Branching Policies

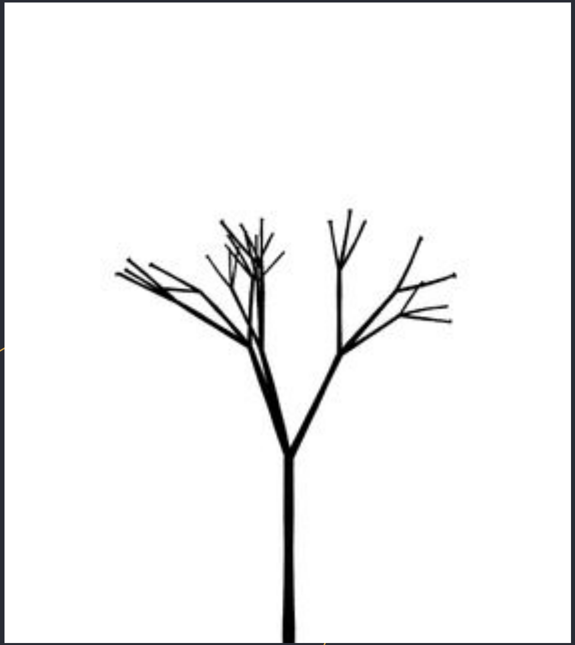
Òscar Canales · CITM Student · @Osvak

TABLE OF CONTENTS

1

INTRODUCTION

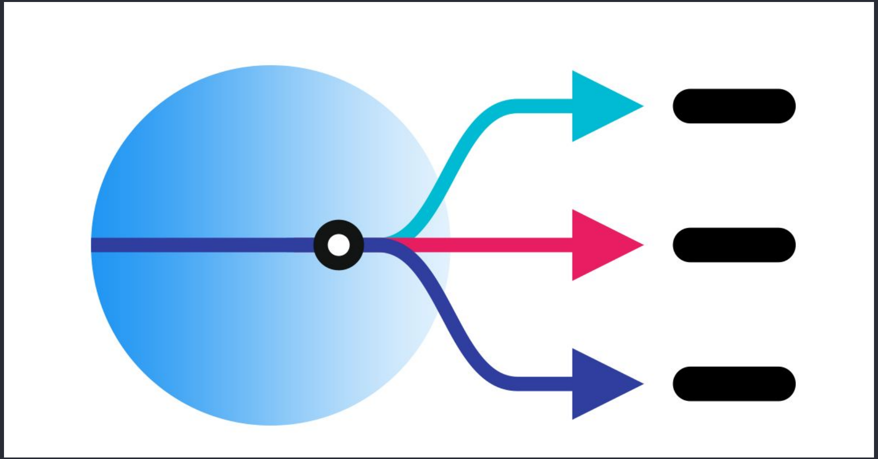
INTRODUCTION



“Divide et Impera” - Julius Caesar

Concept

- Deviating
- Continue
- Avoid conflicts



Git, is not just a name

- Efficiency
- Snapshot system
- “Killer feature”

Git's logo

2

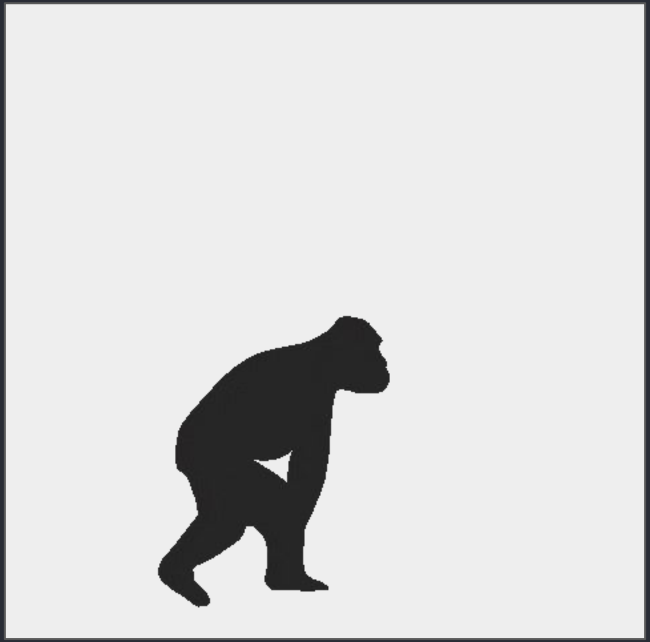
WHY ?

WHY ? - Do we need branches ?

