

Git & Workflow

Introduction of the Git Workflow in Espressif

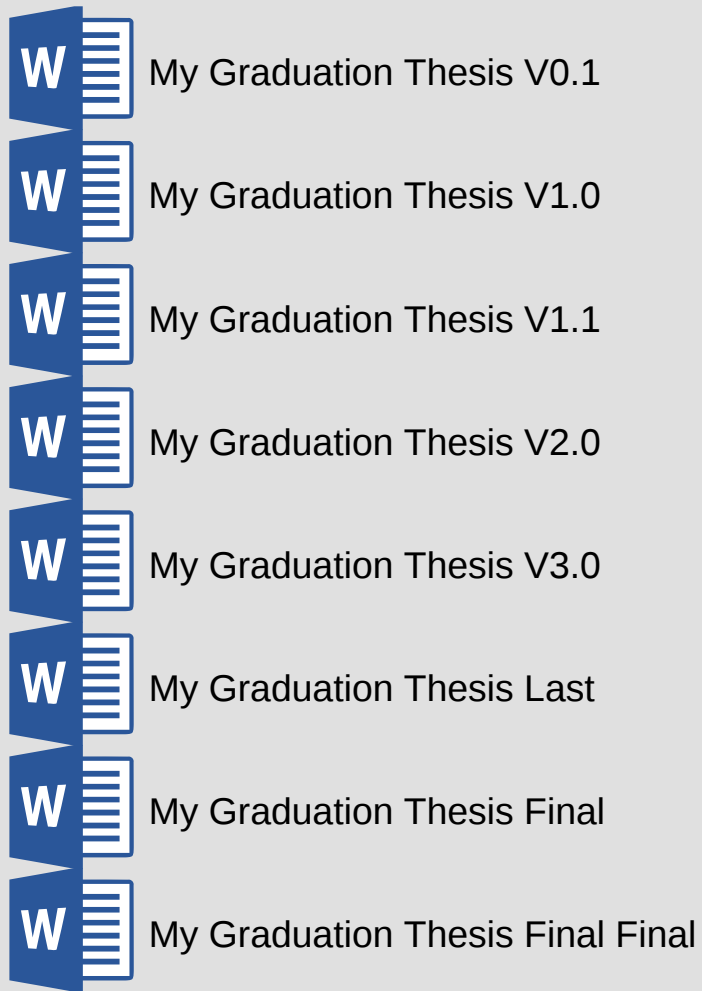
💖 IDF CORE TEAM 💖

Overview

- What is VCS, Git, Smart Git, GitHub, GitLab ...
- How to get started with Git
- How to submit Merge Request

Introduction of Git





➡ A Simple Version Control System (VCS) 🙄

- Efficiency ?
- Speed ?
- Conflict ?

“

Git is a free and
open source
distributed
version control
system with *speed*
and *efficiency*

”

