Github

```
git remote add upstream https://github.com/apache/incubator-mxnet
```

```
git checkout master
git fetch upstream master
git merge upstream/master
git push origin master

git checkout <branchname>
git merge master
git push origin <branchname>
```

```
git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

```
git fetch upstream master
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (9/9), done.
From https://github.com/apache/incubator-mxnet
* branch master -> FETCH_HEAD
afbb72fa1..7b787d3cf master -> upstream/master
```

```
git merge upstream/master
Updating ffeaf315a..7b787d3cf
Fast-forward
```

```
git push origin master
Counting objects: 139, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (56/56), done.
Writing objects: 100% (139/139), 32.48 KiB | 8.12 MiB/s, done.
Total 139 (delta 127), reused 92 (delta 83)
remote: Resolving deltas: 100% (127/127), completed with 95 local objects.
To https://github.com/ChaiBapchya/incubator-mxnet.git
ffeaf315a..7b787d3cf master -> master
```

With this, local, ChaiBapchya/incubator-mxnet and apache/incubator-mxnet are all in sync and on the same commit

Now, for the feature branch I'm working on:

```
 git checkout <branchname>
Switched to branch '<branchname>'
Your branch is up-to-date with 'origin/<branchname>'.
```

```
git merge master
Auto-merging tests/python/unittest/test_random.py
CONFLICT (content): Merge conflict in tests/python/unittest/test_random.py
Automatic merge failed; fix conflicts and then commit the result.
```

looked at the file, edited it

```
git add <file>
git commit
```

```
 git push origin <branchname>
Counting objects: 9, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 802 bytes | 802.00 KiB/s, done.
Total 9 (delta 8), reused 0 (delta 0)
remote: Resolving deltas: 100% (8/8), completed with 8 local objects.
To https://github.com/ChaiBapchya/incubator-mxnet.git
77c3d4923..1e1a79152 <branchname> -> <branchname>
```

Done.

Ammending last commit

```
git add changelog.md
```

```
git commit --amend
```

Edit the message if you want, else

```
git commit --amend --no-edit
```

To push the commit, if you haven't pushed last commit to remote, use normal git push command. However, if last commit is pushed to remote, use force (-f) since you've rewritten your commit history

```
git push -f origin some_branch
```

After PR is merged, safe to delete local branch

```
git branch -d feature_branch
```

Remove from Untracked files

```
git clean -fd
```

Git Config

```
[alias]
    lg = log --color --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %
    ci = commit
    st = status
    co = checkout
    br = branch
    cp = cherry-pick

[color]
    diff = auto
    status = auto
    branch = auto
    interactive = auto
    ui = true
    pager = true

[color "status"]
    added = green
    changed = red bold
    untracked = magenta bold

[color "branch"]
    remote = yellow
```

To work on a specific commit in specific tree of remote,
after git clone repo

```
git fetch https://github.com/apache/incubator-mxnet.git 956e52cf82c6a1963b125619330f6c
```

instead of `git fetch` followed by `git checkout <commit sha>`

For using specific branch,

```
git clone --recursive https://github.com/ChaiBapchya/incubator-mxnet.git
cd incubator-mxnet/
git checkout --track origin/<branchname>
```

At times while tracking new remote branch to local, gives following error

```
fatal: 'origin/flaky_test_randint_generator_fix' is not a commit and a branch 'flaky_t
```

Solution

```
git fetch origin
git checkout --track origin/flaky_test_randint_generator_fix
```

Situation
git commit done
now before pushing, you want to merge upstream and then push that merged + new commit
cant do it

it will go in as a new commit (merging master into branch) or else better is first merge mastrer into branch and then commit your changes.

```
git submodule update --recursive
git submodule update --init --recursive
```

Empty commit

```
git commit --allow-empty -m "Trigger notification"
```

When git status shows new commits in submodule

```
git status
On branch <branchname>
Your branch is up-to-date with 'origin/<branchname>'.

Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git checkout -- <file>..." to discard changes in working directory)

modified: 3rdparty/mkldnn (new commits)

no changes added to commit (use "git add" and/or "git commit -a")
```

Verify if submodule is updated

```
8c8590ad8b03:incubator-mxnet bapac$ git diff
diff --git a/3rdparty/mkldnn b/3rdparty/mkldnn
index 0e7ca7388..9910b4802 160000
--- a/3rdparty/mkldnn
+++ b/3rdparty/mkldnn
@@ -1 +1 @@
-Subproject commit 0e7ca738866d22cc700aa33b8de120b938f910d0
+Subproject commit 9910b480296a0d1496db466531e56729b3922bbf
```

Only update the submodule

```
8c8590ad8b03:incubator-mxnet bapac$ git submodule update --recursive
Submodule path '3rdparty/mkldnn': checked out '0e7ca738866d22cc700aa33b8de120b938f910c
```

Verify if all clean

```
8c8590ad8b03:incubator-mxnet bapac$ git status
On branch <branchname>
Your branch is up-to-date with 'origin/<branchname>'.

nothing to commit, working tree clean
```

Cherrypick a commit from master to release branch
Update the master

```
git fetch upstream
git co master
git pull upstream master –rebase
```

Checkout release branch and copy the commit you want to cherry-pick

```
git co v1.4.x
git cp b5ea1943b
git lg
git push origin v1.4.x
```

Copy contents from 1 local branch to new local branch

```
git checkout -b new_branch old_branch
```

Combination of

```
git checkout old_branch
git branch new_branch
```

Create new branch local, make changes and push to remote

```
git checkout -b new_branch old_branch
git push --set-upstream origin new_branch
```

Generally, old_branch should be master (which has been synced with fork)

Undo last commit

```
git reset --soft HEAD^
```