues.

- Windows 64 bit: https://www.apple.com/itunes/download/win64
- Windows 32 bit: https://www.apple.com/itunes/download/win32

> 💡 *Note: Apple Components will be automatically downloaded and installed by the iMazing Installer on Windows and are available by default on macOS.*

Please refer to the following documentations for more details:

- https://imazing.com/documentation/Install-Uninstall-iMazing-in-Enterprise-Environments.pdf
- https://imazing.com/documentation/iMazing-in-Enterprise-and-Institutional-Environments.pdf

2.2. macOS Apple Components

iMazing requires `MobileDevice.framework`. It is installed by default with macOS. Since macOS High Sierra 10.13, the system automatically updates this framework when a new version is available.

2.3. Get iMazing CLI version

You can run the following command to check which version of iMazing CLI is currently installed:

```
iMazing --version
```

2.4. Check if iMazing CLI or Apple Components should be updated

```
$ iMazing --check-needs-update

iMazing is up to date - 3.0.0.0
Apple Mobile Device is up to date - 988.220.5
```

Check if iMazing CLI or Apple Components should be updated for compatibility with latest iOS versions and devices.

> 💡 *Note: this command doesn't update iMazing CLI nor Apple Components. To update Apple Components on Windows, follow instruction at section 2.5, or download the latest iTunes version. On macOS apply system updates.*

2.5. Update Apple Components on Windows

To update Apple components on Windows, you can run the following iMazing CLI command:

```
$ iMazing --update-mobile-device
```

2.6. Apple Services

iMazing CLI interacts with Apple servers in 3 cases:

- Checking for new iOS versions and downloading them
- Activating iOS devices
- Signing-in to the App Store and downloading apps

The only other reason for iMazing CLI to make outgoing connections is to contact our license management server `https://api.imazing.com`

3. Commands & Usages

iMazing CLI is a powerful tool to manage and interact with iOS devices from a Mac or PC computer. We'll cover here a few example use cases of the most common commands.

On macOS, the CLI is named iMazing. On Windows, it's `imazing-cli` or `iMazing-CLI`

3.1. See all available commands and options with help command

```
$ iMazing --help
```

Only commands and options available in your build will be listed.

3.2. Pair and activate devices

When connecting a new iOS device to a computer, it is necessary to first *pair* the device and the computer to allow communication.

List not paired devices

```
$ iMazing --device-list-not-paired \
--timeout 1

Looking for not paired devices...
00008110-000640CB0F05801E (iPhone)
Command succeed
```

First obtain the ID of the you wish to pair. Set the `--timeout` option to 1 for quick results. The default timeout for this command is 60 seconds.

Pair and activate the device

```
$ iMazing --device-pair \
--udid 00008110-000640CB0F05801E \
--activate

00008110-000640CB0F05801E - Looking for device...
00008110-000640CB0F05801E (iPhone) - Pair device
00008110-000640CB0F05801E (iPhone) - User Interaction: Enter pas
00008110-000640CB0F05801E (iPhone) - User Interaction: Trust cor
00008110-000640CB0F05801E (iPhone) - Pairing successful
```

While pairing a device you will be asked to unlock the device by entering its passcode, reply to the trust dialog box shown on the device and, since iOS 11, enter the device passcode again on the device to validate pairing.

- `--activate` option is optional but it is safe and convenient to pass it always: if the device is not activated, most operations are unavailable.

Note that you can use `--pair` and `--activate` options with other commands too, they will trigger pairing and activation if needed.

> 💡 *Note: you can use `--udid` any to pair all devices which you connect:*

```
$ iMazing --device-pair \
--udid any \
--activate
```

In that case, the CLI won't exit before you stop it, and will pair and activate any device you connect.

You can also associate these options to other commands. The following command will pair, activate and back up any device you connect to iMazing.

```
$ iMazing --backup-device \
--udid any \
--pair \
--activate
```

Important: if the user refuses the trust prompt, you will get a `kAMDUserDeniedPairingError` on every subsequent pairing attempt. This is normal behaviour - iOS refuses pairing until the user disconnects and reconnects the iOS device.

Unpair / Forget devices

```
$ iMazing --device-forget \
--udid 00008110-000640CB0F05801E
```

To unpair a device and remove it from iMazing's cache, execute the above command. This command does not delete backups of the device.

Important: on Windows, this command will ask for elevated privileges.

3.3. Communicate with device, get info and modify options

List devices currently paired and connected

```
$ iMazing --device-list \
--timeout 1 \
--usb
```

This command will return devices connected via USB.

- `--timeout` option is by default set to 60 seconds. It is necessary because iMazing can connect to iOS devices via Wi-Fi. Give iMazing enough time to wake up devices over WLAN. If you only intend to connect to USB connected devices, reduce the timeout to the minimum (1) and use the `--usb` option.

List devices in Recovery or DFU mode

```
$ iMazing --device-list-recovery-or-dfu \
--timeout 1
```

This command will return devices connected via USB which are in Recovery or DFU mode.

- `--timeout` option is by default set to 60 seconds. It is necessary because iMazing can connect to iOS devices via Wi-Fi. Give iMazing enough time to wake up devices over WLAN.

List all connected devices, paired, in Recovery or DFU mode, or not paired

```
$ iMazing --device-list-all-connected
```

Get a list of all connected devices, paired or not. Same options as for the command --device-list are available. Default timeout is 60, use `--timeout` 1 and `--usb` if all you need is a list of USB connected devices, both paired and not paired.

List devices in cache

```
$ iMazing --device-list-cache
```

Get a list of all devices paired with iMazing.

Execute a command on multiple devices

```
$ iMazing --device-info \
--udid 00008110-000640CB0F05801E \
00008120-000867CB0F07801F
```

Most commands can be executed on multiple devices simultaneously by separating device ids with whitespace characters.

Execute a command on any new connected device

```
$ iMazing --device-info \
--udid any
```

In this mode the CLI will not exit and wait for any connected device and execute the specified command for each of them. The command will be executed only once per device, in other words: if you disconnect / reconnect a device it won't be executed multiple times for the same device.

