

# AN INTRO TO GIT

**Orlando Code Camp 2018**

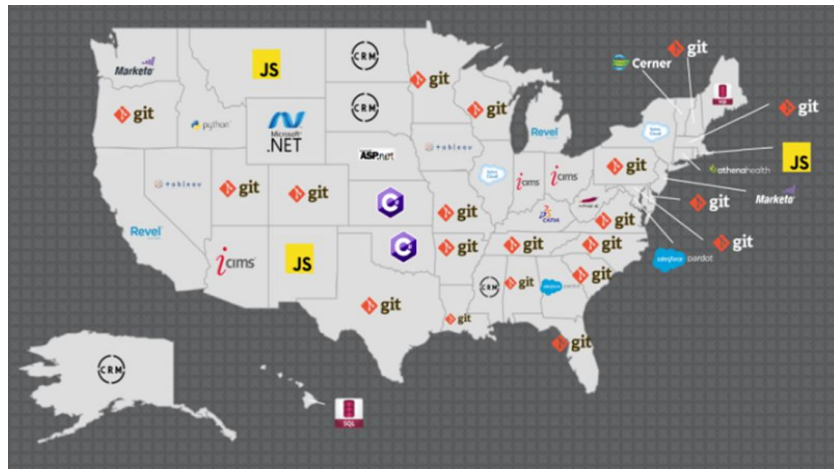
# MISNOMERS ABOUT GIT?

- It's too hard to use.
- It's only command line based.
- Takes too long to use.
- My project is too big and complicated for that.



# WHY USE THIS TOOL?

- Multiple backups fast
- Work in parallel on the same file
- Develop features in different 'versions' or branches
- Fast rollback and feature switching
- It's a skill in high demand



# MY PATH TO VERSION CONTROL

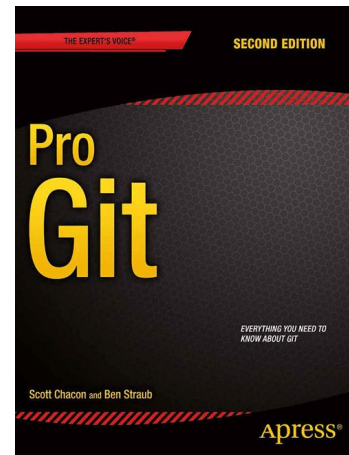
- Lots of information.
- What's the 'best'?
- How do I do 'X'?
- GIT, SVN, etc?
- What do I want out of version control?



Dropbox



SharePoint



# GIT WORDS

add	commit	fsck	mergetool	rev-list	tag
am	commit-tree	gc	mv	rev-parse	update-index
apply	config	gitk	pop	rm	update-ref
archive	count-objects	grep	prune	send-email	update-server-info
bisect	daemon	hash-object	pull	shortlog	verify-pack
blame	describe	help	push	show	write-tree
branch	diff	init	read-tree	show-ref	
bundle	diff-index	instaweb	rebase	stage	
cat-file	fast-import	log	reflog	stash	
checkout	fetch	ls-files	remote	status	
cherry-pick	filter-branch	ls-tree	request-pull	submodule	
clean	for-each-ref	merge	reset	svn	
clone	format-patch	merge-base	revert	symbolic-ref	

# GIT WORDS

<b>add</b>	<b>commit</b>	<b>fsck</b>	<b>mergetool</b>	<b>rev-list</b>	<b>tag</b>
am	commit-tree	gc	mv	rev-parse	update-index
apply	config	gitk	pop	rm	update-ref
archive	count-objects	grep	prune	send-email	update-server-info
bisect	daemon	hash-object	<b>pull</b>	shortlog	verify-pack
blame	describe	help	<b>push</b>	show	write-tree
<b>branch</b>	diff	<b>init</b>	read-tree	show-ref	
bundle	diff-index	instaweb	rebase	<b>stage</b>	
cat-file	fast-import	log	reflog	stash	
<b>checkout</b>	<b>fetch</b>	ls-files	<b>remote</b>	status	
cherry-pick	filter-branch	ls-tree	request-pull	submodule	
clean	for-each-ref	<b>merge</b>	<b>reset</b>	svn	
<b>clone</b>	format-patch	merge-base	revert	symbolic-ref	

TOOL THAT WE'LL USE



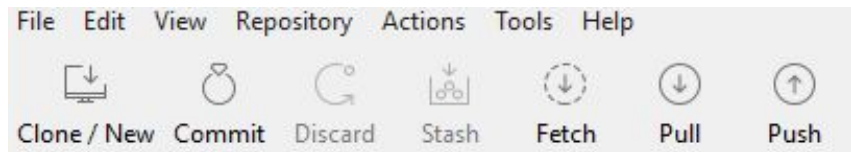
LET'S START !



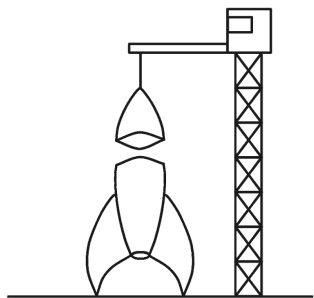
# CLONE/ INIT

Creates a new repo or copies an existing one

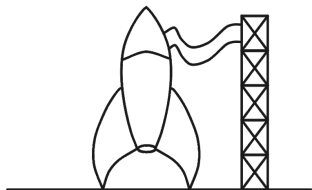
You can always add 'remotes' but it's easier to start from there.