- “The Master branch is already good”
- When you get in a bigger project this won’t work

WHY ? - Personal Benefits

- Powerful tool you need to learn
- The sooner the better



WHY ? - Pros

- Isolate the work from the main branch
- Limits who can contribute to each branch
- Simplifies the QA and bug fix process
- Ensures that a change to a branch must be reviewed
- Branches are cheap
- Agile Workflow

WHY ? - Cons

- For very small projects can makes the process more complicated
- There isn’t a “one fits all” Flow model

3

WHEN ?

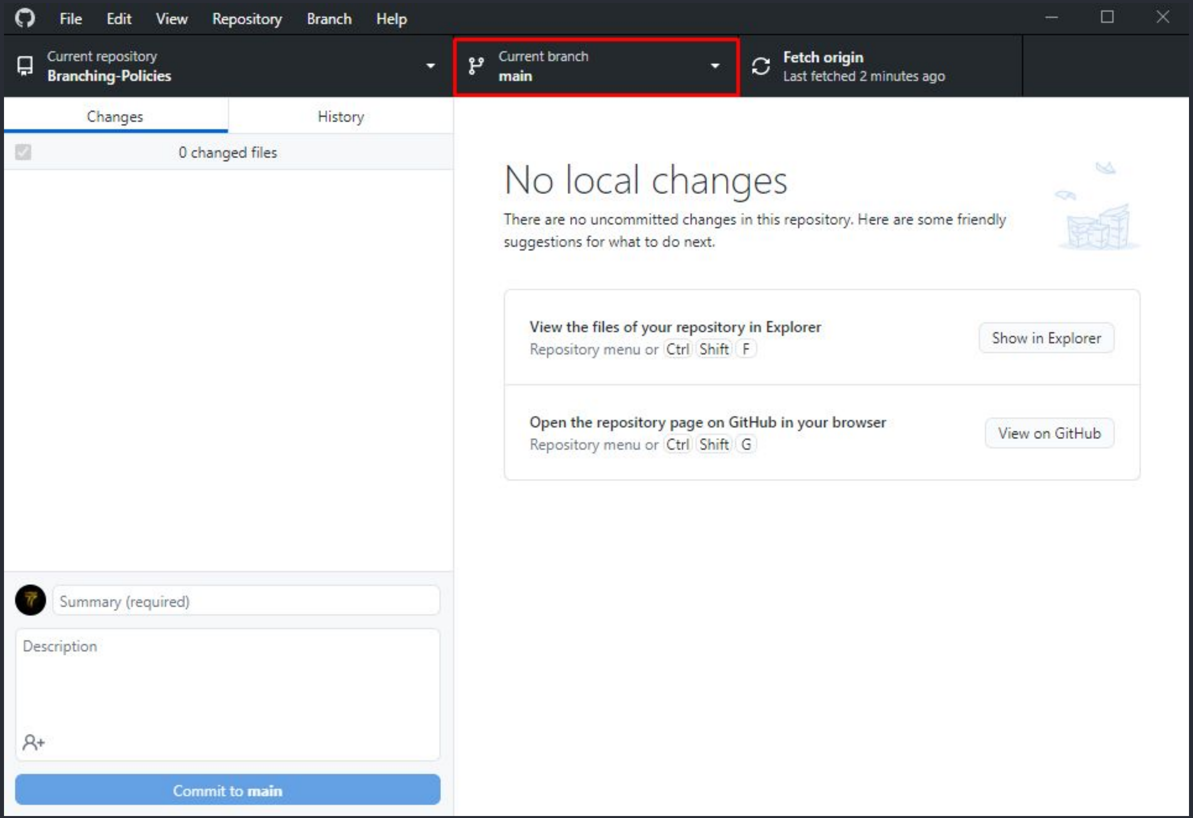
WHEN ?

- Should you do it in any project?
- Never too late
- Every feature deserves a branch

4

HOW ?