Change device name

```
$ iMazing --device-change-name \
--udid 00008110-000640CB0F05801E \
--device-name "My iPhone"
```

Enable Wi-Fi connection

```
$ iMazing --device-enable-wifi \
--udid 00008110-000640CB0F05801E
```

💡 *Note: unlike in iMazing, Wi-Fi connection is by default disabled in the CLI. This setting is persistent.*

Disable Wi-Fi connection

```
$ iMazing --device-disable-wifi \
--udid 00008110-000640CB0F05801E
```

Get device info

```
$ iMazing --device-info \
--udid 00008110-000640CB0F05801E
```

Get device disk usage

```
$ iMazing --device-disk-usage \
--udid 00008110-000640CB0F05801E
```

Get device battery diagnostic

```
$ iMazing --device-battery-diagnostic \
--udid 00008110-000640CB0F05801E
```

```
$ iMazing --device-battery-diagnostic \
--udid 00008110-000640CB0F05801E \
--xml \
--silent
```

(to export in .plist XML format - only available in custom builds)

```
$ iMazing --device-battery-diagnostic \
--udid 00008110-000640CB0F05801E \
--json \
--silent
```

(to export in JSON format - only available in custom iMazing CLI builds.)

Show device console

```
$ iMazing --device-console \
--udid 00008110-000640CB0F05801E
```

Observe device notifications

```
$ iMazing --device-observe-notifications \
--udid 00008110-000640CB0F05801E
```

Shutdown device

```
$ iMazing --device-shutdown \
--udid 00008110-000640CB0F05801E
```

Restart device

```
$ iMazing --device-restart \
--udid 00008110-000640CB0F05801E
```

3.4. Back up device

Backup device

```
$ iMazing --backup-device \
--udid 00008110-000640CB0F05801E \
--backup-location-path "/Users/Je/Desktop/Backups" \
--password 1234 \
--no-archiving
```

- `--backup-location-path` option allows you to change the device's backup location. By default the device backup location path will be:
- macOS: `/Users/<your username>/Library/Application Support/iMazing/Backups/<device id>`
- Windows: `C:\Users\<your username>\AppData\Roaming\iMazing\Backups\<device id>`

This setting is persistent, per device.

- `--password` the correct backup password is needed for iMazing to parse the backup in order to verify it, compute metrics, and archive it.

Note that if the password was saved in the macOS Keychain or Windows Credentials (--save-password option), iMazing CLI will use that when needed.

- `--save-password` option will save the device password in the macOS Keychain or Windows Credentials.
- `--enable-archiving` use this option to enable backup archiving for this device. This is a persistent, per device setting.
- `--disable-archiving` use this option to disable backup archiving for this device. This is a persistent, per device setting.
- `--no-archiving` use this option to disable archiving for this backup only. This is a non persistent option.

For more info about backup archiving, read the following article: https://imazing.com/guides/backup-options-in-imazing

- `--dont-verify-backup` will speedup the backup process by skipping the backup verification and metrics phase at the end of the backup. If this option is provided and backup archiving is disabled, you don't need to provide the password.

Check if device has backup encryption enabled

```
$ iMazing --device-check-backup-encryption \
--udid 00008110-000640CB0F05801E
```

Verify backup password

```
$ iMazing --backup-device-verify-password \
--udid 00008110-000640CB0F05801E \
--password 1234
```

Enable backup encryption

```
$ iMazing --backup-device-change-password \
--udid 00008110-000640CB0F05801E \
--password 1234 \
--new-password 123456 \
--save-password
```

It is necessary to provide the current password if backup encryption is currently enabled on the device.

- `--save-password` option will save the device password in the macOS Keychain or Windows Credentials.

Disable backup encryption

```
$ iMazing --backup-device-remove-password \
--udid 00008110-000640CB0F05801E \
--password 123456
```

It is necessary to provide the current password to disable encryption.

Change device backup location

```
$ iMazing --backup-device-change-location \
--udid 00008110-000640CB0F05801E \
--backup-location-path "/Users/Je/Desktop/Backups"
```

Reset device backup location

```
$ iMazing --backup-device-reset-location \
--udid 00008110-000640CB0F05801E
```

The device backup location will be reset to:

- macOS: `/Users/<your username>/Library/Application Support/iMazing/Backups/<device id>`

- Windows: `C:\Users\<your username>\AppData\Roaming\iMazing\Backups\<device id>`

3.5. Restore backup to device

Check if *Find my iDevice* is enabled

```
$ iMazing --device-check-find-my-device \
--udid 00008110-000640CB0F05801E
```

It is required to disable “Find my iPhone / iPad” before restoring a backup. The command --backup-restore will inform you if you need to disable it and wait until it is done, but it can be useful to check for this setting specifically.

Check if device has data

```
$ iMazing --device-check-has-data \
--udid 00008110-000640CB0F05801E
```

It is useful to know if the target device already contains any data before running the `--backup-restore` command. If it does, it is safer to pass the `--restore-erase` option when restoring, especially if the backup’s UDID and the target device’s UDID don’t match.

Check if an OS update is required on the target device before restoring a backup

```
$ iMazing --device-check-requires-os-update-to-restore-backup \
--udid 00008110-000640CB0F05801E \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
```

This command will check if the iOS version installed on the target device is sufficient to restore the backup. If it is not, you can use the option `--restore-update-os` to automatically update the OS on the target device before restoring the backup.

Restore backup

```
$ iMazing --backup-restore \
--udid 00008110-000640CB0F05801E \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
--password 1234 \
--device-name "New iPhone" \
--restore-update-os \
--restore-erase \
--dont-verify-password
```

- `--backup-path` is required.
- `--password` is also required if the backup is encrypted.
- `--device-name` is optional. If you use this option, iMazing will rename the target device after it reboots.
- `--restore-update-os` is optional. Use to automatically update iOS on the target device if needed.
- `--restore-erase` is optional. Use to delete all data on the target device before

restoring the backup. It is recommended to use this option in most cases, but not if the target device is brand new or already erased of course, in which case it would just slow down the process without providing any benefits.

3.6. iOS Installation, erase and more

Erase device

```
$ iMazing --device-erase \
--udid 00008110-000640CB0F05801E \
--confirm
```

Erasing a device requires “Find My device” to be turned off. The CLI will ask you to turn it off if it’s not already done. `--confirm` is here as an extra security, to reduce chances of accidentally erasing devices. Note that this command works via Wi-Fi too, and induces a near immediate, non-cancellable start of the erase process. Use with care!

- `--confirm` is required, to prevent executing this command by mistake.
- `--preserve-esim-data` is optional, preserves e-SIM data.
- `--prevent-proximity-setup` is optional, after the device restarts the Setup Assistant will not propose to transfer data from another device located closeby.

Exit Setup Assistant

```
$ iMazing --device-kill-setup-assistant \
--udid 5a6cbe61bd6e45e1d02907ce85421fe47fdd73
```

This command will reboot the device and try to bypass the iOS setup assistant if applicable.

> 💡 *Note: exiting the Setup Assistant is not possible on iOS 14 and above.*

Manage IPSW cache

By default, IPSW files downloaded by iMazing as part of a backup restore or iOS update operation are not persistent.