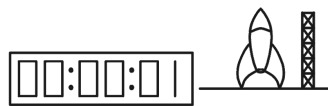
# TYPICAL WORKFLOW



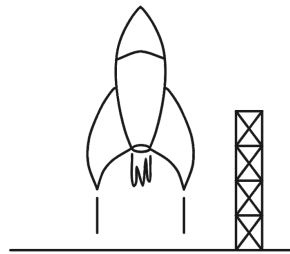
ADD



STAGE



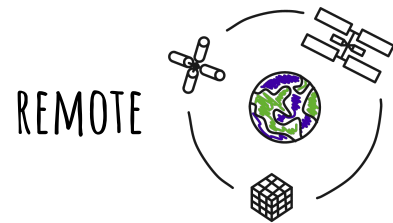
COMMIT



PUSH



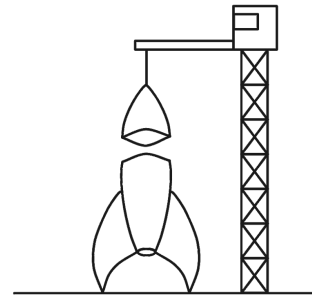
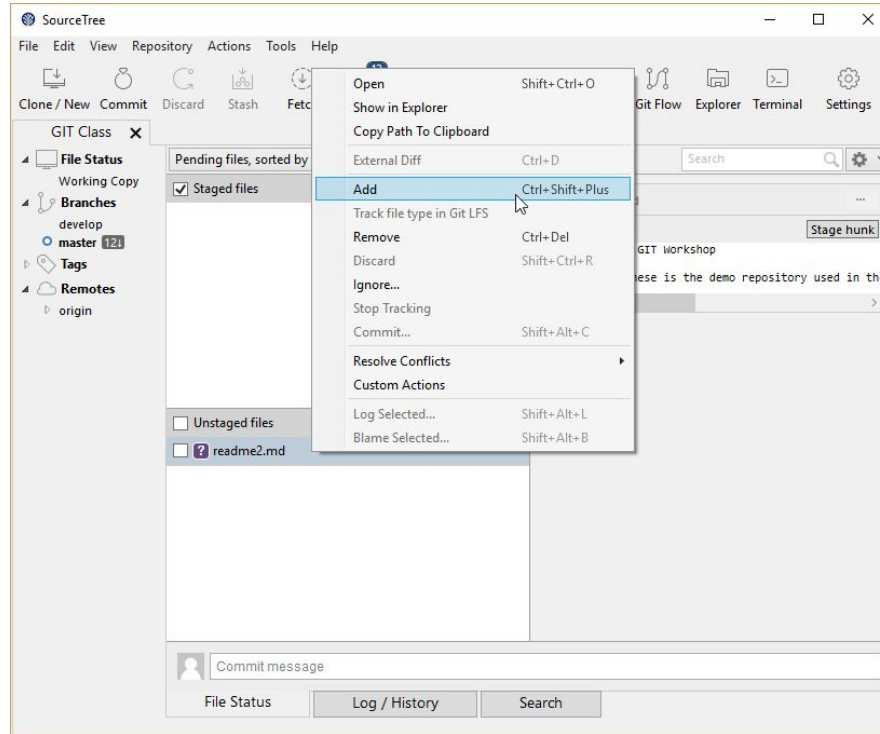
PULL



# ADD

Tells GIT to start tracking the files.

Ignore allows you to not track files in a folder - compiled files or temp files typically.



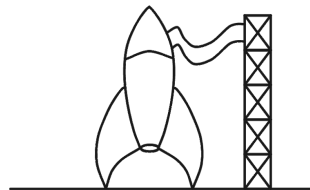
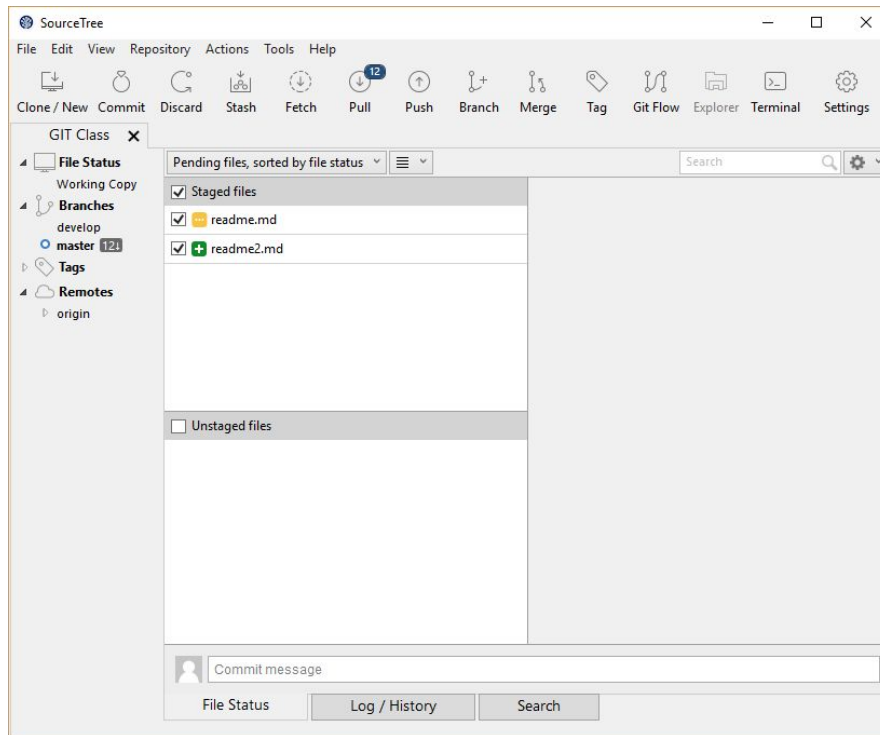
# ADD

# STAGE

Lets you get your commit where you want it.

TIP:

You can split your changes into multiple commits for tracking.

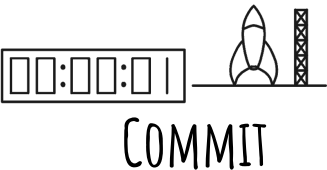
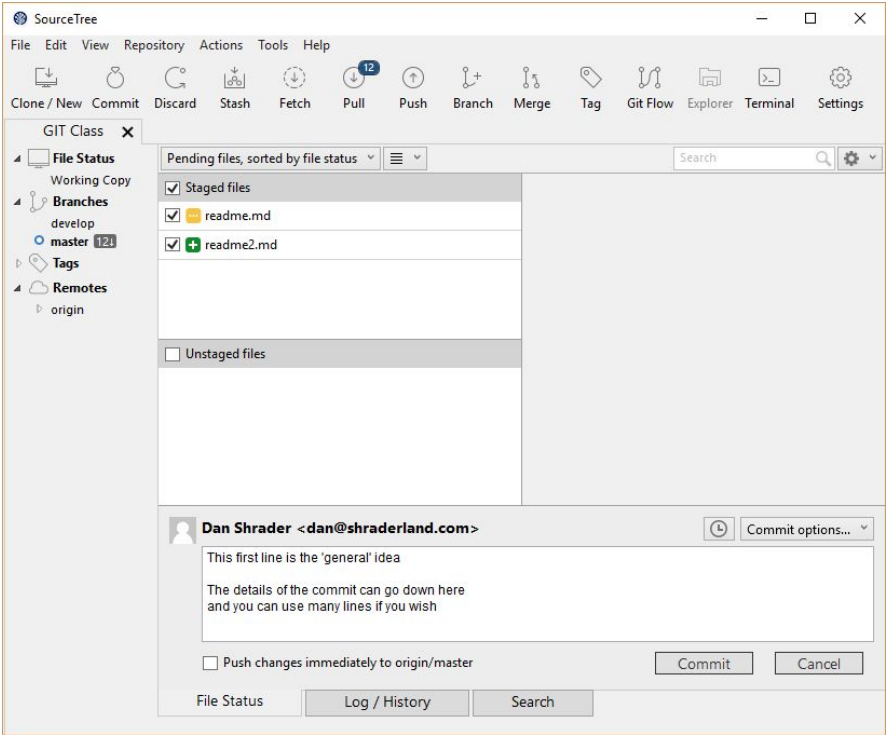


STAGE

# COMMIT

Snapshot in time of your project.

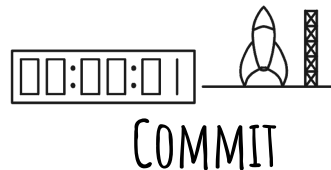
Be detailed on your descriptions. You'll thank yourself later and so will the other contributors to the project.



# COMMIT

	COMMENT	DATE
○	CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
○	ENABLED CONFIG FILE PARSING	9 HOURS AGO
○	MISC BUGFIXES	5 HOURS AGO
○	CODE ADDITIONS/EDITS	4 HOURS AGO
○	MORE CODE	4 HOURS AGO
○	HERE HAVE CODE	4 HOURS AGO
○	AAAAAAAAA	3 HOURS AGO
○	ADKFJSLKDFJSDKLFJ	3 HOURS AGO
○	MY HANDS ARE TYPING WORDS	2 HOURS AGO
○	HAAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT  
MESSAGES GET LESS AND LESS INFORMATIVE.