HOW ? - Step 1



No local changes

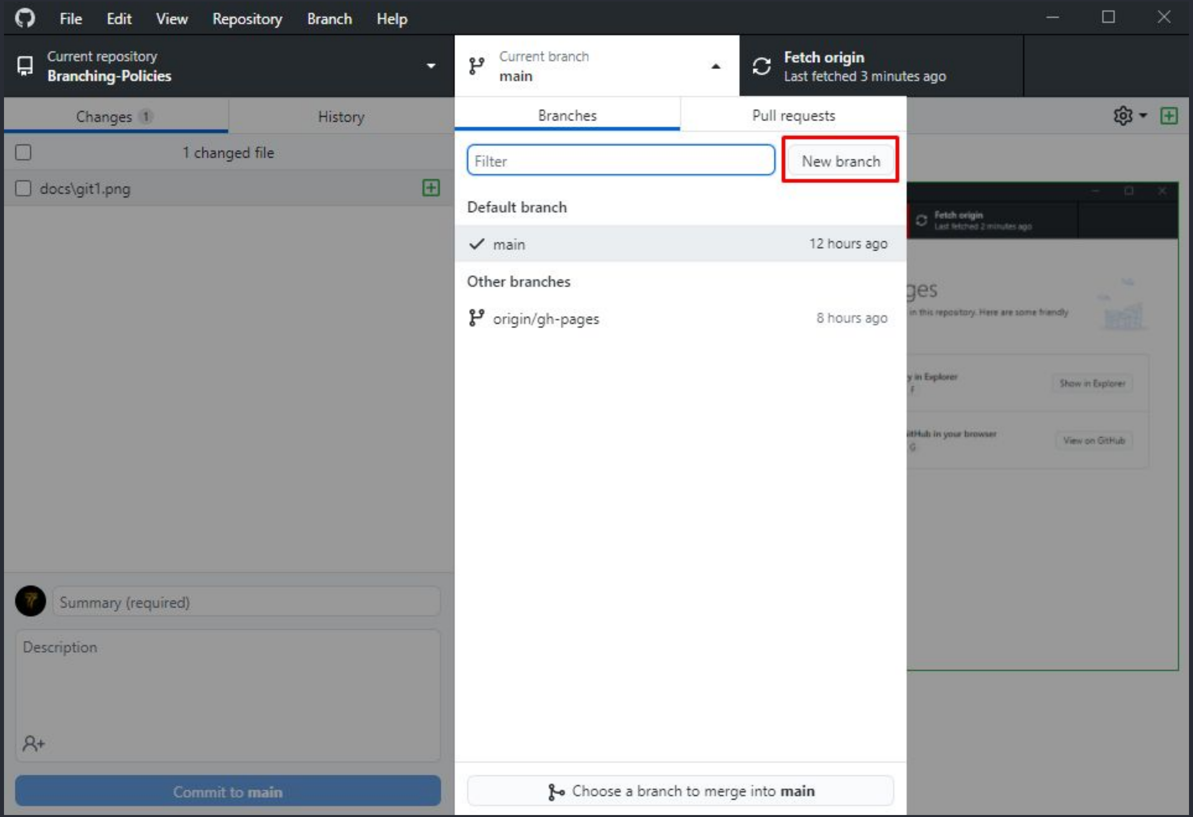
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

View the files of your repository in Explorer
Repository menu or Ctrl Shift F
Show in Explorer