- `--cache-folder` option can be used to point iMazing to your own cache. If iMazing needs to update iOS, it will look for an appropriate IPSW file in the specified folder, and if it does not find one, it will use the folder to store the download persistently.

Update OS

```
$ iMazing --device-update-os \
--udid 00008110-000640CB0F05801E \
--ipsw "/Users/Je/Desktop/IPSWs/iPhone_5.5_Restore.ipsw" \
--backup
```

This command will backup the device and update its OS with the specified IPSW file. Only IPSW files signed by Apple can be installed. If the device is in Recovery

or DFU mode, this command won't back up devices in Recovery or DFU mode when using `--backup`.

- `--ipsw` is optional. If not specified, iMazing will download from Apple servers the latest available iOS version *for the specified device*.
- `--backup` is also optional.

> 💡 *Note: all backup options are also available for this command when used in conjunction with the `--backup` option (cf: command `--backup-device`).*

Enter in Recovery mode

```
$ iMazing --device-enter-in-recovery \
--udid 00008110-000640CB0F05801E
```

This command will put a device (currently in Normal mode) in Recovery mode.

Exit Recovery mode

```
$ iMazing --device-exit-recovery \
--udid GQJ5CQ4DWY
```

This command will exit Recovery mode when a device is in that mode. It only works when a device is in Recovery mode, not in DFU mode. To exit DFU mode, you must reinstall the OS.

Reinstall OS

```
$ iMazing --device-reinstall-os \
--udid GQJ5CQ4DWY \
--ipsw "/Users/Je/Desktop/IPSWs/iPhone_5.5_Restore.ipsw" \
--erase
```

This command will reinstall the device's OS. It works with devices in Normal mode but also with devices in Recovery or DFU mode.

- `--ipsw` is optional, if not specified, iMazing will download from Apple servers the latest available iOS version *for the specified device*.
- `--erase` will fully reinstall the device without attempting to keep user data. Omitting the erase option is possible, but doesn't always work: the operation may fail. This is one of the key differences between reinstall-os and update-os commands, the latter being designed to update iOS whilst preserving user data.

Export raw device data

```
$ iMazing --device-export-raw-data \
--udid 00008110-000640CB0F05801E \
--target-path "/Users/Je/Desktop/iPhone Raw Data"
```

This command will back up the device and export all data from it.

Export Unified Logs

```
$ iMazing --device-export-unified-logs \
--udid 00008110-000640CB0F05801E \
--target-path "/Users/Je/Desktop/iPhone Unified Logs"
```

This command will export approximately 15 days of system logs in Apple's *Unified Log* format (`.logarchive`). You can read this log format using the macOS *Console* app.

Export Sysdiagnose

```
$ iMazing --device-export-sysdiagnose \
--udid 00008110-000640CB0F05801E \
--target-path "/Users/Je/Desktop/iPhone Sysdiagnose"
```

This command will export all existing *Sysdiagnose* reports if any are available. If none are present, the user will be prompted to trigger a *Sysdiagnose*, wait for the report to be prepared, and then transfer it to the target path.

> 💡 *Note: Generating a report requires a user interaction. To start sysdiagnose, press and hold both volume buttons along with the side or top button on your device for 1 to 1.5 seconds. Once you release the buttons, sysdiagnose will start and your device will take a screenshot. Only on iPhone, not on iPad, you will feel a short vibration when sysdiagnose starts.*

Export Logs and Crash Reports

```
$ iMazing --device-export-logs-and-crash-reports \
--udid 00008110-000640CB0F05801E \
--target-path "/Users/Je/Desktop/iPhone Sysdiagnose"
```

This command will export all logs and crash reports available in the Logs partition.

Export Process list

```
$ iMazing --device-process-list \
--udid 00008110-000640CB0F05801E
```

This command will export the list of processes currently running on the device in CSV format.

Detect Malware and Spyware activity

```
$ iMazing --device-check-malware-activity \
--udid 00008110-000640CB0F05801E \
--target-path "/Users/Je/Desktop/iPhone Analysis"
```

This command will back up your device and analyze its backup to find indicators of malware and spyware activity. This feature is based on Amnesty International's work. More information here:

https://imazing.com/blog/detecting-pegasus-spyware-with-imazing

- `--target-path` is required, you must pass a folder path on your local file system where the CSV report will be written to.
- `--source-path` is optional, if not specified, iMazing will download a list of known malicious email addresses, domains, process names and file names from publicly available STIX files. You can provide with this option your own STIX files for research purposes. Use a local folder path containing at least 1 `.stix` or `.stix2` files.
- `--backup` is also optional, when passed, it will back up your device, if not passed, your device must have an existing backup.

> 💡 *Note: all backup options are also available for this command when used in conjunction with the `--backup` option (cf: command `--backup-device`).*

3.7. Get backup info and extract data

Get backup info

```
$ iMazing --backup-info \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
```

Get backup disk usage

```
$ iMazing --backup-disk-usage \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
```

Extract files and folders from backup (only available in custom builds)

```
$ iMazing --backup-extract \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
--source-path "HomeDomain/Library/SMS/sms.db" \
"MediaDomain/Library/SMS" \
```

```
--target-path "/Users/Je/Desktop/Extracted Messages" \
--password 1234
```

This command will export all Messages data from a backup. Messages files are stored in two separate domains. Here, we specify the database file from the `HomeDomain`, and the folder containing attachments from the `MediaDomain`, both to a folder on our desktop named Extracted Messages. You can use iMazing's backup browser to learn about iOS backups' folder structure: https://imazing.com/guides/how-to-extract-files-and-data-from-an-encrypted-iphone-backup

- `--source-path` you can specify multiple source paths separated by a space
- `--target-path` is the target folder. This folder must already exists
- `--password` you need to provide the backup password if the backup is encrypted and the password is not stored in macOS Keychain or Windows credentials.

Extract all files and folders from backup (only available in custom builds)

```
$ iMazing imazing --backup-extract-all \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
--target-path "/Users/Je/Desktop/Backup Content" \
--password 1234
```

Detect Malware and Spyware activity

```
$ iMazing --backup-check-malware-activity \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
--target-path "/Users/Je/Desktop/Backup Analysis"
```

This command will analyse a backup to find indicators of malware and spyware activity. This feature is based on Amnesty International's work. More information here:

https://imazing.com/blog/detecting-pegasus-spyware-with-imazing

- `--target-path` is required, you must pass a folder path on your local file system where the CSV report will be written to.
- `--source-path` is optional, if not specified, iMazing will download a list of known malicious email addresses, domains, process names and file names from publicly available STIX files. You can provide with this option your own STIX files for research purposes. Use a local folder path containing at least 1 `.stix` or `.stix2` files.

