ESP-IDF Versions & Releases

The full history of releases can be found on the GitHub repository Releases page. There you can find release notes, links to each version of the documentation, and instructions for obtaining each version.

Documentation

The documentation for the current stable release version can always be found at this URL: https://docs.espressif.com/projects/esp-idf/en/stable/ Documentation for the latest version (master branch) can always be found at this URL: https://docs.espressif.com/projects/esp-idf/en/latest/

Versioning Scheme

ESP-IDF uses Semantic Versioning. This means that:

- Major Releases, like v3.0
- Minor Releases like v3.1
- Bugfix Releases like v3.0.1

If updating to a new bugfix release (for example, from v3.0 to v3.0.1), you do not need to change any code in your project, and you only need to re-test the functionality directly related to bugs listed in the release notes on the Releases page.

Checking the Current Version

The local ESP-IDF version can be checked by using git:

```
cd $IDF_PATH
git describe --tags --dirty
```

The ESP-IDF version is also compiled into the firmware and can be accessed (as a string) via the macro IDF_VER. The default ESP-IDF bootloader will print the version on boot (the version information is not always updated in code, it only changes if that particular source file is recompiled). Examples of ESP-IDF versions:

Version String	Meaning
v3.2-dev-306-gbeb3611ca	Master branch pre-release. v3.2-dev - in development for version 3.2. 306 - number of commits after v3.2 development started. beb3611ca - commit identifier.
v3.0.2	Stable release, tagged v3.0.2.
v3.1-beta1-75-g346d6b0ea	Beta version in development (on a release branch). v3.1-beta1 - pre-release tag. 75 - number of commits after the pre-release beta tag was assigned.346d6b0ea - commit identifier.

Version String	Meaning
v3.0.1-dirty	Stable release, tagged v3.0.1. dirty means that there are modifications in the local ESP-IDF directory.

Updating ESP-IDF

These instructions assume that you already have a local copy of ESP-IDF cloned. E.g. by folowing the Getting Started guide. Updating ESP-IDF depends on which version(s) you wish to follow:

- Updating to Stable Release is recommended for production use.
- Updating to Master Branch is recommended for the latest features, development use, and testing.
- Updating to a Release Branch is a compromise between the first two.

After updating ESP-IDF, **execute install.sh again**, in case the new ESP-IDF version requires different versions of tools.

Updating to Stable Release

To update to a new ESP-IDF release (recommended for production use), this is the process to follow. When major or minor updates are released, check the Release Notes on the releases page and decide if you want to update or to stay with your current release. When a bugfix release for the version you are using is released (for example, if using v3.0.1 and v3.0.2 is released):

```
cd $IDF_PATH
git fetch
git checkout vX.Y.Z
git submodule update --init --recursive
```

Updating to a Pre-Release Version

It is also possible to git checkout a tag corresponding to a pre-release version or release candidate, the process is the same as Updating to Stable Release. (Pre-release tags are not always found on the Releases page.)

Updating to Master Branch

Using Master branch means living “on the bleeding edge” with the latest ESP-IDF code.This is the process to follow. Check out the master branch locally:

```
cd $IDF_PATH
git checkout master
git pull
git submodule update --init --recursive
```

Periodically, re-run git pull to pull the latest version of master. Note that you may need to change your project or report bugs after updating your master branch. It is strongly recommended to regularly run git pull and then git submodule update –init –recursive so a local copy of master does not get too old. Arbitrary old master branch revisions are effectively unsupportable “snapshots” that may have undocumented bugs. For a semi-stable version, try Updating to a Release Branch instead.

To switch from master to a release branch or stable version, run git checkout as shown in the other sections.

Updating to a Release Branch

In terms of stability, using a release branch is part-way between using the master branch and only using stable releases. A release branch is always beta quality or better, and receives bug fixes before they appear in each stable release. For example, to follow the branch for ESP-IDF v3.1, including any bugfixes for future releases like v3.1.1, etc:

```
cd $IDF_PATH
git fetch
git checkout release/v3.1
git pull
git submodule update --init --recursive
```

Each time you git pull this branch, ESP-IDF will be updated with fixes for this release.