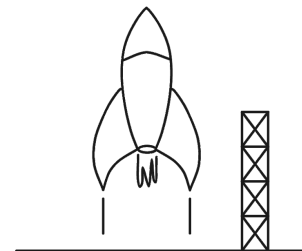
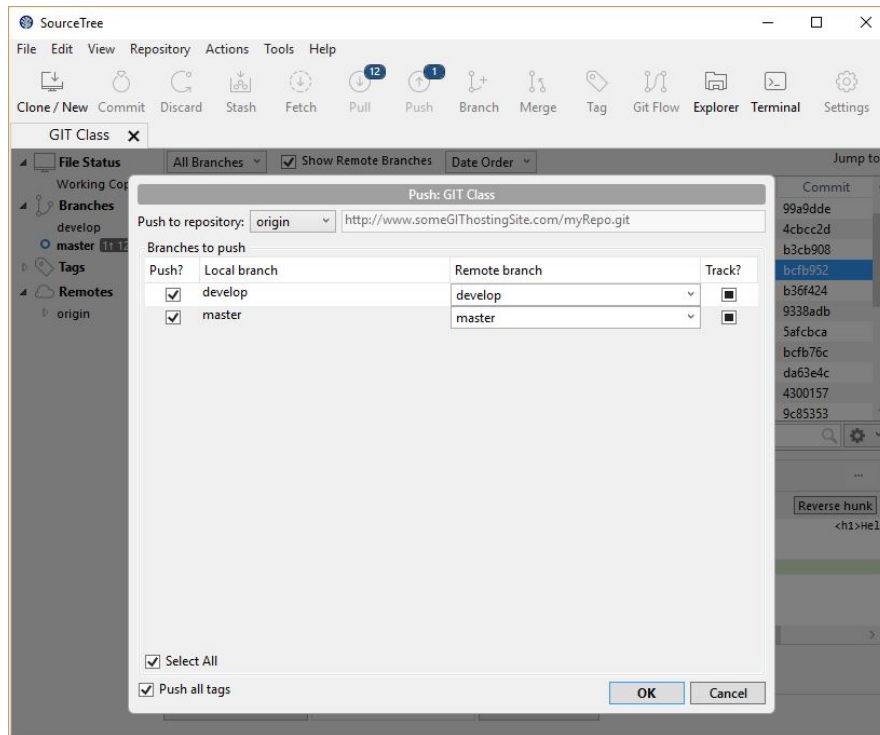


COMMIT

# PUSH

Sends your commits to a centralized server.

Note: there are different types of workflows. We are referencing the 'centralized' workflow.



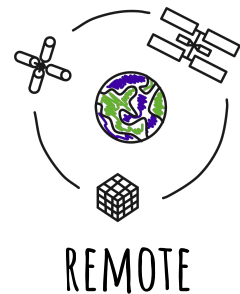
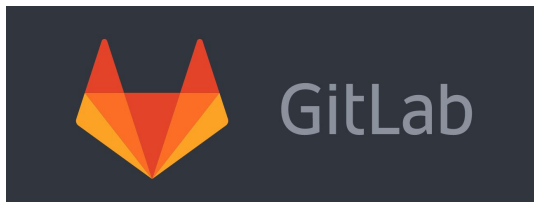
PUSH

# REMOTE

A place where your  
code is hosted.



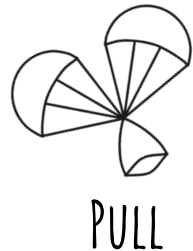
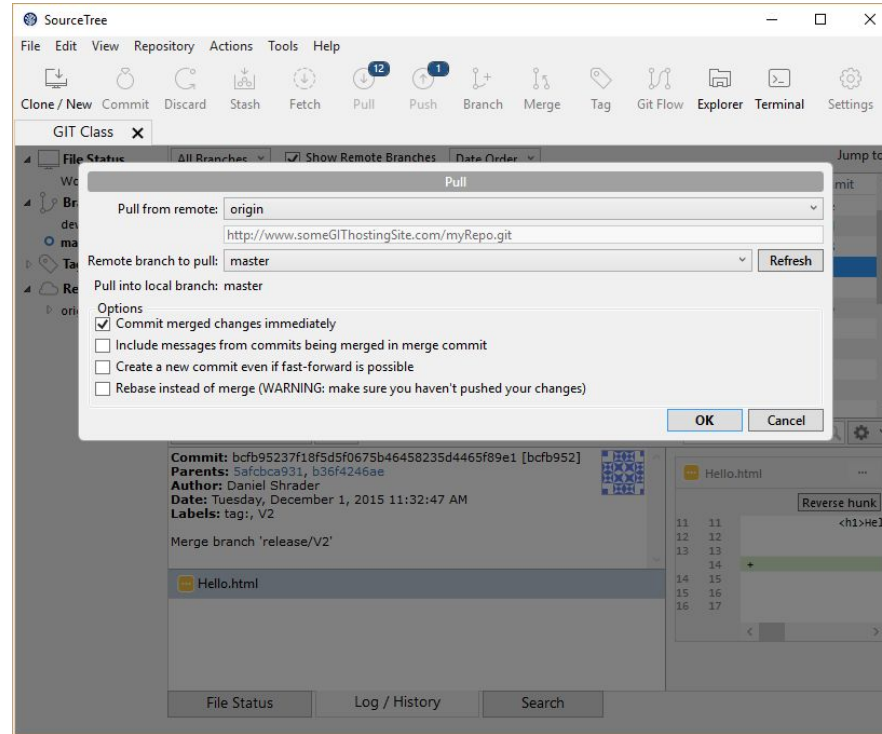
# GitHub



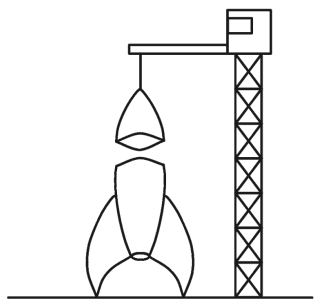


# PULL

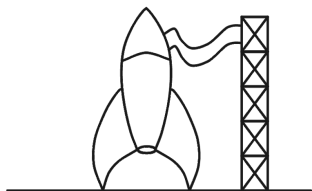
Brings changes committed to the remote into your branch.