3.8. Apps

List installed apps

```
$ iMazing --device-list-apps \
--udid 00008110-000640CB0F05801E \
--app-type user \
--file-sharing

org.videolan.vlc-ios, "VLC", (v. 3.1.2), 143532032
org.whispersystems.signal, "Signal", (v. 2.29.3), 64667648
org.wikimedia.wikipedia, "Wikipedia", (v. 6.0.1), 81125376
```

This command will list only user installed apps which have file sharing enabled. The first field is the app bundle id, important for next commands. The last field is the app total disk usage in bytes (including its stored data)

- `--app-type` you can specify the app type: `user`, `system`, `all` (default is `user`)
- `--file-sharing` is optional and will return only file sharing enabled apps

Get app info

```
$ iMazing --device-app-info \
--udid 00008110-000640CB0F05801E \
```

```
--bundle-id org.videolan.vlc-ios
```

- `--bundle-id` you can specify one or multiple app bundle ids

Install app

```
$ iMazing --device-install-app \
--udid 00008110-000640CB0F05801E \
--source-path "/Users/Je/iOS Apps/org.videolan.vlc-ios.ipa" \
"/Users/Je/iOS Apps/com.DigiDNA.FileAid.ipa"
```

- `--source-path` you can specify one or multiple .ipa files to install. You can also restore `.imazingapp` files (app data backups)

Uninstall app

```
$ iMazing --device-uninstall-app \
--udid 00008110-000640CB0F05801E \
--bundle-id org.videolan.vlc-ios \
org.wikimedia.wikipedia
```

- `--bundle-id` you can specify one or multiple app bundle ids

Extract app data

```
$ iMazing --device-extract-app-data \
--udid 00008110-000640CB0F05801E \
--bundle-id org.videolan.vlc-ios
```

- `--bundle-id` you can specify one or multiple app bundle ids

Download apps from the App Store (only available in custom builds)

```
$ iMazing --download-app \
--apple-id myaccount@icloud.com \
```

```
--apple-id-password 1234 \
--save-password \
--bundle-id org.videolan.vlc-ios \
--target-path /Users/Je/Downloads/VLC.ipa
```

- `--apple-id` if you don't specify the Apple ID, the last one used will be used
- `--apple-id-password` you don't need to specify this option if you have previously chosen to save the Apple ID password in macOS Keychain or Windows Credentials.
- `--save-password` to save the Apple ID password in macOS Keychain or Windows Credentials.
- `--bundle-id` you must specify the app bundle ID you want to download
- `--target-path` if you don't specify this option, the app will be save in iMazing's App Library (see iMazing *Preferences > Apps* to learn what the exact location is)

Transfer files to apps (equivalent to the iMazing "Quick Transfer" action)

List compatible apps with specific file types

The command below allows you to identify which apps installed on the device are compatible with specific file types.

One file:

```
$ iMazing --device-apps-for-files \
--udid 00008110-000640CB0F05801E \
--files "/Users/Je/Documents/Doc 1.pdf"
```

Multiple files:

```
$ iMazing --device-apps-for-files \
--udid 00008110-000640CB0F05801E \
```

```
--files "/Users/Je/Documents/Doc 1.pdf" \
"/Users/Je/Documents/Doc 2.docx"
```

All files contained in a folder:

```
$ iMazing --device-apps-for-files \
--udid 00008110-000640CB0F05801E \
--files "/Users/Je/Documents"
```

All files contained in multiple folders:

```
$ iMazing --device-apps-for-files \
--udid 00008110-000640CB0F05801E \
--files "/Users/Je/Documents" "/Users/Je/Downloads"
```

Transfer files to a compatible app

The command below enables you to transfer multiple files or a folder to a compatible app, which can be found using the `--device-apps-for-files` command.

```
$ iMazing --transfer-files-to-apps \
--udid 00008110-000640CB0F05801E \
--bundle-id org.videolan.vlc-ios \
--files "/Users/Je/Downloads/Move1.mov" \
"/Users/Je/Downloads/Move2.mov" \
"/Users/Je/Downloads/All Movies”
```

Note: iMazing CLI supports transferring files to all third-party apps that have file sharing enabled.

You can also transfer files to the following Apple apps:

- Photos - bundle id: `com.apple.mobileslideshow`
- Music - bundle id: `com.apple.Music`
- Books - `com.apple.iBooks`
- Podcasts - `com.apple.podcasts`

- Ringtones - bundle id: `com.apple.Ringtones`
- TV - bundle id: `com.apple.tv`
- Contacts - bundle id: `com.apple.MobileAddressBook`
- Apps (`.ipa`) - bundle id: `com.apple.AppStore`
- Configuration Profiles (`.mobileconfig`), Provisioning Profiles (`.mobileprovision`) or iMazing Configurator blueprints (`.blueprint`) - bundle id: `com.apple.Profiles`

> 💡 **Important:** For a more direct method to browse or transfer files to and from third-party apps, refer to the next section, *3.9*.

3.9. File System

List files

This command will list files and folders at the root of the Media partition:

```
$ iMazing --device-fs-list \
--udid 00008110-000640CB0F05801E \
--item-path "/"

"Podcasts", Directory, 0
"Downloads", Directory, 0
"CloudAssets", Directory, 0
"Books", Directory, 0
"Photos", Directory, 0
"Deferred", Directory, 0
"EnhancedAudioSharedKeys", Directory, 0
"Recordings", Directory, 0
"PhotoStreamsData", Directory, 0
"Radio", Directory, 0
"Espresso", Directory, 0
"Memories", Directory, 0
"DCIM", Directory, 0
```

```
"iTunesRestore", Directory, 0
"iMazing", Directory, 0
"iTunes_Control", Directory, 0
"MediaAnalysis", Directory, 0
"PhotoData", Directory, 0
"PublicStaging", Directory, 0
"Purchases", Directory, 0
"LoFiCloudAssets", Directory, 0
"AirFair", Directory, 0
```

This command will list files and folders of the `Documents` folder of a specific file sharing enabled app:

```
$ iMazing --device-fs-list \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--item-path "/Documents"

"Doc.pdf", File, 22067
"Screenshot 1.png", File, 1371063
"Downloads", Folder, 0
```