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G
View on GitHub

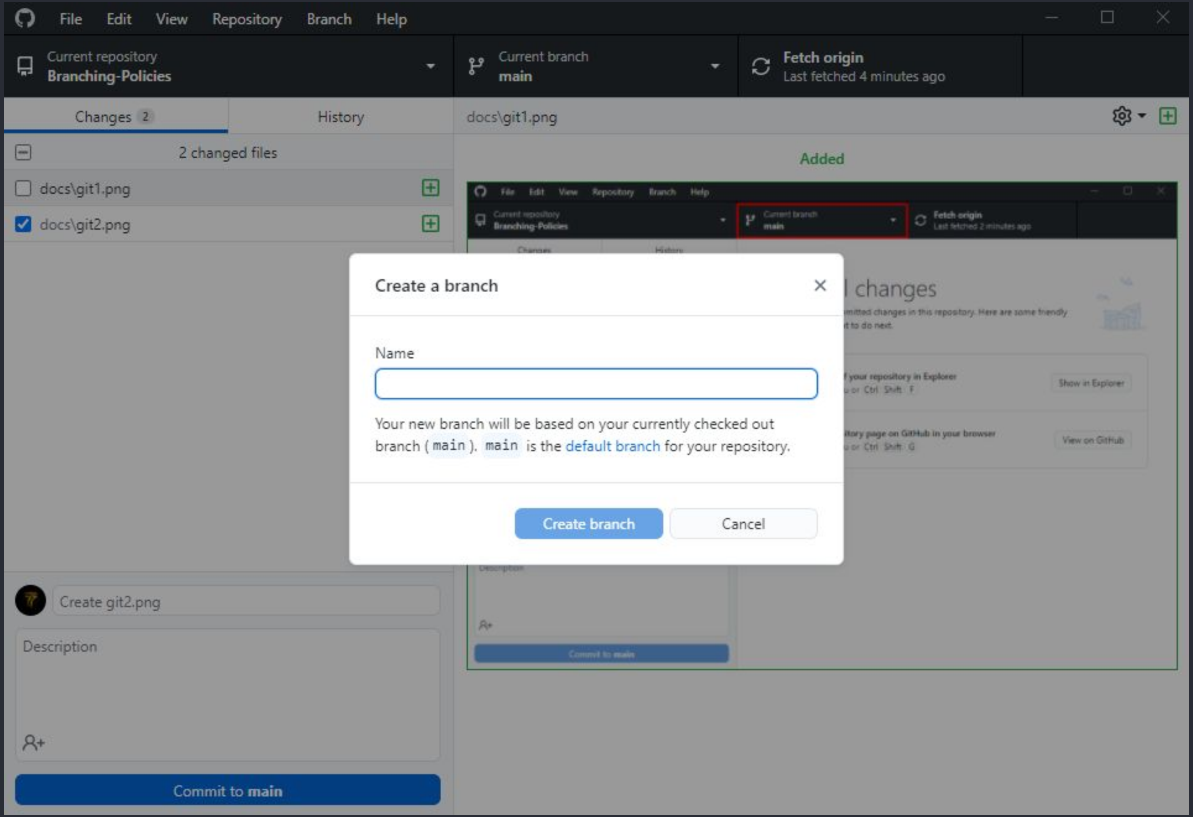
1. Open the branch selector

HOW ? - Step 2



2. Select the "New branch" option

HOW ? - Step 3



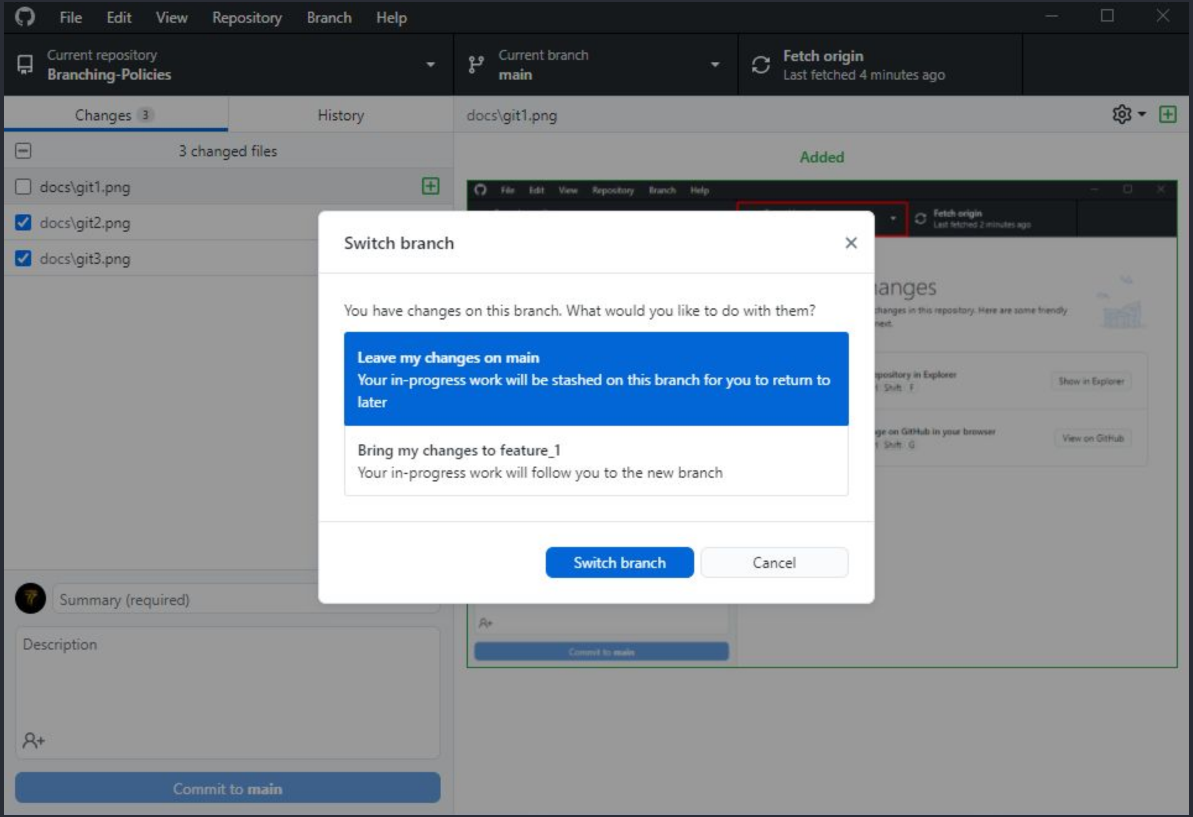
Create a branch

Name

Your new branch will be based on your currently checked out branch (`main`). `main` is the default branch for your repository.