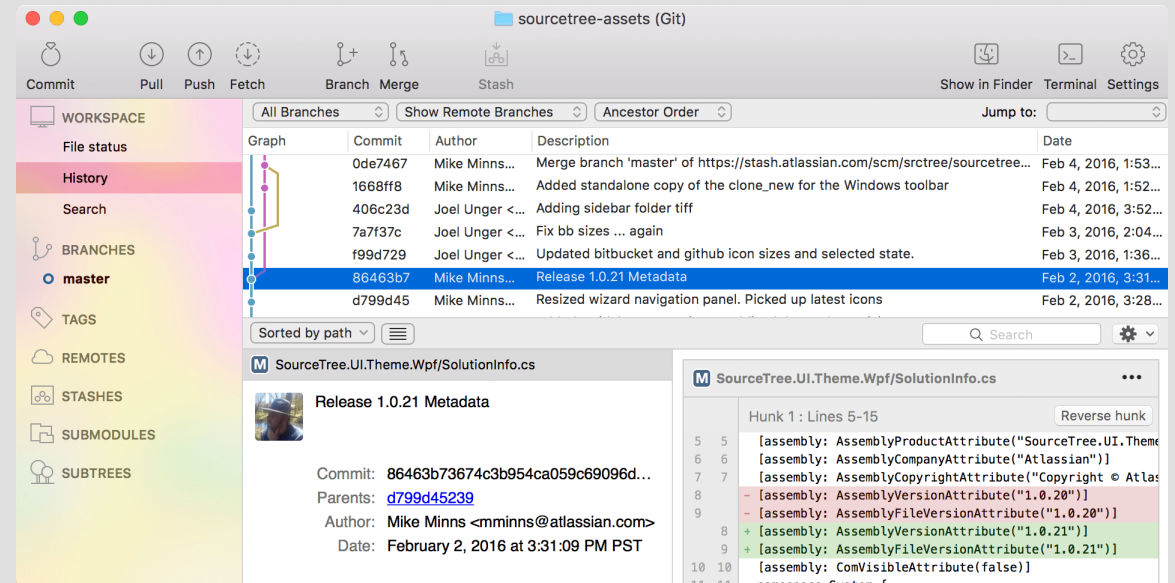
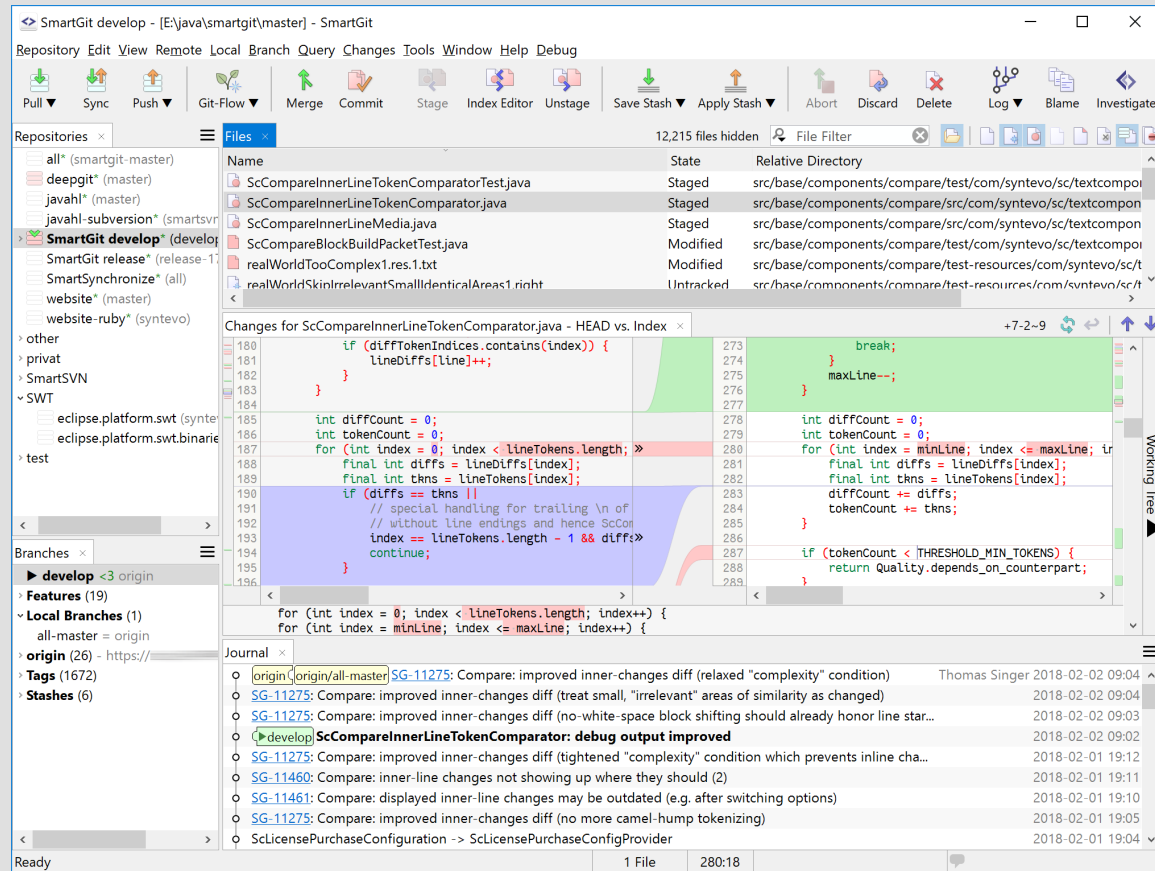