This command will list files and folders at the root of the crash report and logs partition:

```
$ iMazing --device-fs-list \
--udid 00008110-000640CB0F05801E \
--fs-crash-reports \
--item-path "/"

"proactive_notification-2023-04-10-104420.000.ips", File, 919
"intelligenceplatformd.cpu_resource-2023-04-10-131116.ips", File
"JetsamEvent-2023-04-10-113726.ips", File, 266385
"PerfPowerServicesSignpostReader.cpu_resource-2023-04-10-131041
"SiriSearchFeedback-2023-04-10-130040.ips", File, 347
"JetsamEvent-2023-04-10-111623.ips", File, 262462
```

```
"proactive_notification-2023-04-10-104418.0002.ips", File, 763
"JetsamEvent-2023-04-10-102311.ips", File, 252945
"SiriSearchFeedback-2023-04-10-130040.000.ips", File, 345
"xp_amp_app_usage_dnu-2023-04-10-120232.ips", File, 143005
"proactive_notification-2023-04-10-104420.ips", File, 810
"proactive_notification-2023-04-10-104418.000.ips", File, 763
"Retired", Directory, 0
"SiriSearchFeedback-2023-04-10-114128.ips", File, 347
"Panics", Directory, 0
"photoanalysisd.cpu_resource-2023-04-10-102725.ips", File, 7284
"proactive_notification-2023-04-10-104418.ips", File, 763
"JetsamEvent-2023-04-10-111510.ips", File, 261436
"proactive_notification-2023-04-10-111932.ips", File, 814
"JetsamEvent-2023-04-10-114244.ips", File, 273990
"proactive_notification-2023-04-10-111931.ips", File, 810
"JetsamEvent-2023-04-10-113715.ips", File, 266452
"Baseband", Directory, 0
"awdd-2023-04-10-113840-1279.consolidated.metriclog", File, 750
"JetsamEvent-2023-04-10-130527.ips", File, 280438
"JetsamEvent-2023-04-10-102807.ips", File, 252984
"Assistant", Directory, 0
```

File system command options:

- `--bundle-id` with this option, you can specify an app bundle id to connect to its container, otherwise by default you will access the Media folder.
- `--fs-root` if you use this option you will see the root of an app container (only available for apps in development mode). If `--bundle-id` is not set, you will access the root of the file system of a jailbroken device.
- `--fs-crash-reports` allows you to access a special partition containing all crash reports and log files. Don't pass options `--bundle-id` and `--fs-root` when using this option.

Note: If no file system command option is passed (either `--bundle-id` , `--fs-root` or `--fs-crash-reports`), it defaults to the Media partition.

Get file or folder info

```
$ iMazing --device-fs-info \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--item-path "/Documents/Doc.pdf"
```

Remove file or folder

```
$ iMazing --device-fs-remove \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--item-path "/Documents/Doc.pdf"
```

Rename / move file or folder

```
$ iMazing --device-fs-rename \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--source-path "/Documents/Doc.pdf" \
--target-path "/Documents/Downloads/Doc 1.pdf"
```

Create folder

```
$ iMazing --device-fs-create-dir \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--item-name "My Subfolder" \
--target-path "/Documents/Downloads"
```

Transfer file to device

```
$ iMazing --device-fs-copy-to-device \
--udid 00008110-000640CB0F05801E \
```

```
--bundle-id com.DigiDNA.FileAid \
--source-path "/Users/Je/Desktop/Files" \
"/Users/Je/Desktop/Doc.pdf" \
--target-path "/Documents/Downloads"
```

- `--source-path` you can specify one or multiple paths to copy
- `--target-path` must be an existing folder in a writable location on the target device

Transfer file to computer

```
$ iMazing --device-fs-copy-to-computer \
--udid 00008110-000640CB0F05801E \
--bundle-id com.DigiDNA.FileAid \
--source-path "/Documents/Downloads/Doc.pdf" \
--target-path "/Users/Je/Desktop"
```

- `--source-path` you can specify one or multiple paths to copy
- `--target-path` must be an existing folder in a writable location on the target computer

3.10. Export Datasets

Export a single or multiple datasets from a connected device

```
$ iMazing --device-export-datasets \
--udid 00008110-000640CB0F05801E \
--backup \
--bundle-id com.apple.mobilephone \
--target-path "/Users/Je/Desktop" \
--item-name CallHistory.csv \
--format csv
```

Export a single or multiple datasets from a backup

```
$ iMazing --backup-export-datasets \
--backup-path "/Users/Je/Desktop/Backups/00008110-000640CB0F058(
--bundle-id com.apple.mobilephone \
--target-path "/Users/Je/Desktop" \
--item-name CallHistory.csv \
--format csv
```

These two commands will export a specific dataset in a given format.

- `--backup` back up device (some datasets need to be extracted from a backup, see below)
- `--bundle-id` with this option, you can specify the dataset bundle id:
  - Photos: `com.apple.mobileslideshow` (needs backup)
  - Contacts: `com.apple.MobileAddressBook` (needs backup)
  - Calendars: `com.apple.mobilecal` (needs backup)
  - Messages: `com.apple.MobileSMS` (needs backup)
  - WhatsApp: `net.whatsapp.WhatsApp` (needs backup)
  - Call History: `com.apple.mobilephone` (needs backup)
  - Voicemail: `com.apple.Voicemail` (needs backup)
  - Notes: `com.apple.mobilenotes` (needs backup)
  - Safari History, Reading List and Bookmarks: `com.apple.mobilesafari` (needs backup)
  - Profiles: `com.apple.Profiles` (needs backup)
  - Music: `com.apple.Music`
  - Movies, TV Shows, Videos: `com.apple.tv`
  - Books: `com.apple.iBooks`
  - Ringtones: `com.apple.Ringtones`
  - Voice Memos: `com.apple.VoiceMemos` (needs backup)
  - All third-party apps' data: `com.apple.AppStore`

- `--target-path` must be an existing folder in a writable location on the target computer
- `--target-filename` specify the target filename, it only works when exporting Call History or Safari History (the other datasets are exported to multiple files)
- `--create-subfolders-hierarchy` create sub folders hierarchy when exporting from multiple devices, multiple datasets or for datasets such as Messages, Photos etc. with multiple items (chats, moments...)
- `--format` the export format (only available for Contacts, Calendars, Messages, WhatsApp and Call History)
  - All format: `all`
  - Text: `text`
  - CSV: `csv`
  - Excel: `excel`
  - RSMF: `rsmf` (for messages)

> 💡 *Note: all backup options are also available for the command `--device-export-datasets` when used in conjunction with the `--backup` option (cf: command `--backup-device`).*

3.11. Configuration and Provisioning

List iMazing Configurator blueprints

```
$ iMazing --library-list-blueprints
```

This command will list all blueprints stored in the local iMazing Configurator library. iMazing Configurator is a powerful tool to provision, configure, repair and manage multiple Apple mobile devices in bulk (learn more: https://imazing.com/configurator).