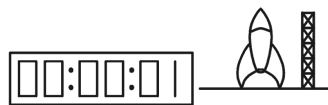
# WORKFLOW ONCE MORE



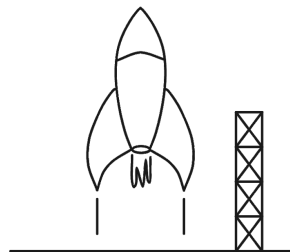
ADD



STAGE



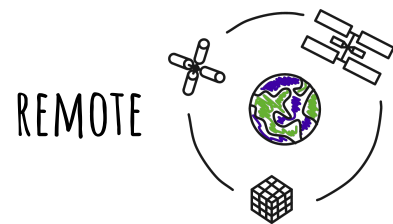
COMMIT



PUSH

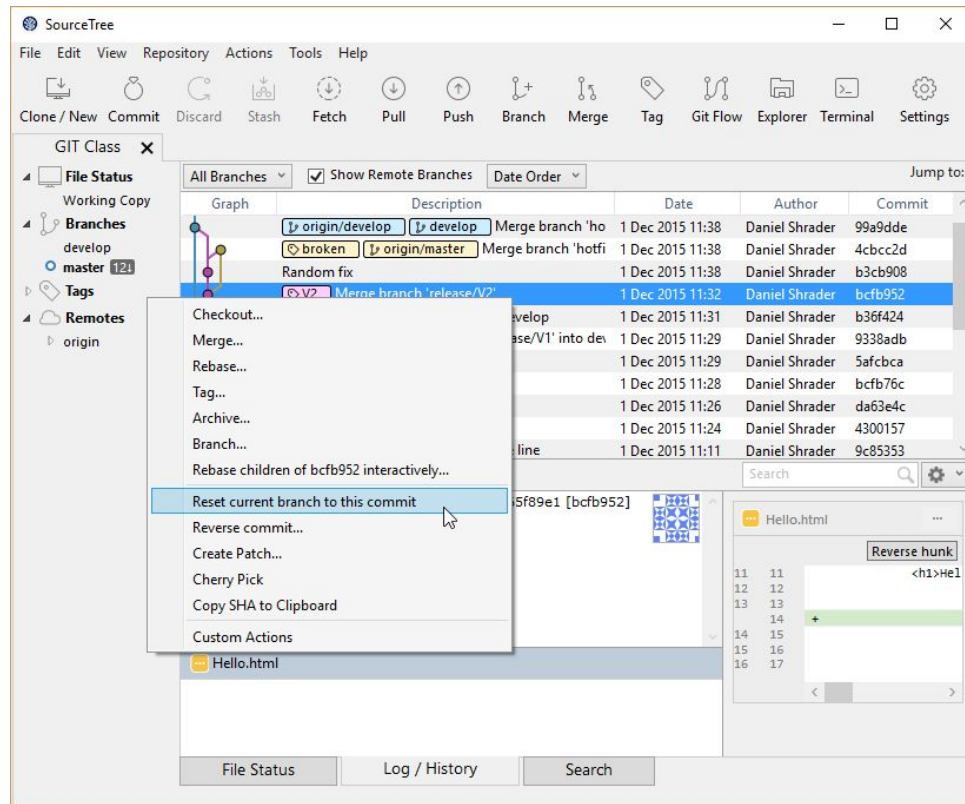
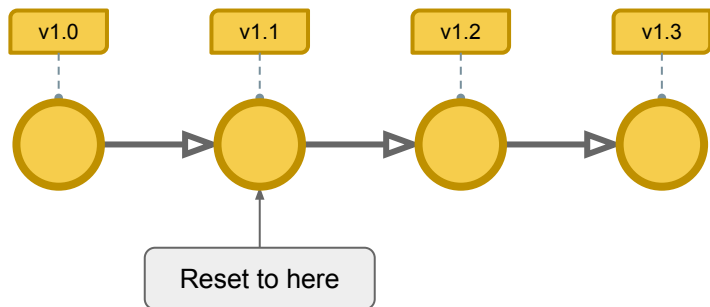


PULL



# RESET

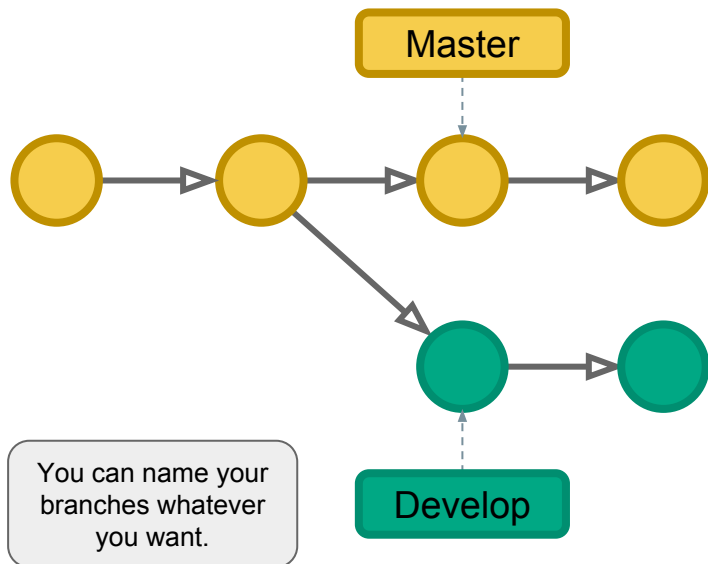
Lets you move around in time on your project. Really useful when looking for bug introductions.



BRANCHES!

# BRANCH

A place to work which does not affect the main project.



SourceTree

File Edit View Repository Actions Tools Help

Clone / New Commit Discard Stash Fetch Pull Push Branch Merge Tag Git Flow Explorer Terminal Settings

GIT Class

File Status Working Copy Branches develop master 12 Tags Remotes origin

Graph	Description	Date	Author	Commit
o	Awesome Commit	1 Dec 2015 11:24	Daniel Shrader	4300157
o	origin/Branch1 Added Duplicate line	1 Dec 2015 11:11	Daniel Shrader	9c85353
o	Commit 2	1 Dec 2015 11:06	Daniel Shrader	58d5eb8
o	Commit 1	1 Dec 2015 11:05	Daniel Shrader	54a8710
o	master 12 behind Added Readme	30 Nov 2015 11:35	Daniel Shrader	2dc551c
o	Implemented Bootstrap to HTML	30 Nov 2015 11:18	Daniel Shrader	00316a3
o	Added Bootstrap files	30 Nov 2015 11:14	Daniel Shrader	c0db284
o	Fix Typo	30 Nov 2015 10:09	Daniel Shrader	a2f258b
o	Added Content	30 Nov 2015 10:08	Daniel Shrader	e55ccbd
o	Added HTML Structure	30 Nov 2015 10:06	Daniel Shrader	1f3137a
o	Initial Commit	30 Nov 2015 10:00	Daniel Shrader	cf4846b

Sorted by file status

Commit: 430015737dc717653635076b5e54ea6873b26d34 [4300157]  
Parents: 9c8535338f  
Author: Daniel Shrader  
Date: Tuesday, December 1, 2015 11:24:00 AM  
Awesome Commit  
Better Message

