

Git Branching Guide (With Git Bash)

- Summary Diagram
- Start Of Sprint - Create New Story Branch
- Story Complete - Merge Story Branch To Develop
- Sprint Complete - Merge Develop To Master
- Post Sprint Clean Up - Delete Story Branches
- Create Release

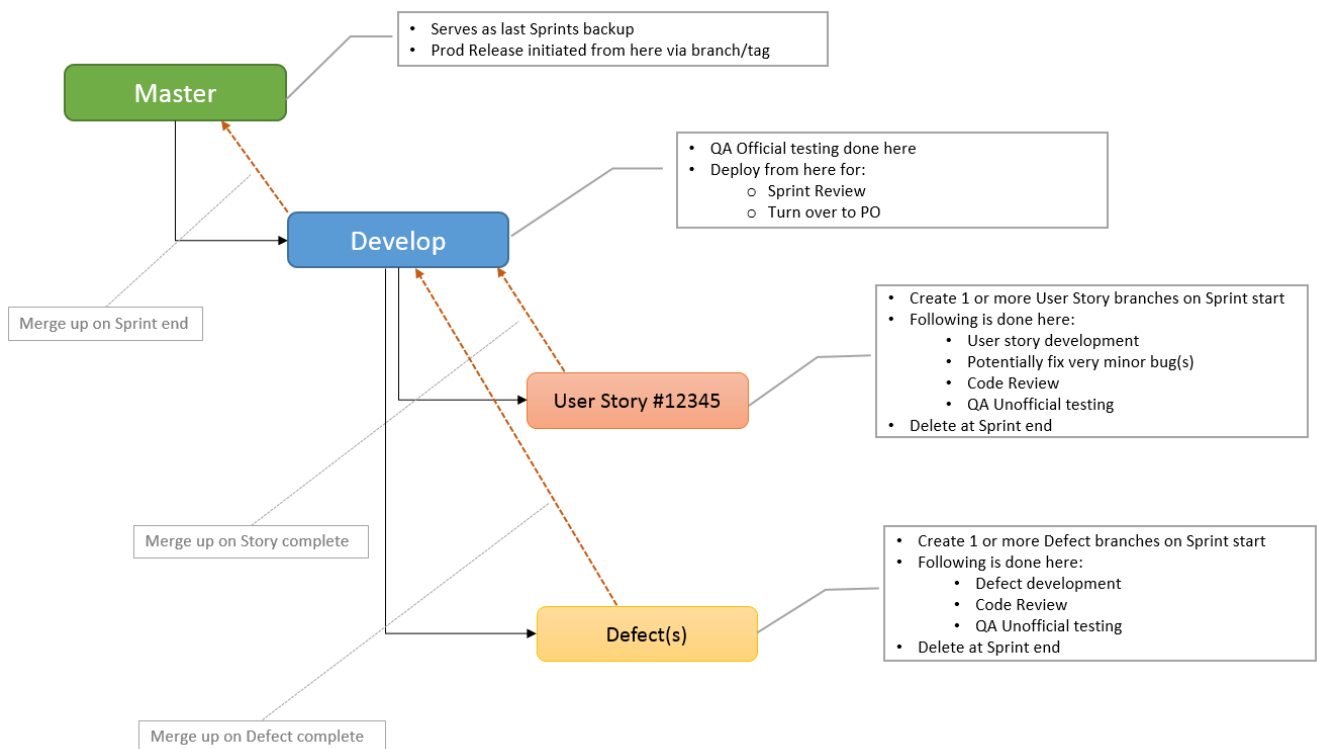
This page covers the branching strategy to be used by Custom. It is based on the [Gitflow](#) workflow.



Run the following in Git Bash to store off your credentials:

```
git config --global credential.helper wincred
```

Summary Diagram



Start Of Sprint - Create New Story Branch

1. Checkout develop and ensure it is up to date.

```
git checkout develop
git pull
```

2. Create new feature branch from develop for story and switch to branch.

```
git checkout -b [name of new branch]
```

3. Push the branch to origin.

```
git push origin [name of new branch]
```

Story Complete - Merge Story Branch To Develop



Before merging your story to develop, ensure your code has been reviewed via a [pull request](#).

NB - the base for the pull request should be **develop**, not master.

1. Merge origin/develop into story branch.

```
git fetch
git merge origin/develop
```

2. Fix any merge conflicts and commit.
3. Run full build against feature branch, running all unit/integration tests (including Grunt linting and JS unit tests).
4. Push branch to origin

```
git push origin [name of story branch]
```

5. Ensure your code is reviewed via a pull request
6. Checkout develop

```
git checkout develop
```

7. Pull latest version into develop.

```
git pull
```

8. Merge story branch into develop.

```
git merge [name of story branch]
```



This should always be a fast-forward merge. If any merge conflicts appear at this point then roll back the merge (`git merge --abort`) and merge the latest develop into the story branch again.