GitHub GitLab

- Web-based Git repository
- Issue tracking
- GitLab: built-in **CI/CD**
- GitHub: community **PR**



Git GUI



Get Started with Git



Command	Description
init	initialize an empty repository
config	setup name, email, etc
add	add changes to staging area
rm	remove file from repository
rm --cached	remove file only from staging area
commit	commit the changes in staging area
clone	clone a remote repository to local
status	check status of current branch

Command	Description
stash / stash pop	push / pop changes to / from stack area
mv	rename or move file
log	investigate the logs of commits
reflog	check history operation
push --tags	push the tag to remote repository
commit --amend	update the latest commit
commit --date	specify the commit date
checkout <file>	undo the modification of some file

Command	Description
reset <file>	un-stage some file
reset --soft	undo the last commit (keep the changes in staging area)
remote -v	check the url of remote repository
pull	pull changes from remote repository (auto-merge)
remote add	add a remote repository
diff <file>	find out changed lines in some file
tag	give the current commit a new tag
blame <file>	find out who introduced a bug 🤔

Command	Description
branch	list all branches
branch <nb> <hash>	create a new branch based on some commit
branch -d <name>	delete a branch
branch -m <name>	rename a branch
checkout <nb>	switch to another branch
checkout -b <nb>	create a new branch and then switch to
checkout -	switch to last branch
checkout <tag>	switch to a specific tag

Command	Description
push <remote> <branch>	push a branch to remote repository
merge <branch>	merge another branch to current one
fetch	get changes from remote without merge
bisect	find out which commit introduced bug
cherry-pick <hash>	make use of the commits from others
rebase	rebase current branch onto another one
add <file> --edit	specify the lines to stage
revert	undo a commit which has been pushed to remote

tldr

man

```
$ tldr git-log
```

- Show the sequence of commits starting from the current one, **in** reverse chronological order:

```
git log
```

- Show the **history** of a particular file or directory, including differences:

```
git log -p path/to/file_or_directory
```

- Show only the first line of each commit message:

```
git log --oneline
```

- Show an overview of **which** file(s) changed **in** each commit:

```
git log --stat
```

- Show a graph of commits **in** the current branch:

```
git log --graph
```

- Show a graph of all commits, tags and branches **in** the entire repo:

```
git log --oneline --decorate --all --graph
```

- Show only commits whose messages include a given string (**case-insensitively**):

```
git log -i --grep search_string
```

Set alias for common used commands in ~/.gitconfig file

```
[alias]

gl = log -n 30 --date-order --format="%Cgreen%h %Cred[%ci] %Creset <%an>%C(yellow)%d%Creset %Creset %Cgreen%s %Creset \"
hist = log --pretty=format:\"%C(yellow)%h %C(red)%d %C(reset)%s %C(green)[%an] %C(blue)%ad\" --topo-order --graph
st = status -uno -s
st2 = status -s
co = checkout
bl = blame --date=short
ci = commit
dt = difftool
dif = diff --word-diff
di = diff --no-ext-diff
```

***ignore* = do not put the matched files to staging area**

.gitignore in esp-idf 📎

```
.config # ignore .config file
*.o # ignore all object file
*.pyc # ignore all python byte code file
examples/**/sdkconfig # ignore sdkconfig files inside examples
examples/**/build # ignore build directory inside examples
```

```
# ignore all .a files but lib.a
*.a
!lib.a
```