Hello.html

File Status Log / History Search

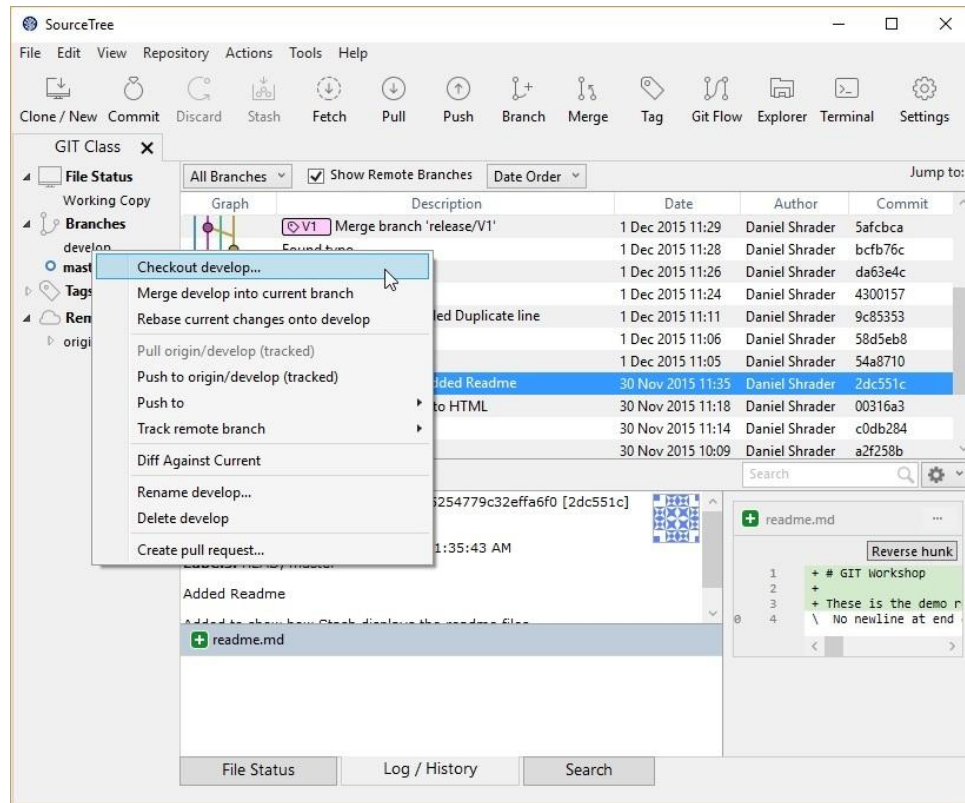
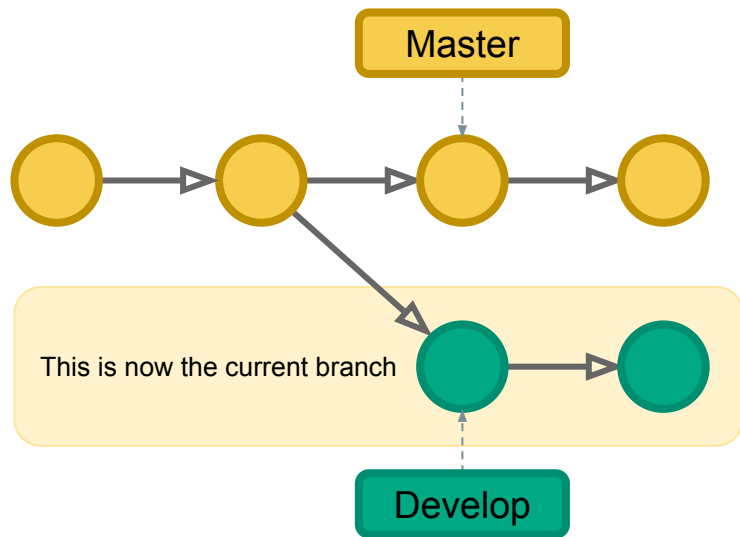
Hello.html

Reverse hunk

```
11 11      <h1>Hello GIT!</h1>
12 12      <h2>Hi</h2>
13 13      <h2>Hi</h2>
14 14      + <p>Jira 1</p>
15 15      </div>
16 17      <p> This is a
```

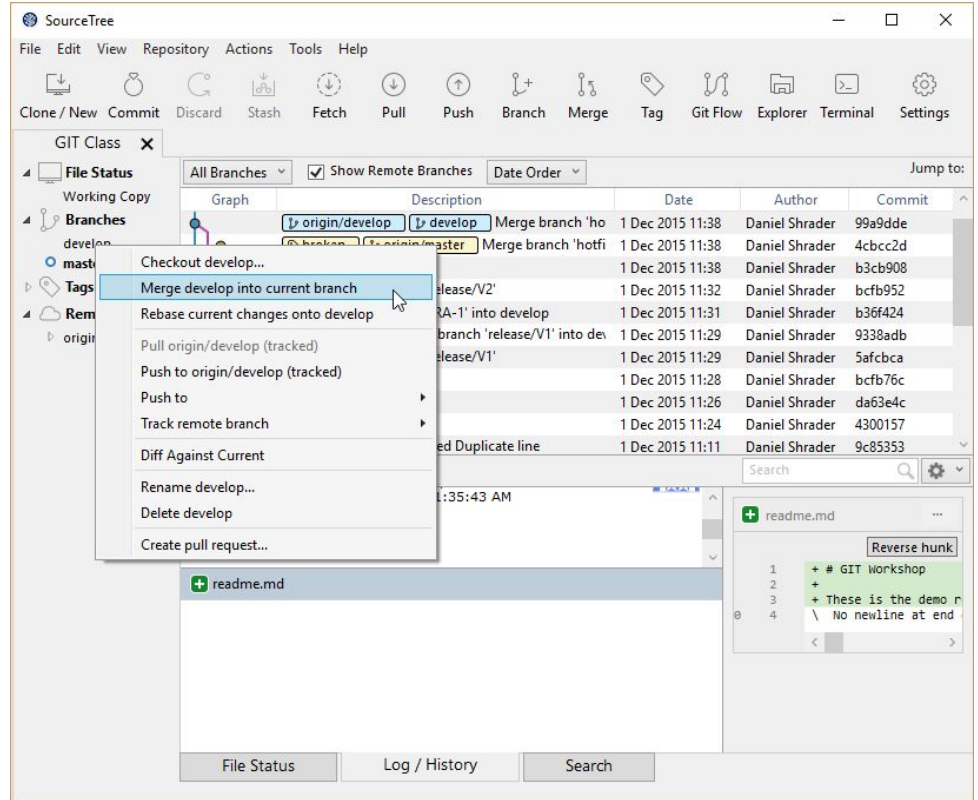
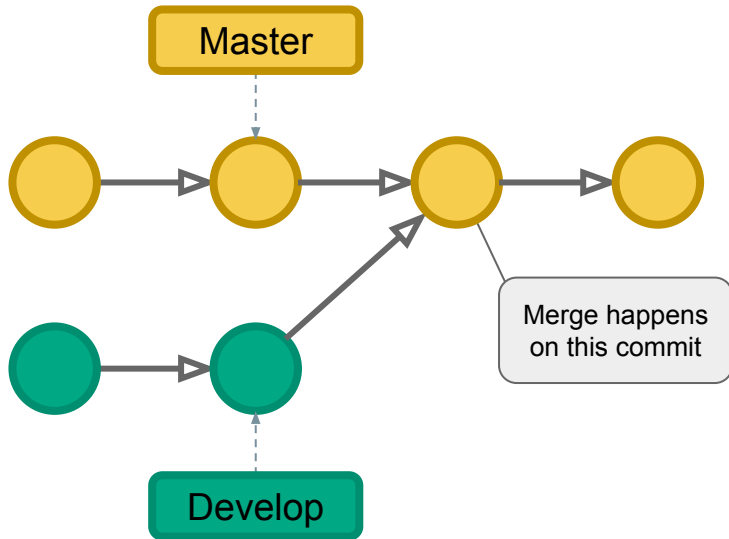
# CHECKOUT

Switches the branch you are on.



# MERGE

Brings a branch into another. Be cautious of conflicts.