9. Push the merge to develop.

```
git push origin develop
```

Sprint Complete - Merge Develop To Master

1. Check all story branches have been merged to develop. Command below lists all branches which haven't been merged.

```
git branch --no-merged origin/develop
```

2. Merge develop to master (this should always be a fast forward merge)

```
git checkout master  
git pull  
git merge origin/develop
```

3. Push the merge to master.

```
git push origin master
```

Post Sprint Clean Up - Delete Story Branches

1. Delete story branches locally.

```
git branch -d [name of story branch]
```

2. Delete story branches from origin.

```
git push origin :[name of story branch]
```

3. Update list remote branch.

```
git remote update origin --prune
```

4. List both local and remote branches that have not been merged - useful in determining which branches are ready to be deleted

```
git branch -a --no-merged
```

Create Release

1. Checkout master and update to latest version.

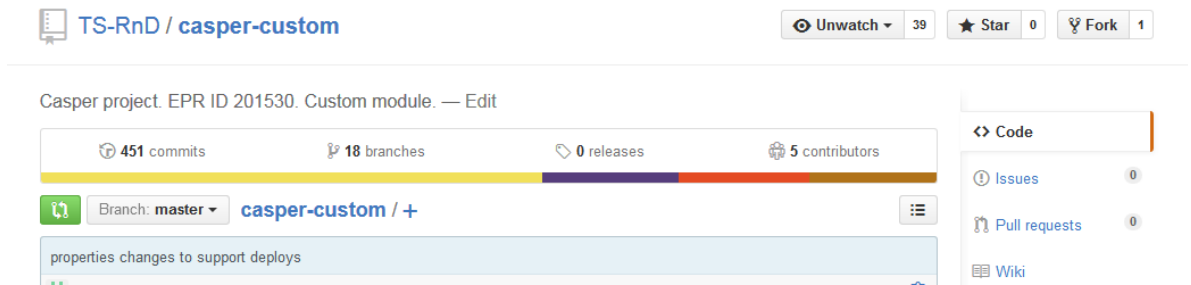
```
git checkout master  
git pull
```

2. Update master branch pom versions to release version.
3. Push the master branch to origin.

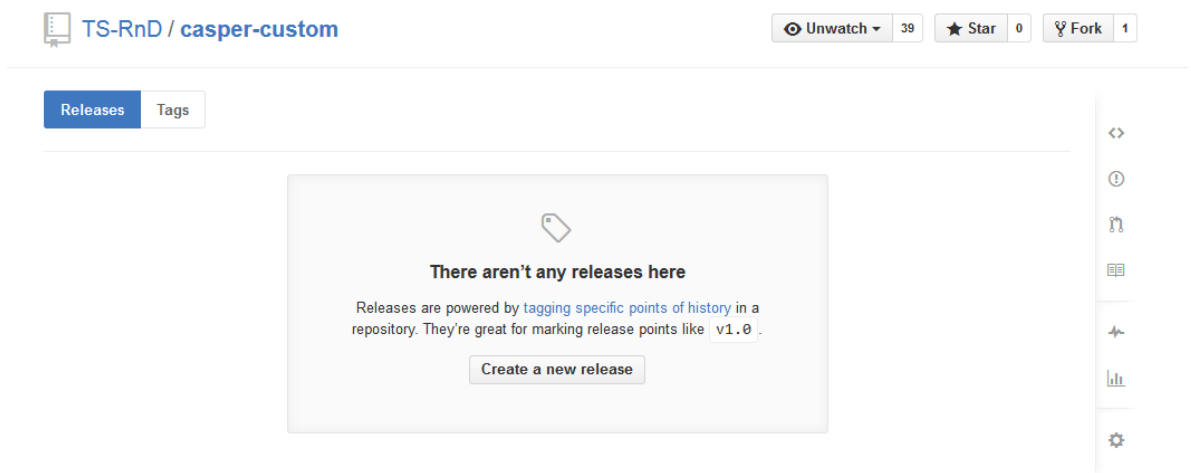
```
git push origin master
```

4. Run Jenkins build against master (uploads release artifacts to nexus).

5. Create release in GitHub:
- Click on the **Releases** tab



- Click **Create a new release**



- Fill in the release details and click **Publish release**

Releases

Tags

v.1.5.1

@

Target: master

Excellent! This tag will be created from the target when you publish this release.

Version 1.5.1 of Casper Custom

Write

Preview

Markdown supported

Introduces a lot of cool stuff.

Attach images by dragging & dropping or [selecting them](#).



Attach binaries by dropping them here or [selecting them](#).

☐ This is a pre-release

We'll point out that this release is identified as non-production ready.

Publish release

Save draft

Tagging suggestions

It's common practice to prefix your version names with the letter v. Some good tag names might be v1.0 or v2.3.4.

If the tag isn't meant for production use, add a pre-release version after the version name. Some good pre-release versions might be v0.2-alpha or v5.9-beta.3.

Semantic versioning

If you're new to releasing software, we highly recommend reading about [semantic versioning](#).

6. Update the develop poms to the next snapshot version.