Apply blueprints

```
$ iMazing --device-apply-blueprint \
--udid 00008110-000640CB0F05801E \
--blueprint-id 3A2EB642-EB04-406B-932C-089DA39753E2
```

```
$ iMazing --device-apply-blueprint \
--udid 00008110-000640CB0F05801E \
--blueprint-path "/Users/Je/MyBlueprint.blueprint" \
--blueprint-password 1234
```

This command applies a blueprint to a device. You can apply a blueprint to multiple devices using either multiple UDIDs:

```
$ iMazing --device-apply-blueprint \
--udid 00008110-000640CB0F05801E 00008220-000904CB0F07801F \
--blueprint-id 3A2EB642-EB04-406B-932C-089DA39753E2
```

or any to apply it to any new connected device:

```
$ iMazing --device-apply-blueprint \
--udid any \
--blueprint-id 3A2EB642-EB04-406B-932C-089DA39753E2 \
--usb
```

- `--blueprint-id` this is the blueprint unique identifier. You can create and edit blueprints within iMazing standard version.
- `--blueprint-path` this is the blueprint file path when applying a blueprint exported from the library
- `--blueprint-password` this is the blueprint file path when applying a blueprint exported from the library
- `--device-number` this option allows you to specify a device number which will be used to identify the device for various purposes. Contact our support team for more details on this option

- `--log-path` this option allows you to specify a folder path on your computer's local filesystem where logs should be written
- `--device-number` an optional number that you can assign to the device to identify it

List device installed profiles

```
$ iMazing --device-list-profiles \
--udid 00008110-000640CB0F05801E
```

Install configuration or provisioning profile to device

```
$ iMazing --device-install-profile \
--udid 00008110-000640CB0F05801E \
--profile-path "/Users/Je/MyConfig.mobileconfig"
```

Remove profile from device

```
$ iMazing --device-remove-profile \
--udid 00008110-000640CB0F05801E \
--profile-id Jes-MacBook-Pro.287A429F-D8FD-4B0C-9055-974E52A1AC5
```

4. JSON Output (only available in custom builds)

JSON output for all CLI commands is available upon request, for large scale projects and by contract only. Use the `--json` option to output JSON formatted objects. `--silent` option is automatically enabled when this mode is used.

4.1. JSON Options

Output JSON to stderr instead of stdout

- `--json-to-stderr`

This option can be useful to facilitate parsing when extra outputs are printed by Apple components in the stdout. Extra outputs are especially printed when using the command `--backup-device` on Windows.

Ignore strings files

- `--ignore-strings-files`

This option is useful if you want to ignore language `.strings` files and receive text keys instead values to facilitate messages parsing.

4.2. Message Type `Result`

Result messages are used for simple commands, not for lengthy operations. Result type depends on command.

Boolean

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Succeeded",
        "Result": true
    }
}
```

Dictionary

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Succeeded",
        "Result":
        {
            "2afed850c28ec5838277dca45aa0de735d4349f5": "Device
```

```
            "5ca9cf818b5b53f8390ab428c5a5d6219c283bea": "Device
        }
    }
}
```

String

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Succeeded",
        "Result": "iOS 12"
    }
}
```

Number

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Succeeded",
        "Result": 125
    }
}
```

Array

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Succeeded",
```

```
        "Result":
        {
            "Value 1",
            "Value 2"
        }
    }
}
```

Possible values for Status field

- `Succeeded`
- `Failed`

In case `Failed` is returned, you will get the an error code and the associated text.

```
{
    "MessageType": "Result",
    "Message":
    {
        "Status": "Failed",
        "ErrorCode": 100001,
        "Text":
        {
            "Label": "Cannot connect to Jay’s iPhone."
        }
    }
}
```

4.3. Message Type `Status`

`Status` messages are used only during lengthy operations: backup, restore, erase, update...

```
{
    "MessageType": "Status",
    "Message":
```

```
    {
        "Phase": "BackupDevice",
        "UID": "2afed850c28ec5838277dca45aa0de735d4349f5",
        "Name": "Jay's iPhone",
        "Status”: "Failed",
        "ErrorCode": 3,
        "Text":
        {
            "Label": "Connection was lost before the backup c
ould complete.\n\nIf your device disconnects repeatedly, try
to restart it as well as your computer."
        }
    }
}
```

Possible values for Status field

- `Started`
- `Succeeded`
- `Failed`
- `Cancelled`

The Started message is always sent when the operation begins. `Succeeded`, `Failed` and `Cancelled` are sent at the end.

The `Phase` field is the sub-operation context. Jump to the Phases

4.4. Message Type `Progress`

```
{
    "MessageType": "Progress",
    "Message":
    {
        "Phase": "InstallIPSW",
        "UID": "2afed850c28ec5838277dca45aa0de735d4349f5",
```

```
        "Name": "Jay's iPhone",
        "Progress":
        {
            "IsIndeterminate" : False,
            "Value" : 0.5
        },
        "Text":
        {
            "Label": "Installing iOS - Unmounting filesystems, 5
        }
    }
}
```

The Phase field is the sub-operation context. Jump to Phases

4.5. Message Type `Interaction`

Interaction messages are sent when user interaction is required.