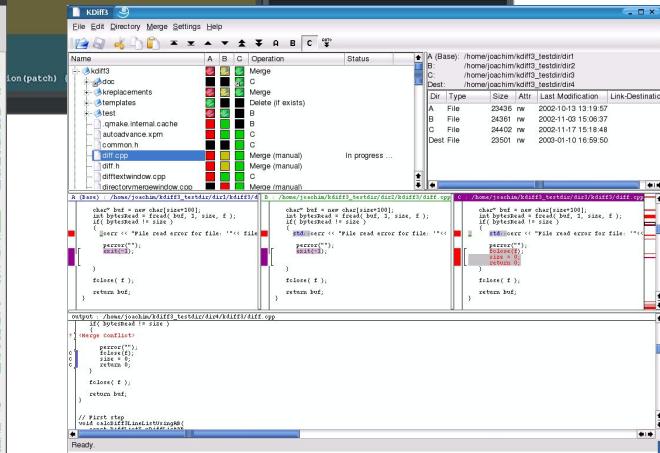
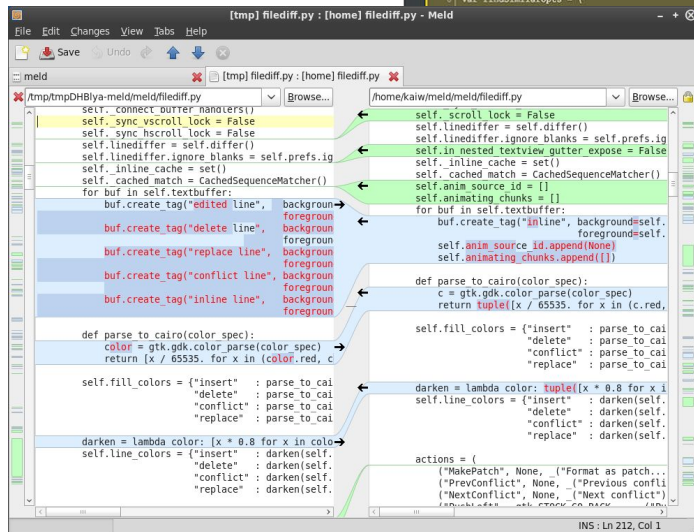
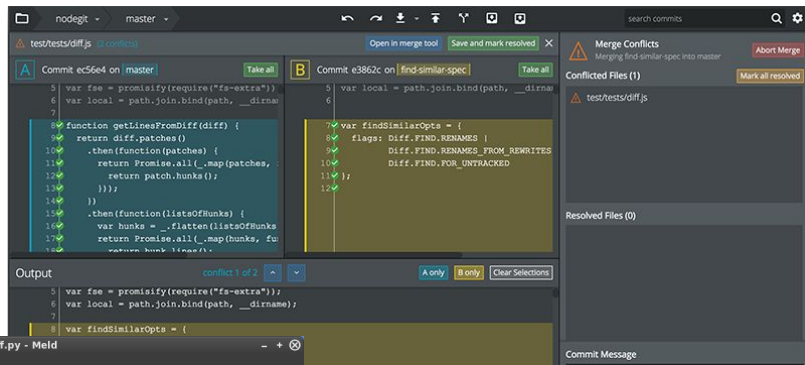


# MERGETOOL - CONFLICTS

Simply your tool of choice to deal with the eventual code conflicts

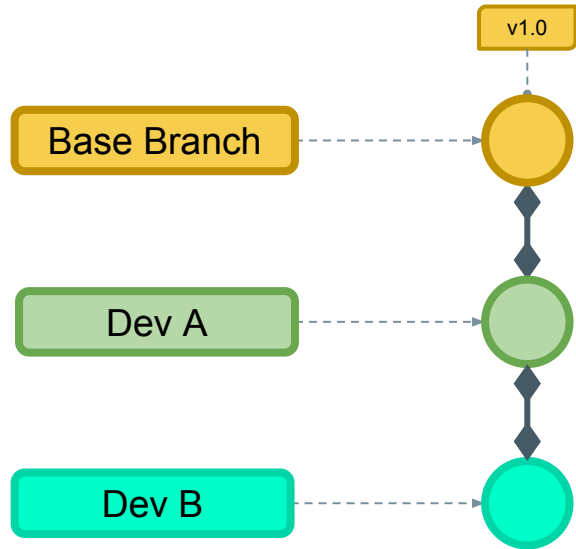
A few are:

