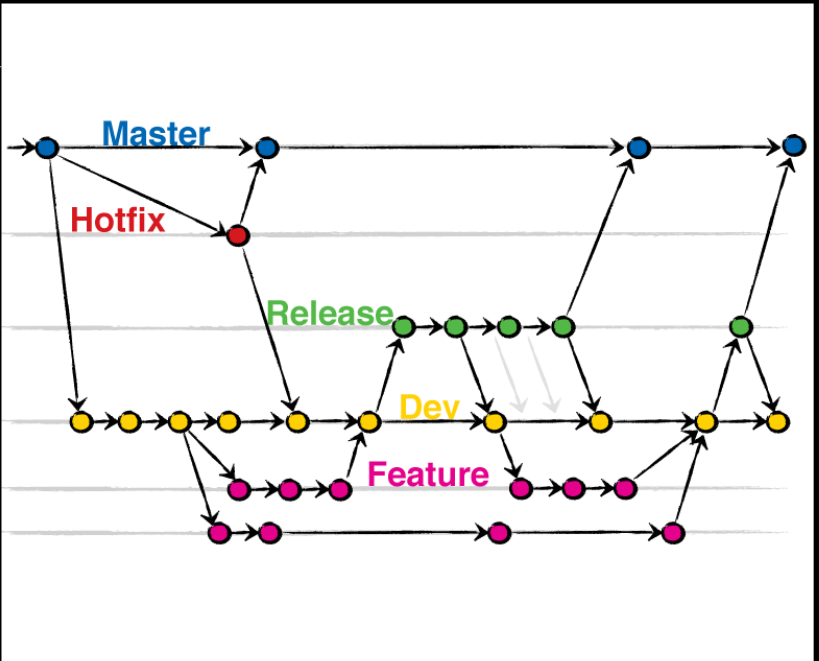


Release branches	Hotfix branches	Feature branches
From: develop To: develop и master	From: master To: master, develop and/or release	From: develop To: develop
release-*	hotfix-*	any, except master, develop, release-* or hotfix-*
Preparing for release. Creating metadata for release (version, build date, etc.). Release takes version number at Release branch only, not before!	Update version number after creation of branch! When the bug is fixed changes merge to Master and Develop or Release (if it exist) .	For new features to current or future release. Can be unknown in what release it will be added. Exist while developing. Merge to Develop or remove. Usually exist at developers repositories only. Not in origin/master .
Make branch with new version name:	Make branch, changes metadata:	Merge with develop after developing:
<pre>\$ git checkout -b release-1.2 develop</pre> <pre>\$./bump-version.sh 1.2</pre> Files modified successfully, version bumped to 1.2. <pre>\$ git commit -a -m "Bumped version number to 1.2"</pre>	<pre>\$ git checkout -b hotfix-1.2.1 master</pre> <pre>\$./bump-version.sh 1.2.1</pre> Files modified successfully, version bumped to 1.2.1. <pre>\$ git commit -a -m "Bumped version number to 1.2.1"</pre>	<pre>\$ git checkout develop</pre> <pre>\$ git merge --no-ff myfeature</pre> <pre>\$ git branch -d myfeature</pre> <pre>\$ git push origin develop</pre>
Closing branch when it's ready for release!	Closing branch	 <p>The diagram illustrates the Git branching strategy. It shows five horizontal tracks representing branches: Master (blue), Hotfix (red), Release (green), Dev (yellow), and Feature (pink). Master has a sequence of blue dots representing production-ready releases. Dev has a sequence of yellow dots representing the current state of integration. Feature branches (pink) are used for developing new features, branching off from Dev and merging back to Dev upon completion. Hotfixes (red) are used to quickly address production issues, branching off from Master and merging back to both Master (to create a new production version) and Dev (to update the integration branch). Releases (green) represent production-ready versions that are also reflected in the Dev branch.</p>
1. Merge to master (every commit to master = new release!) <pre>\$ git checkout master</pre> <pre>\$ git merge --no-ff release-1.2</pre> 2. Tagging commit <pre>\$ git tag -a 1.2</pre> 3. Add changes from release branch to develop <pre>\$ git checkout develop</pre> <pre>\$ git merge --no-ff release-1.2</pre> 4. Delete release-1.2 branch: <pre>\$ git branch -d release-1.2</pre>	1. Обновляем master, добавляем тег: <pre>\$ git checkout master</pre> <pre>\$ git merge --no-ff hotfix-1.2.1</pre> <pre>\$ git tag -a 1.2.1</pre> 2. Переносим исправление в ветвь develop: <pre>\$ git checkout develop</pre> <pre>\$ git merge --no-ff hotfix-1.2.1</pre> 3. Удаляем ветвь: <pre>\$ git branch -d hotfix-1.2.1</pre>	