submodule = 3rd party libraries/tools that reside in *another repository*

.gitmodules in esp-idf 📌

```
[submodule "components/esptool_py/esptool"]
  path = components/esptool_py/esptool
  url = ../../espressif/esptool.git
[submodule "components/esp_wifi/lib_esp32"]
  path = components/esp_wifi/lib_esp32
  url = ../../espressif/esp32-wifi-lib.git
```

```
# after a fresh git clone, submodule needs update accordingly
git submodule update --init --recursive
# add a new submodule
git submodule add <submodule_url> submodule_local_path
```

```
maoshengrong@FA000617:~/myrepo$ git gl
299c9c2 [2019-07-19 11:37:41 +0800] <suda-morris> (HEAD -> master) fileB: second commit
6b1c9b2 [2019-07-19 11:37:00 +0800] <suda-morris> fileC: third commit
299f6bd [2019-07-19 11:36:19 +0800] <suda-morris> fileA: first commit
598daa7 [2019-07-19 11:35:37 +0800] <suda-morris> Initial setup
maoshengrong@FA000617:~/myrepo$
```

What if the commits are
in a wrong order ?

```
git rebase -i <hash>
```



possible conflict

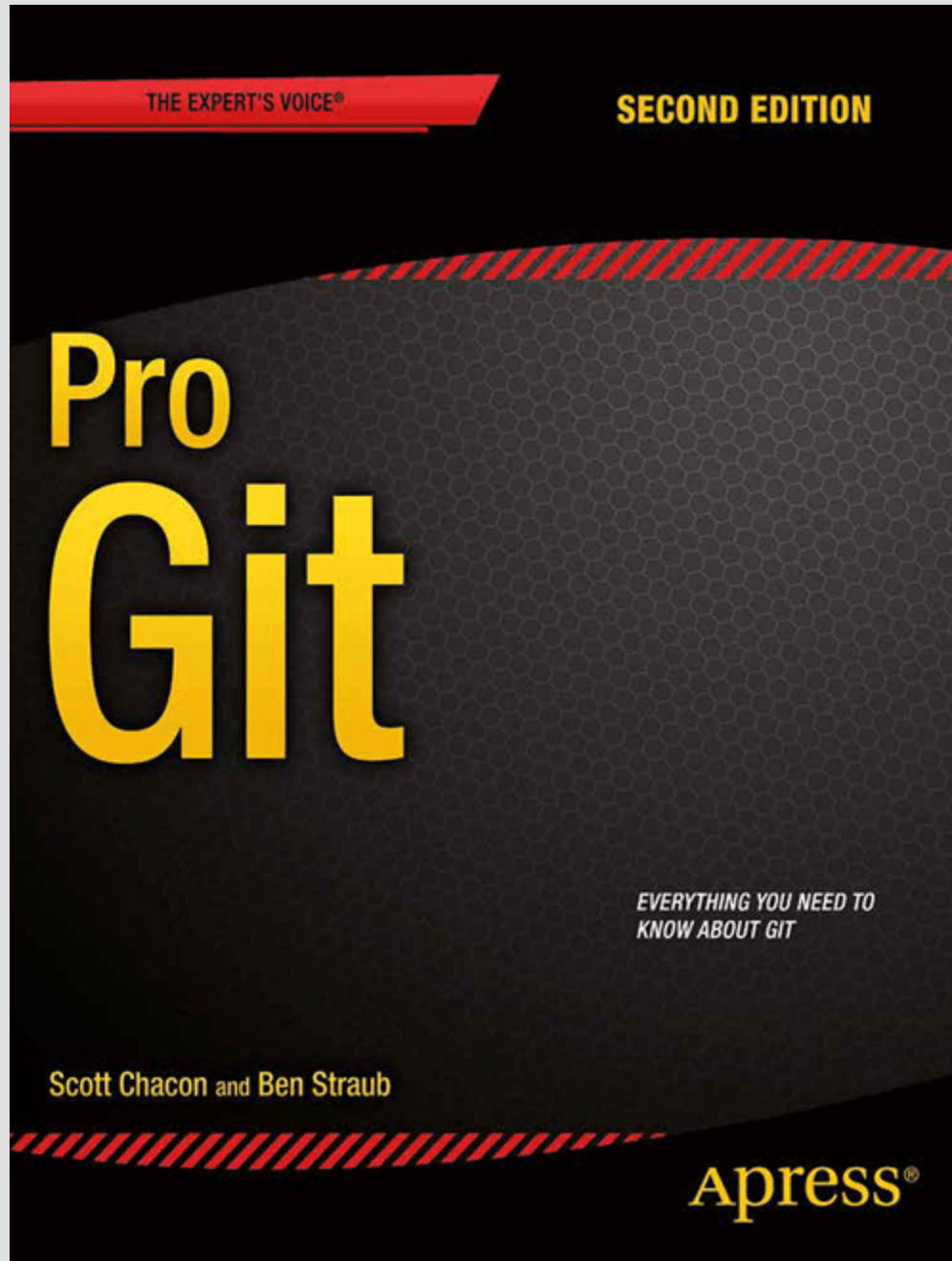
```
maoshengrong@FA000617:~/myrepo$ git gl
37907eb [2019-07-19 11:39:32 +0800] <suda-morris> (HEAD -> master) fileC: third commit
bbf4ad5 [2019-07-19 11:39:32 +0800] <suda-morris> fileB: second commit
299f6bd [2019-07-19 11:36:19 +0800] <suda-morris> fileA: first commit
598daa7 [2019-07-19 11:35:37 +0800] <suda-morris> Initial setup
maoshengrong@FA000617:~/myrepo$
```