- meld
- kdiff3
- gitKraken

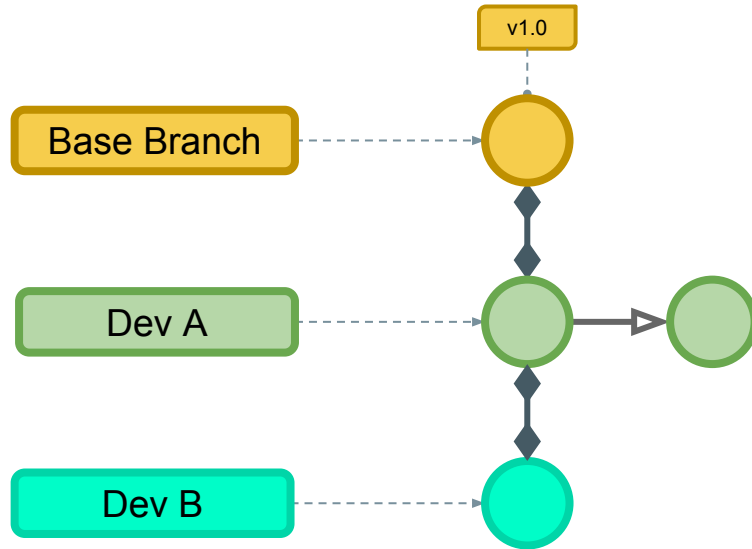




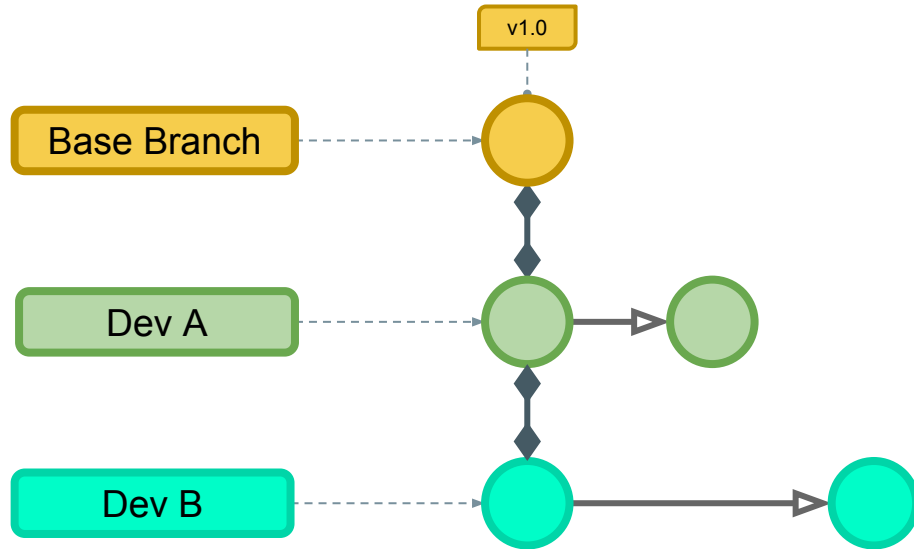
# MERGE - CONFLICTS



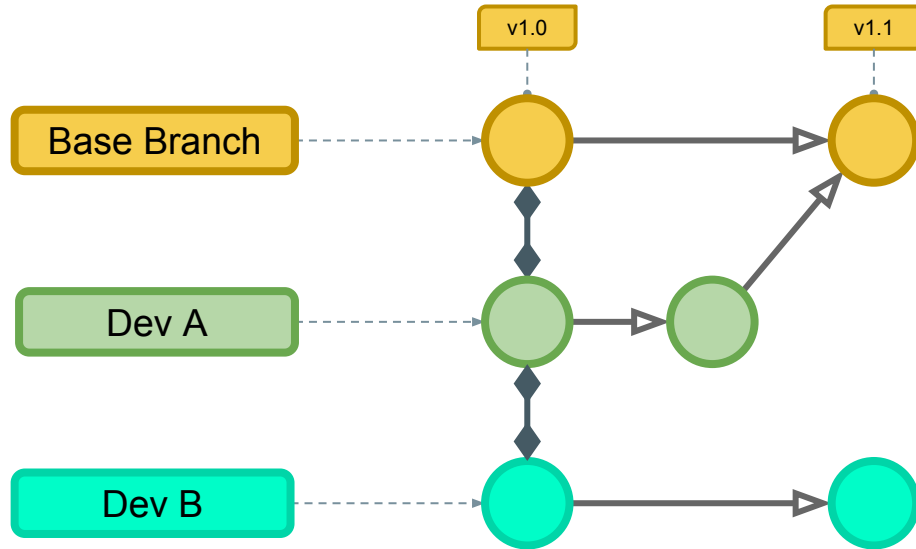
# MERGE - CONFLICTS



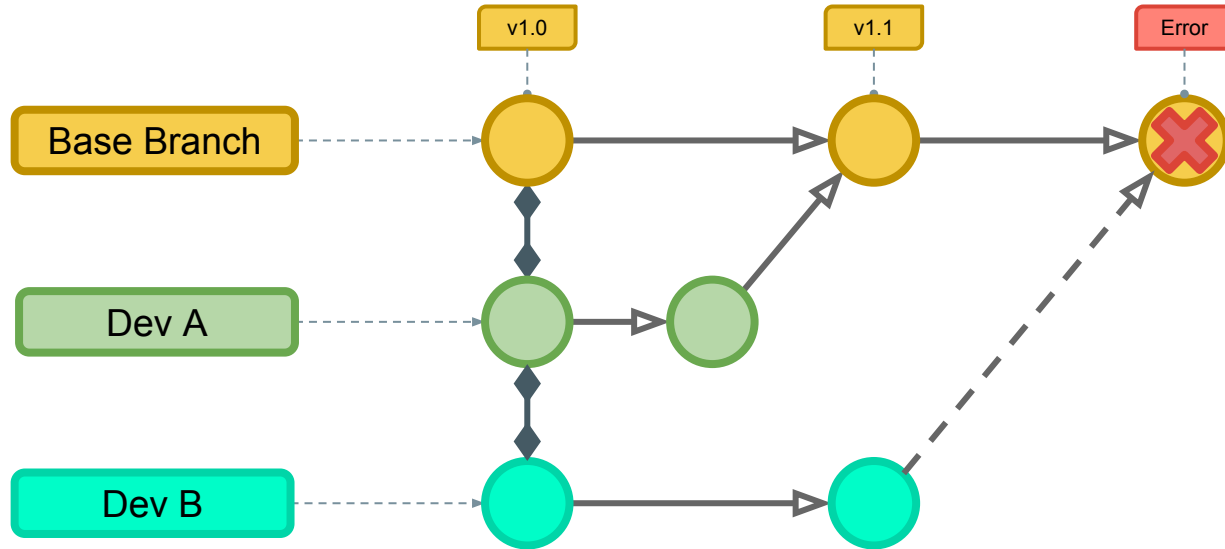
# MERGE - CONFLICTS



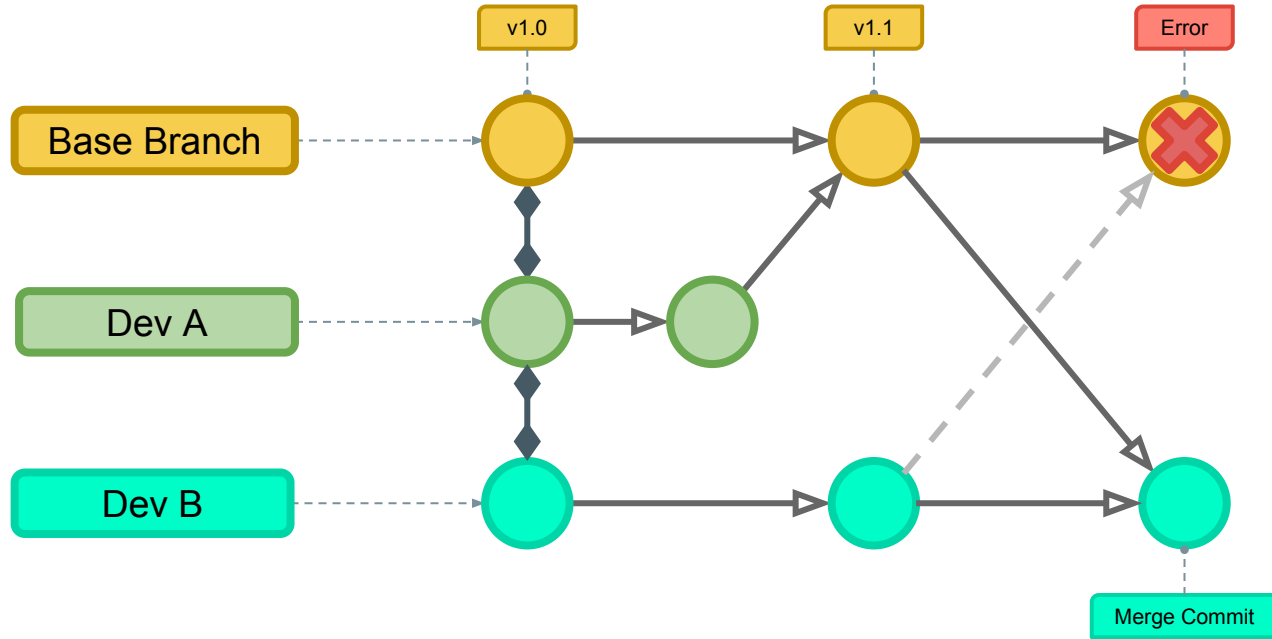
# MERGE - CONFLICTS



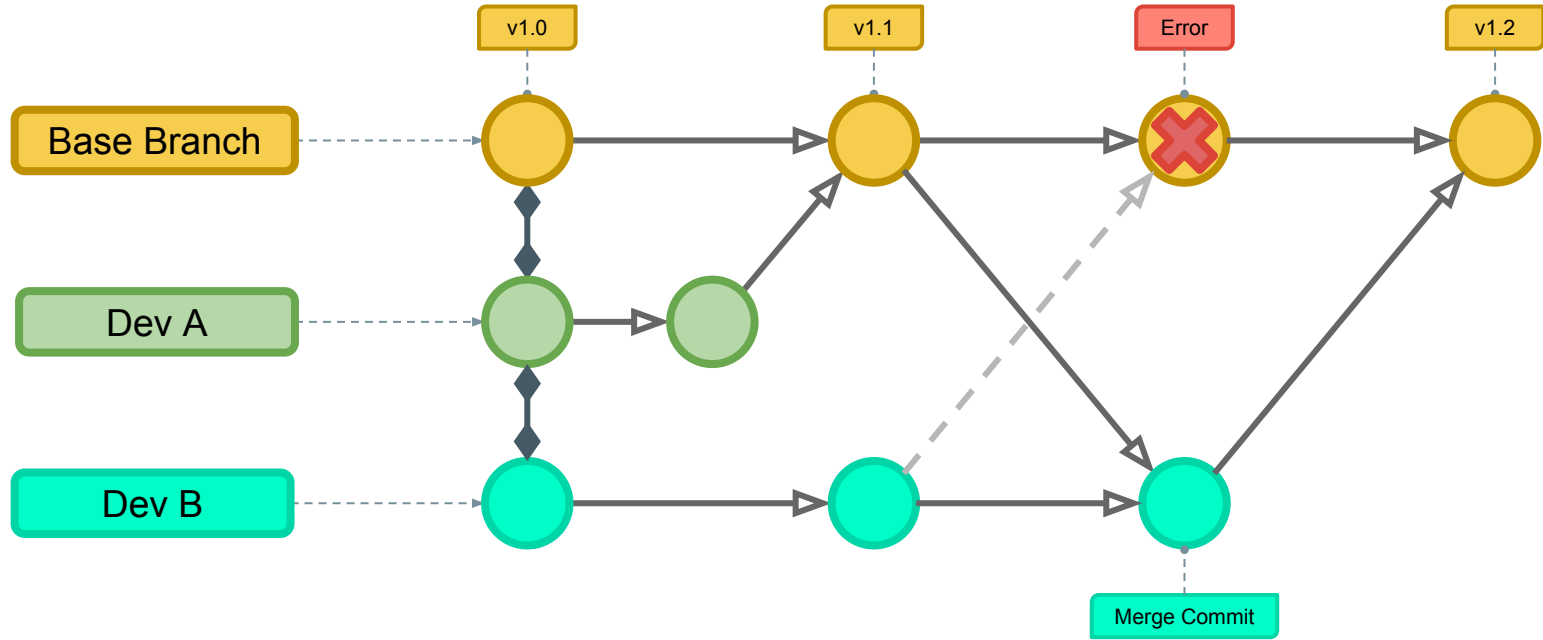
# MERGE - CONFLICTS



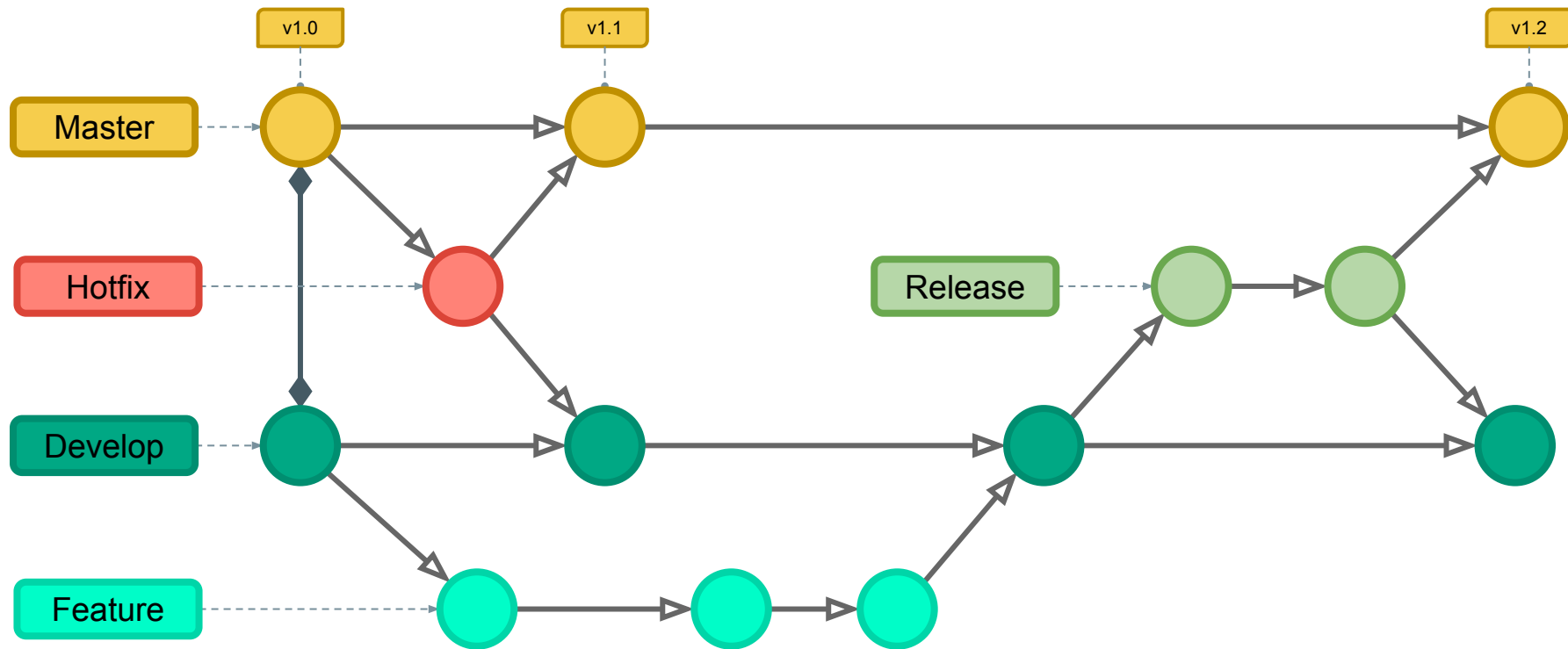
# MERGE - CONFLICTS



# MERGE - CONFLICTS



# GIT FLOW WORKFLOW

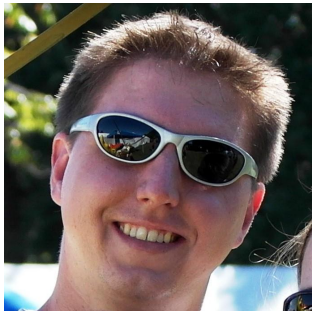




# APPENDIX

- <https://www.atlassian.com>
- <http://www.mnu.edu/business/software-skills-demand>
- <https://xkcd.com/1296/>
- <https://xkcd.com/1597/>
- <http://danielkummer.github.io/git-flow-cheatsheet/>
- <https://training.github.com/classes/essentials/>
- <https://www.gitlab.com/>
- <https://github.com/>
- <https://www.sourcetreeapp.com/>
- <https://blog.axosoft.com/>
- <http://meldmerge.org/>
- <http://kdiff3.sourceforge.net/>
- <https://git-scm.com>
- <https://arstechnica.com/information-technology/2017/05/90-of-windows-devs-now-using-git-creating-1760-windows-builds-per-day/>

# ABOUT ME



I've been dabbling with code since the late 90's - currently work with Microsoft Business Intelligence products and Amazon Web Services. When given the chance, I write single page applications in JavaScript. I'm also a stickler for automating as much of the day to day tasks as I can. When I'm not coding I enjoy camping and hanging out with my wife and sons.

Email: [Dan@ShraderLand.com](mailto:Dan@ShraderLand.com)