BTR Git Cheat Sheet

Basics

Help: git <command> --help
master: default release branch
_Test: default development branch

HEAD: current commit
HEAD^ or HEAD~1: parent of HEAD
HEAD^^ or HEAD~2: grandparent of HEAD
'HEAD@{yesterday}': HEAD at EOD yesterday
'HEAD@{2 weeks ago}': Head at EOD 2 weeks ago

When Using File Specs: full path, case specific, and can use Tab autocompletion for aid.
goGitRepos: Opens folder containing upstream repositories.
Open current console folder: ex .

Create

Create local and remote repository

```
git create {Parent} {Client}
```

Create, add all files and commit

```
git create {Parent} {Client} "Msg"
```

From existing BTR repository

```
git clone-btr {Parent} {Client}
```

From existing repository

```
git clone ~/existing ~/new
git clone git://host.org/project.git
git clone ssh://user@host.org/project.git
```

{Parent}-BTR, Buck, 4.5, 6.0, Shared{}

Browse

Description of last N commits

```
git l -N # Last 5 if omitted
```

Changes in working directory

```
git st
```

Files changed between commits

```
git ch  #HEAD / HEAD^
git ch id #id / id^
git ch id1 id2 #di1 / id2
```

Files changed between branches

```
git ch-b  <br1> {br2} # Current if omitted
```

Differences working directory and index

```
git dt <file>
git dt <file1> <id> # file1 diffs b/w work and id
```

Differences between commits

```
git dt-c id1 <id2> #id1^ if omitted
git dt-c id1 id2 <file>
```

Differences between branches

```
git dt-b br1 file1 # file1 diffs b/w br1 and curr
```

Differences index and HEAD

```
git dt-i <file> #if add, no commit
```

Differences work and HEAD

```
git dt-h <file> #if add, no commit, edit
```

Files currently tracked

```
git ls-files
```

Commits file was a part of

```
git fhs file1 # all commits
git fhs file1 5 # last 5 commits
```

Files deleted from repository

```
git deletes
```

Branch

List all branches

```
git br # -r for remote branches
```

Create and switch to Feature1 branch

```
git co -b Feature1
```

Switch to _Test branch

```
git co _Test
```

Pull new Feature1 branch from remote

```
git co -b Feature1 origin/Feature1
```

Merge _Test branch into master

```
git co master #switch to master
git merge _Test #merge _Test
```

Files changed between branches

```
git ch-br br1 br2
```

Delete a remote branch

```
git del-rb <remote> <branch>
```

Commit

Commit all local changes with short message

```
git com -am "Short Description, fixes #taskid"
```

Always include task ID in commit

```
git com -am "Spec Update, fixes #12345"
```

Add files to next commit

```
git add . # . Adds all, can put specific names
```

Add (forgotten) files to previous commit

```
git com –amend -a #Able to edit message
```

Commit all local changes, prompt for detail

```
git com -a #text editor prompts for detail
```

Pull latest changes from origin

```
git pull #does REBASE based on settings
```

Fetch latest changes from origin

```
git fetch #does NOT merge changes
```

Revert

Undo changes to a file

```
git co <file>
```

Pull previous version of file

```
git previous <commit> <file>
```

Restore deleted file

```
git undelete <file>
```

Discard ALL local changes

```
git reset –hard #CANNOT BE UNDONE
```

Undo changes from a commit

```
git revert id #creates new 'cancel' commit
```

Resolve Merge Conflicts

Resolve conflicting file after pull/rebase

```
e path/to/file #opens file in editor
# find >>>> and save appropriate code
git add .
git rebase –continue
```

Undo merge after pull/rebase conflicts

```
git rebase --abort
```

Publish

Push current branch to origin/branch

```
git push #local/remote names must match
```

Push new Topic1 branch to origin/Topic1

```
git push origin Topic1
```

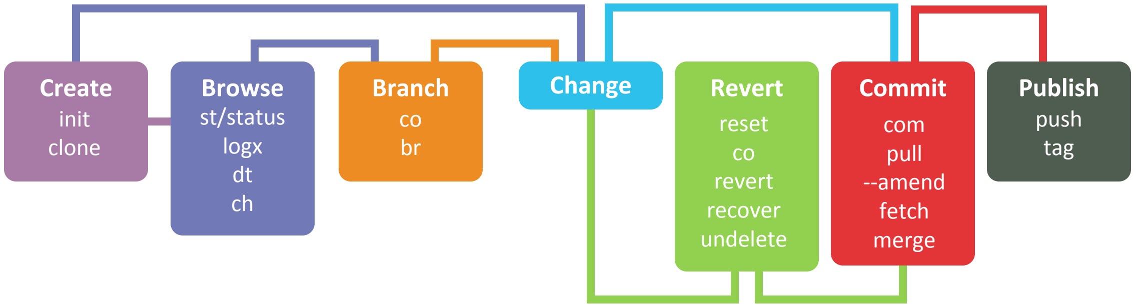
Make a version or milestone

```
git tag v1.0
```

Delete Topic1 branch from origin

```
git push origin :Topic1 #still need to del local
```