What if
too many similar commits ?

```
git rebase -i <hash>
```

100
squash before merge

 Still far from sufficient



ProGit Book






Learn by Gaming -- GitHub

GitHub Walk Through Guide

Learn in interactive way

Learn Git Branching

Submit Merge Request in GitLab 🙌

- **master branch** 
 - Name: `master`
 - Where *integration* happens
 - Always produce working code which compiles and passes CI tests
- **feature and bugfix branches** 
 - Name: `feature/xxx` and `bugfix/xxx`
 - Where *development* happens
 - May contain broken or incomplete or testing code
- **release branches** 
 - Name: `release/v4.0`
 - Where releases are maintained and bugfixes are backported

What a good commit message looks like ?

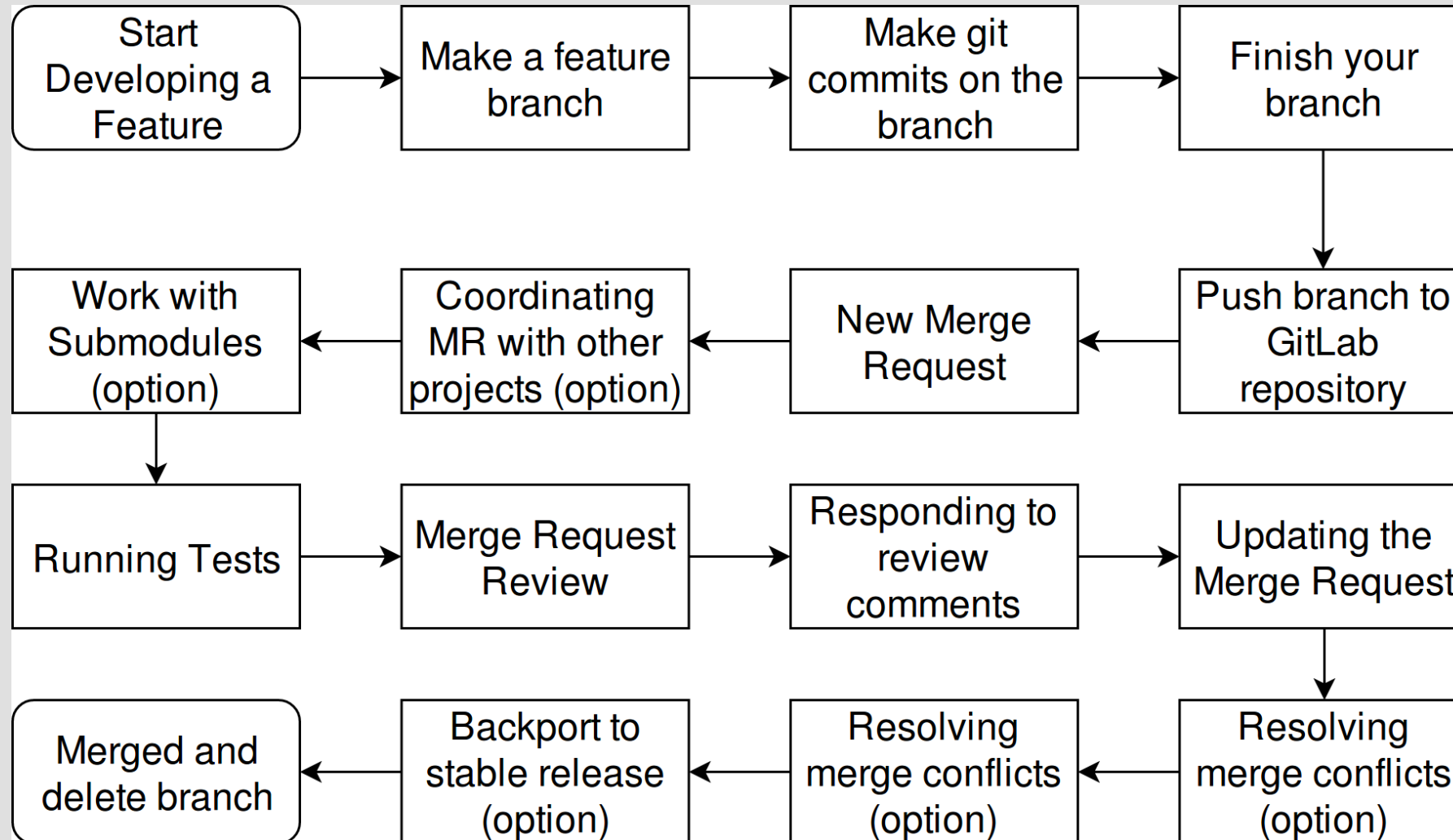
```
vfs/fatfs: fix stat call failing when called for mount point
```

```
FATFS does not support f_stat call for drive root. When handling stat  
for drive root, don't call f_stat and just return struct st with S_IFDIR  
flag set.
```

```
Closes https://github.com/espressif/esp-idf/issues/984
```

- **First Line Rule:** `component-name` `Add/Fix/Remove/Change` `THING`
 - esp32: Fix startup timeout issue 🙌
- **Detailed story after the first line**
 - Full link to Github Issues or PRs: `Fixes https://github.com/espressif/esp-idf/issues/73`
 - Jira ticket number: `Closes IDF-123`

Development workflow



What a good MR Description looks like ?