3. Give an appropriate name to the branch

HOW ? - Step 4



Switch branch

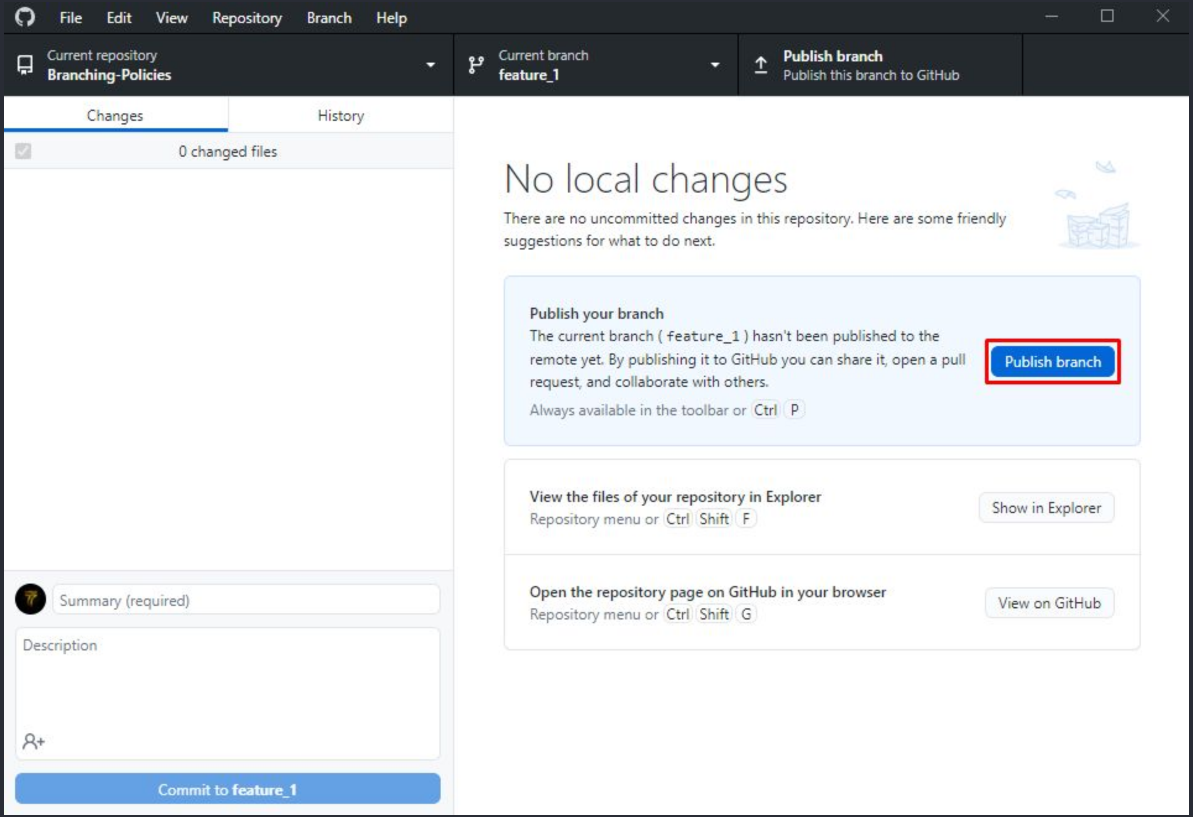
You have changes on this branch. What would you like to do with them?

Leave my changes on main
Your in-progress work will be stashed on this branch for you to return to later

Bring my changes to feature_1
Your in-progress work will follow you to the new branch

4. Switch the branch you want to work in

HOW ? - Step 5



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Publish your branch

The current branch (`feature_1`) hasn't been published to the remote yet. By publishing it to GitHub you can share it, open a pull request, and collaborate with others.