```
{
    "MessageType": "Interaction",
    "Message":
    {
        "Phase": "BackupDevice",
        "UID": "2afed850c28ec5838277dca45aa0de735d4349f5",
        "Name": "Jay's iPhone",
        "Interaction": "UnlockDevice",
        "Text”:
        {
            "Label": "Jay's iPhone is locked.\nPlease enter y
our passcode on the device, or unlock it with Touch ID or Fac
e ID."
        }
    }
}
```

Possible values for Interaction field

- `ConnectDevice`
- `UnlockDevice`
- `TrustComputer`
- `InsertSIMCard`
- `UnlockSIMCard`
- `TurnOffFindMyDevice`

The Phase field is the sub-operation context. Jump to Phases

4.6. Phases

`Status`, `Progress` and Interaction messages all have a Phase string value.

This represents the current sub-operation context. The Phase value is most useful when reporting progress.

- `PairDevice`
- `ActivateDevice`
- `ConnectToDevice`
- `CheckFindMyDevice`
- `BackupDevice`
- `RestoreBackup`
- `DownloadIPSW`
- `InstallIPSW`
- `EraseDevice`
- `ApplyConfiguration`

5. Common Error Codes

Error codes not listed here come straight from iOS services or Apple services.

Use `Phase` for context, and Text/Label to display the message if needed.

- `DeviceCannotConnect` = *100001*
  The device is either not connected via USB or not reachable over the Wi-Fi network.
- `DeviceCannotStartSession` = *100002*
  Starting a session fails if the device is not paired or if the pairing records are corrupted or have been invalidated device side (pairing revoked). In that case the user must unpair (forget) the device and repair it. Use command: `--device-forget` and `--device-pair`
- `DeviceDisconnected` = *100003*
  The device has been disconnected during an operation.
- `DeviceLocked` = *100004*
  The device is locked and needs to be unlocked. The user must enter his passcode to allow this operation.
- `BackupInsufficientFreeDiskSpaceOnTheDeviceToBackup` = *100005*
  The device cannot be backed up because there is not enough space available on it. A minimum amount of space is required for iOS to prepare the backup.
- `BackupUnableToBackupUntilSetupIsFinished` = *100006*
  The device cannot be backed up when the iOS *Setup Assistant* is displayed. The user must finish the Setup Assistant step to allow backing up the device.
- `BackupDriveNotAvailable` = *100007*
  The device backup location is not available. The user needs to make sure the backup location is available and if the location is on an external drive or NAS, make sure it is properly connected and available on macOS or Windows. The user can also change the device backup location in the device options.
- `BackupDriveInsufficientSpace` = *100008*
  The device cannot be backed up when the iOS *Setup Assistant* is displayed on the device. The user must finish the Setup Assistant steps to allow backing up the device.
- `BackupComputerWentToSleep` = *100009*
  The computer entered in sleep mode while an operation was running.

- `BackupTimedout` = *100010*
  The backup service running on the device didn't respond to iMazing. The user should restart both computer and device and try again.
- `BackupRestoreInsufficientOSVersion` = *100011*
  iMazing cannot restore a backup to the device because it was made with a more recent version of iOS. Updating iOS on the device is necessary to restore this backup.
- `BackupRestoreInsufficientFreeSpaceOnDeviceToRestore` = *100012*
  iMazing cannot restore a backup because there is not enough space available on the target device. The user should try to erase the target device before restoring this backup.
- `BackupCorrupted` = *100013*
  The backup is corrupted and cannot be restored. The user should try to use the
  Backup Repair function available in iMazing, command: `--backup-repair`
- `BackupRestoreFindMyDeviceMustBeTurnedOff` = *100014*
  To restore a backup, the user must turn off
  Find My iPhone / iPad on the target device.
- `BackupCancelled` = *100015*
  The backup has been cancelled by the user.
- `RestoreCancelled` = *100016*
  The restore has been cancelled by the user.
- `BackupPasswordIsInvalid` = *100017*
  The backup encryption password is invalid. The user must enter the right password to allow iMazing to read the backup. Option
  `--password`
- `DeviceRequiresBackupEncryption` = *100018*
  The device has a restriction configuration profile installed which enforces backup encryption to be enabled when backing up. When a device has this restriction, it is also not possible to restore a non-encrypted backup. In that case the user must either uninstall the restriction configuration profile or erase the device before restoring an encrypted backup.

- `BackupInsufficientOSVersion` = *100019*
  iMazing can only backup devices on iOS 4 and above. Older iOS versions are not supported.
- `BackupGenericError` = *100020*
  iMazing received a generic error from the backup service running on the device. This message description usually provides more information about the issue. More info here: https://support.imazing.com/hc/en-us/articles/115002137493-Backup-General-iOS-device-backup-issues
- `BackupUnableToOpenAnAppDomainDirectory` = *100021*
  An app installed on the device seems to be corrupted and therefore iMazing cannot back up the device. The user should try to open the device console while backing up the device and search for *BackupAgent* to see all errors related to the backup, to try to find which app is causing the issue.
- `BackupErrorRemovingSnapshotDirectory` = *100022*
  iMazing cannot remove the backup "Snapshot" directory in the device backup location.
- `ActivationViaAppleServersFailed` = *100023*
  iMazing cannot activate the device because it cannot establish a connection with Apple servers. The user must activate the device manually in the Setup Assistant.
- `ActivationFailedSIMCardLockedByCarrier` = *100024*
  iMazing cannot activate the device because the device SIM card has been locked by the carrier.
- `PairingFailed` = *100025*
  iMazing cannot pair the device and computer together. The user should try to restart both the computer and device and try again.
- `ValidatePairingFailed` = *100026*
  iMazing cannot validate the device and computer pairing. The user should try to restart both the computer and device and try again.

- `PairingProhibitedWithComputer` = *100027*
  iMazing cannot pair the device because it is supervised by a supervision identity which prevents pairing without the supervision host certificate. The user must import the appropriate supervision Organization in iMazing to allow pairing this device:

  https://imazing.com/guides/configurator-overview#orgs
- `BackupCannotGetAppsInfoFromDevice` = *100028*
  iMazing cannot retrieve installed 3rd party apps' info from the device and therefore cannot back it up. The user should try to restart the device, relaunch iMazing and try again.
- `BackupDBNotReadable` = *100029*
  iMazing cannot read the backup database because it is corrupted.
- `BackupDBNotWritable` = *100030*
  iMazing cannot write the backup database because it is corrupted.
- `BackupItemNotFound` = *100031*
  iMazing cannot find a file or folder in the backup.
- `BackupCannotCopyItem` = *100032*
  iMazing cannot export a file from the backup.
- `BackupLoadingAborted` = *100033*
  The user tried to open a backup to copy it in the default iMazing backup location but an existing backup was already in the backup location.
- `BackupInfoPlistNotReadable` = *100034*
  The `Info.plist` file stored in the device backup location is either missing or corrupted.
- `BackupUnlockKeybagFailed` = *100035*
  iMazing cannot read an encrypted backup because the provided password is not valid.
- `BackupUnlockKeybagCancelled` = *100036*
  The user didn't enter the right backup encryption password and decided to cancel the operation.

- `FSCannotReadLocal` = *100037*
  iMazing cannot read a file on the computer or external drive file system.
- `FSCannotWriteLocal` = *100038*
  iMazing cannot write a file or folder on the computer file system or external drive.
- `FSItemNotFound` = *100039*
  iMazing cannot find a file or folder on the local file system, external drive or on the device.
- `FSPermissionDenied` = *100040*