tools: add Dockerfile

This Dockerfile allows building Docker images of ESP-IDF.

Such images can be used by downstream projects in CI or during regular development, saving effort replicating the ESP-IDF installation steps.

These images should be tagged by IDF versions.

Example usage:

1. Building a project with GNU Make

```
$ docker run --rm -v $PWD:/build -w /build espressif/idf make
```

2. Building a project with CMake

```
$ docker run --rm -v $PWD:/build -w /build espressif/idf idf.py build
```

3. Using interactively

```
$ docker run --rm -v $PWD:/build -w /build -it espressif/idf
# inside the container
$ idf.py menuconfig
```

4. Specifying version

```
$ docker run --rm -v $PWD:/build -w build espressif/idf:v3.3 idf.py build
```

TODO

- ✓ add CI job to build the image (need some trick in the Dockerfile to use current IDF source and not the one from Github)
- ✓ set up Docker Hub CI to build images for us, or push from our CI runner to the Docker Hub
- ✓ add a documentation page (in which section?) to mention this image and its usage

Closes [IDFGH-1304](#)

Closes [IDF-590](#)

Edited 1天前 by Ivan Grokhotkov



Community Etiquette

- Be **polite**, even if the code is not good
- Try best to **review** PR/MR within 1 week of them being opened
- If the MR looks good to you, don't forget to give a **thumbup** 👍



- [ESP-IDF Internal Wiki](#)
- [Sourcegraph](#)
- [Read The Docs](#)