Always available in the toolbar or Ctrl P

View the files of your repository in Explorer

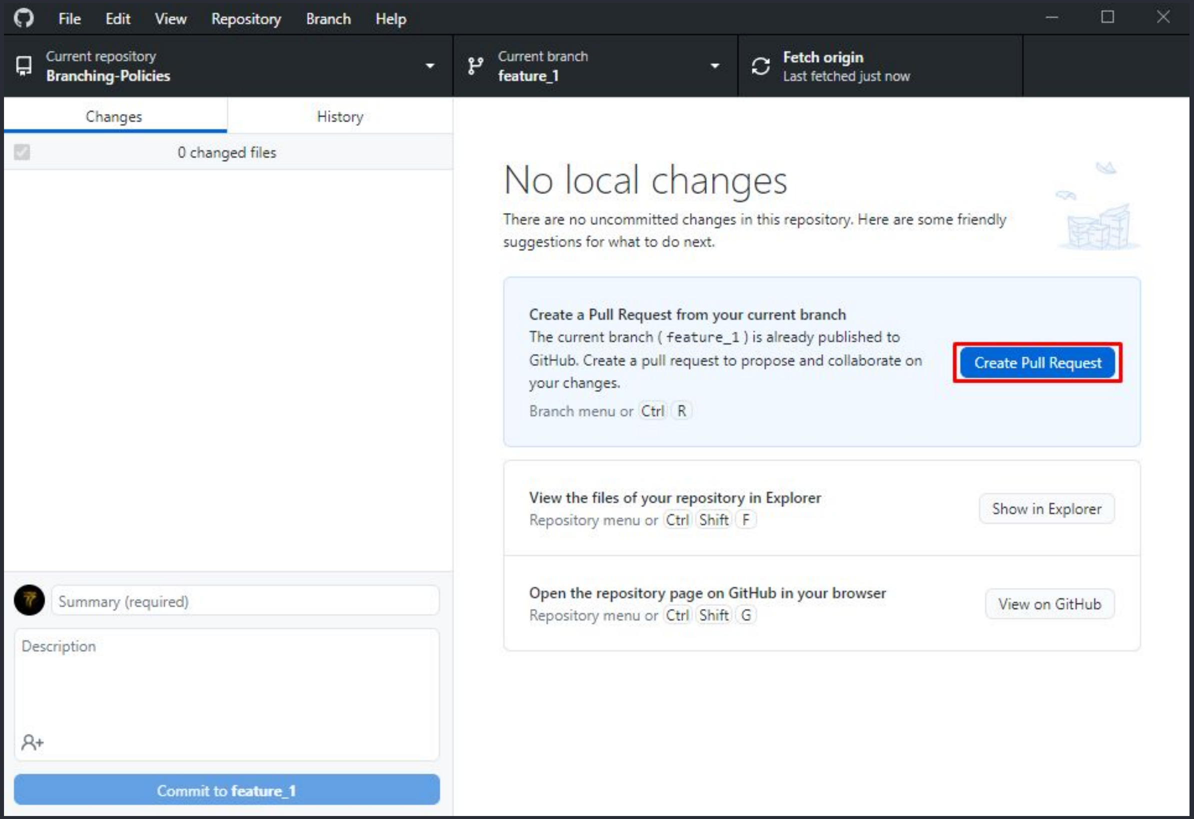
Repository menu or Ctrl Shift F

Open the repository page on GitHub in your browser

Repository menu or Ctrl Shift G

5. Publish the new branch to the repository

HOW ? - Step 6



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Create a Pull Request from your current branch
The current branch (`feature_1`) is already published to GitHub. Create a pull request to propose and collaborate on your changes.
Branch menu or Ctrl R