  iMazing cannot access a file or folder on the local file system, external drive or on the device. It doesn't have the right permissions to access it.
- `FSAFCConnectionNotAvailable` = *100041*
  iMazing cannot connect to the iOS service in charge of file transfers. The user should try to restart both device and computer and try again.
- `FSCannotCreateOperation` = *100042*
  iMazing cannot create a file system operation. This should never happen.
- `FSCannotProcessOperation` = *100043*
  iMazing cannot process a file system operation. The device may have been disconnected before the operation can run. The user should try to restart both device and computer.
- `FSNoManager` = *100044*
  iMazing cannot find its internal file system manager. This should never happen.
- `FSNoAppBundleID` = *100045*
  iMazing cannot find its internal file system manager. This should never happen.
- `FSAppFileSharingDisabled` = *100046*
  iMazing cannot access a 3rd party app container on the device because the file sharing has not been enabled by the developer.
- `FSCannotCreateAFCConnection` = *100047*
  iMazing cannot create a connection to the iOS service in charge of file

transfers. The user should try to restart both device and computer and try again.

- `DeviceCannotStartService` = *100048*
  iMazing cannot start a particular service on the device. The user should try to restart both device and computer and try again.
- `DeviceCannotSetSecureContext` = *100049*
  iMazing cannot set the secure context while connecting to a particular service running on the device. The user should try to restart both device and computer and try again.
- `ServiceConnectionUnavailable` = *100050*
  The connection with a service running on the device was lost. The user should try to restart both device and computer and try again.
- `ServiceTooManyRunning` = *100051*
  There are too many simultaneous connections to a particular service running on the device. The user should try to restart the device and try again.
- `ServiceCannotSendMessage` = *100052*
  iMazing cannot send a message to a particular service running on the device. The user should try to restart the device and try again.
- `ServiceCannotReceiveMessage` = *100053*
  iMazing cannot receive a message from a particular service running on the device. The user should try to restart the device and try again.
- `ServiceCannotGetSocket` = *100054*
  iMazing cannot obtain the connection socket to communicate with a particular service running on the device. The user should try to restart the device and try again.
- `ServiceDeviceLinkInvalidVersion` = *100055*
  iMazing cannot communicate with the backup or sync service running on the device because the installed iOS version is too old.
- `ServiceDeviceLinkInvalidResponse` = *100056*
  iMazing received an invalid response from the backup or sync service running on the device. The user should try to restart the device and try again.

- `HTTPRequestFailed` = *100057*
  iMazing cannot send HTTP requests to a particular server. The user should make sure the computer is connected to the Internet and that no proxy or firewall is blocking the connection.
- `AuthenticationFailed` = *100058*
  iMazing cannot connect to the App Store to download an app because the user didn't provide valid
  Apple ID credentials.
- `InitFailed` = *100059*
  iMazing cannot initialize the connection to the
  App Store. The user should make sure the computer is connected to the Internet and that no proxy or firewall is blocking the connection.
- `Authentication2FACodeRequired` = *100060*
  An Apple ID two factor authentication code must be entered by the user.
- `Cancelled` = *100061*
  An operation has been cancelled by the user.
- `DownloadFailed` = *100062*
  iMazing cannot download a file. The user should make sure the computer is connected to the Internet and that no proxy or firewall is blocking the connection.
- `AppNotFound` = *100063*
  iMazing cannot download an app because it is not available on the App Store.
- `AppNotPurchased` = *100064*
  iMazing cannot download an app because it has not been purchased from the App Store by the authenticated Apple ID.
- `ProfileInstallationFailed` = *100065*
  iMazing cannot install a configuration profile to the device. The reason will be available in the error message description.
- `AppleDriversAreNotProperlyInstalled` = *100066*
  Apple Components or Drivers are not properly installed. On macOS the user needs to apply all system updates or upgrade to a newer version of macOS. On Windows the user can reinstall iMazing, install the latest iTunes version or

go to iMazing Preferences and click
Reinstall Mobile Device Services.

- `ItemNotFound` = *100067*
  Generic error when an item cannot be found.

  `VPPClientContextIsDifferent` = *100068*
  The App Volume Purchase account used to download apps has its ownership tied to another application or MDM service.

- `UserDeniedPairing` = *100069*
  The user tapped
  Cancel on the device pairing *Trust* dialog shown on the device while establishing a pairing with the computer.

- `ServiceDidNotSendOrReceiveAllBytes` = *100070*
  iMazing cannot send or receive a certain number of bytes to or from a particular service running on the device. The user should try to restart the device and try again.

- `LibraryDriveNotAvailable` = *100071*
  iMazing's Library location is not available. The user needs to make sure the library location is available and if the location is on an external drive or NAS, make sure it is properly connected and available on macOS or Windows. The user can also change the library location in iMazing's Preferences.

- `AppleDriversInsufficientVersion` = *100072*
  The Apple components or drivers are outdated and need to be updated. On macOS the user needs to apply all system updates or upgrade to a newer version of macOS. On Windows the user can reinstall iMazing, install the latest iTunes version or go to iMazing Preferences and click
  Reinstall Mobile Device Services.

- `ActivationDeviceIsLockedByFindMyDevice` = *100073*
  iMazing cannot activate the device because
  Find My iPhone / iPad was enabled before the device was erased. The user must enter his *Apple ID* credentials to let iMazing activate the device or activate the device manually.

- `NotEnoughFreeSpaceAvailableOnTarget` = *100074*
  There is not enough free space available on the target device to transfer a file or folder.
- `CertificateNotFound` = *100075*
  iMazing cannot find a supervision certificate in the user's macOS *Keychain* or the Windows *Certificate Store*. The user should try to re-import the organization in iMazing:

  https://imazing.com/guides/configurator-overview#orgs
- `CertificateNotTrusted` = *100076*
  iMazing did find a supervision certificate in the user's macOS *Keychain* or the Windows *Certificate Store* but it is not trusted. The user should try to re-import the organization in iMazing or trust the certificate manually in macOS *Keychain* or Windows *Certificate Store*:

  https://imazing.com/guides/configurator-overview#orgs
- `CertificatePrivateKeyNotFound` = *100077*
  iMazing did find a supervision certificate in the user's macOS *Keychain* or the *Windows Certificate* store but its private key is not accessible. The user should try to re-import the organization in iMazing:

  https://imazing.com/guides/configurator-overview#orgs
- `LicensingInternetConnectionUnavailable` = *100078*
  iMazing cannot connect to internet to validate the license.
- `LicensingServerUnavailable` = *100079*
  iMazing licensing server (*api.imazing.com*) is unavailable.
- `DeviceEnterPasscodeToBackup` = *100080*
  The user must enter his device passcode while the device passcode prompt screen is displayed to allow backing up the device. This passcode prompt was introduced by Apple in iOS >= 16.1 and iOS >= 15.7.1.
- `iCloudAdvancedDataProtectionEnabled` = *100081*

iCloud Advanced Data Protection is enabled and therefore iMazing cannot download images or videos from *iCloud Photo Library*. *iCloud Advanced Data Protection* must be disabled,

- `ServiceDoesNotReturnData` = *100082*
  A service running on the iOS device does not return data.
- `Unknown` = *INT_MAX*
  An unknown error occurred. The error description usually provides more information.