Workflow – Most Common Path Is Across Top



Create
init
clone

Browse
st/status
logx
dt
ch

Branch
co
br

Change

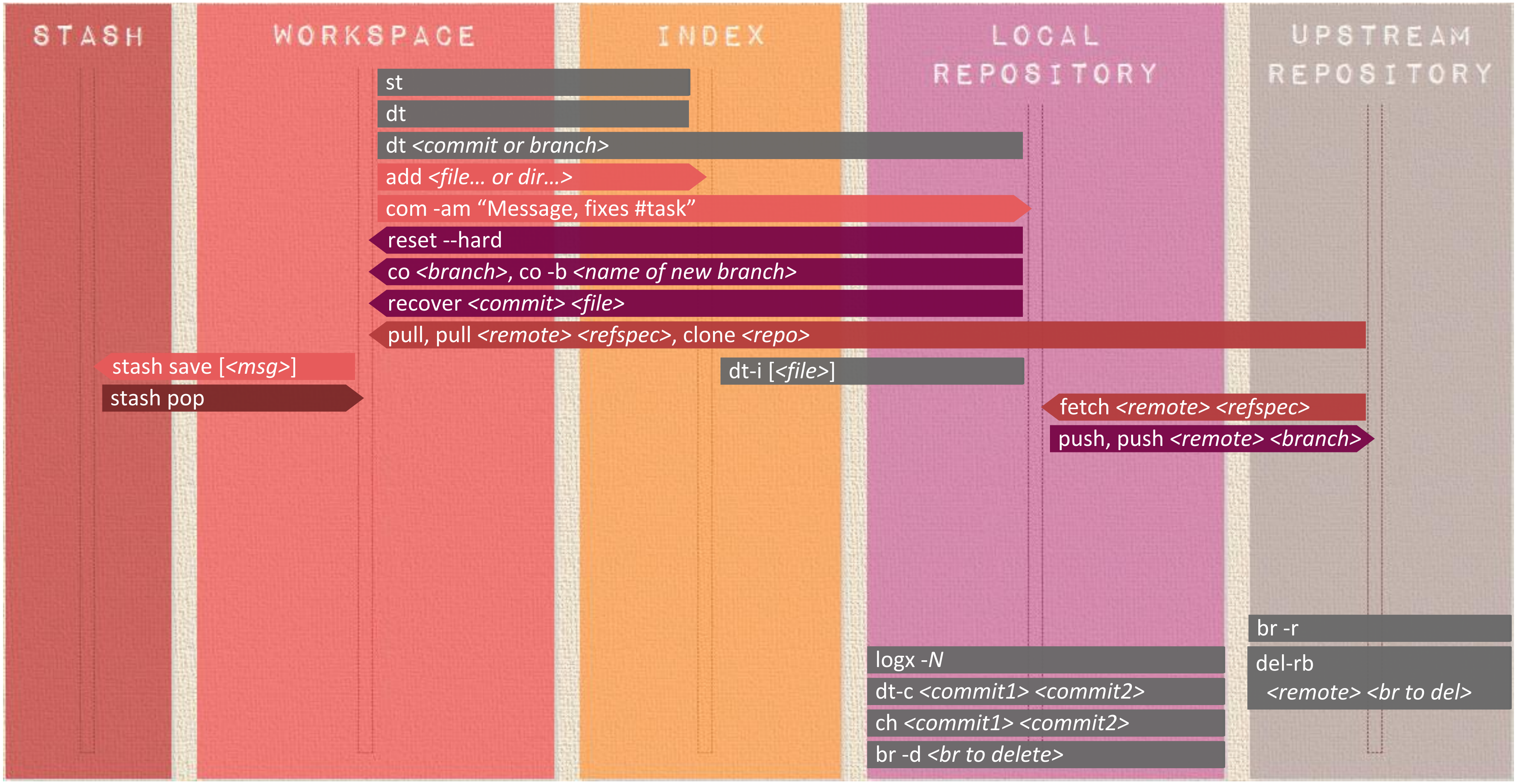
Revert
reset
co
revert
recover
undelete

Commit
com
pull
--amend
fetch
merge

Publish
push
tag

Based on work by Zack Rusin (http://zrusin.blogspot.com/2007/09/git-cheat-sheet.html)
Additional tips and documentation: http://try.github.com/, http://ndpsoftware.com/git-cheatsheet.html, http://git-scm.com/doc
Working with dates/revision selection: http://alexpeattie.com/blog/working-with-dates-in-git/

BTR Git Cheat Sheet



STASH

WORKSPACE

INDEX

LOCAL REPOSITORY

UPSTREAM REPOSITORY

- st
- dt
- dt *<commit or branch>*
- add *<file… or dir…>*
- com -am “Message, fixes #task”
- reset --hard
- co *<branch>*, co -b *<name of new branch>*
- recover *<commit> <file>*
- pull, pull *<remote> <refspec>*, clone *<repo>*
- stash save [*<msg>*]
- stash pop
- dt-i [*<file>*]
- fetch *<remote> <refspec>*
- push, push *<remote> <branch>*
- br -r
- logx -*N*
- del-rb *<remote> <br to del>*
- dt-c *<commit1> <commit2>*
- ch *<commit1> <commit2>*
- br -d *<br to delete>*

Basics

Stash – A place to hide modifications while you work on something else.

Workspace – Local check out / working directory.

Index – The Index (or “staging area”) holds a snapshot of the content of the working tree. This snapshot is taken as the contents of the next commit.

Local Repository – A subdirectory named .git that contains all of your necessary repository files – a Git repository skeleton. Typical branches: **master, _Test, feature-x, bugfix-y**.

Upstream Repository - Version(s) of your project that are hosted on the Internet or network, ensuring all your changes are available for other developers. Default is “origin”. Typical branches here: **master, shared-feature-x, release-y**.