Create Pull Request

View the files of your repository in Explorer
Repository menu or Ctrl Shift F

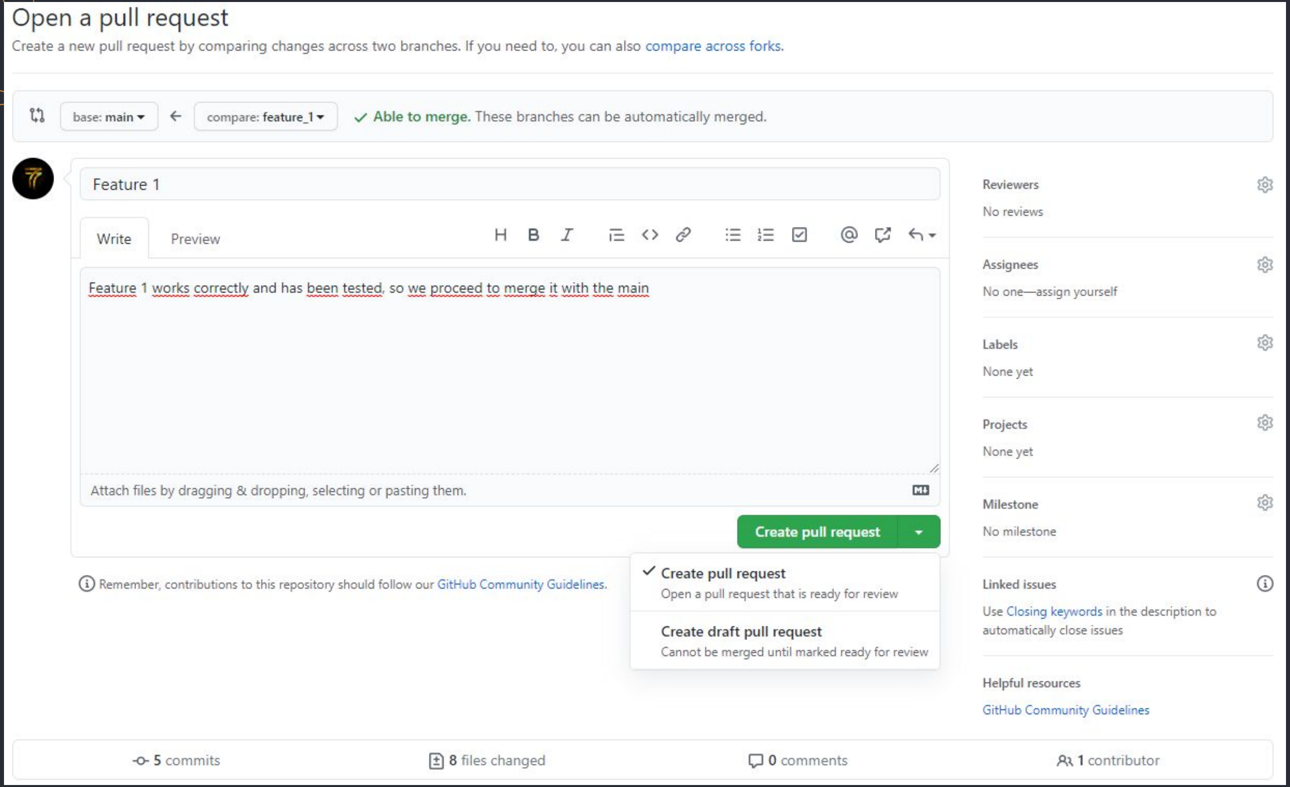
Show in Explorer

Open the repository page on GitHub in your browser
Repository menu or Ctrl Shift G

View on GitHub

6. Create a Pull Request

HOW ? - Step 7



Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.

base: main ← compare: feature_1 ✓ Able to merge. These branches can be automatically merged.

Feature 1

Write Preview

Feature 1 works correctly and has been tested, so we proceed to merge it with the main

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

✓ Create pull request
Open a pull request that is ready for review

Create draft pull request
Cannot be merged until marked ready for review

Remember, contributions to this repository should follow our GitHub Community Guidelines.

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

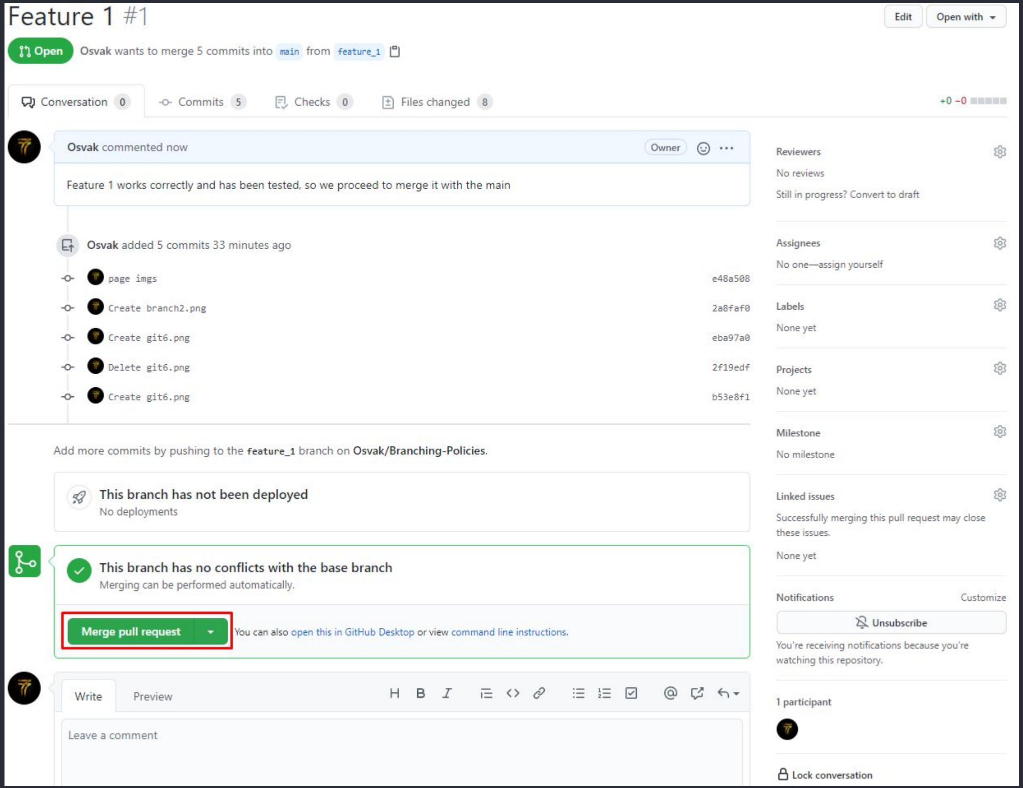
Linked issues
Use Closing keywords in the description to automatically close issues

Helpful resources
GitHub Community Guidelines

5 commits 8 files changed 0 comments 1 contributor

7. Open the Pull Request

HOW ? - Step 8



Feature 1 #1

Edit | Open with

Open | Osvak wants to merge 5 commits into `main` from `feature_1`

Osvak commented now — Owner

Feature 1 works correctly and has been tested, so we proceed to merge it with the main

Osvak added 5 commits 33 minutes ago

```
page imgs           e48a508
Create branch2.png  2a8faf0
Create git6.png     eba97a0
Delete git6.png     2f19edf
Create git6.png     b53e8f1
```

Add more commits by pushing to the `feature_1` branch on **Osvak/Branching-Policies**.

This branch has not been deployed
No deployments

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request — You can also open this in GitHub Desktop or view command line instructions.

Reviewers: No reviews
Still in progress? Convert to draft

Assignees: No one—assign yourself

Labels: None yet

Projects: None yet

Milestone: No milestone

Linked issues: Successfully merging this pull request may close these issues.
None yet

Notifications — Customize
Unsubscribe
You're receiving notifications because you're watching this repository.

1 participant

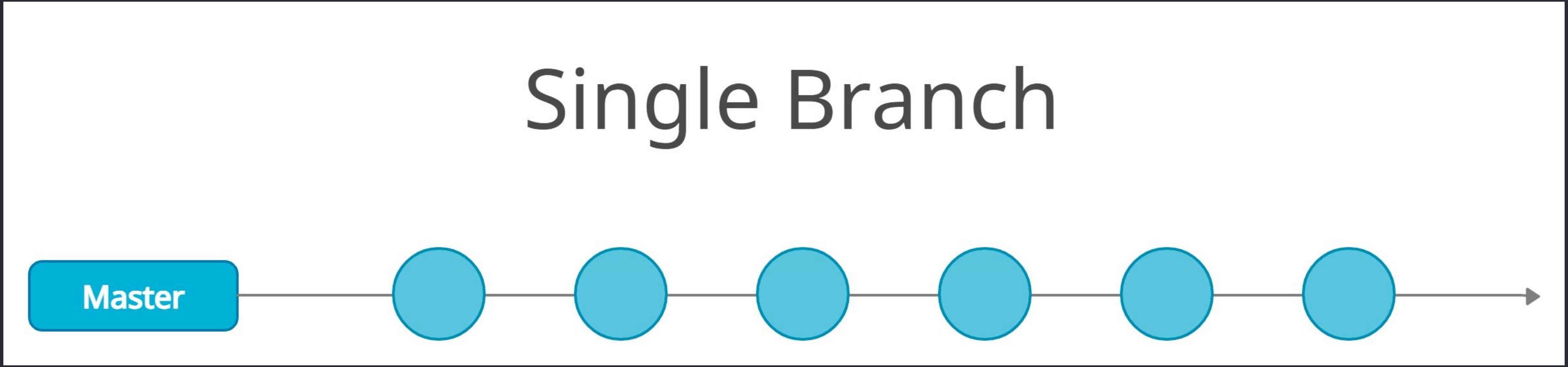
Lock conversation

8. Merge the Pull Request to the Branch

5

MOST USED FLOW MODELS

FLOW MODELS - Single Branch

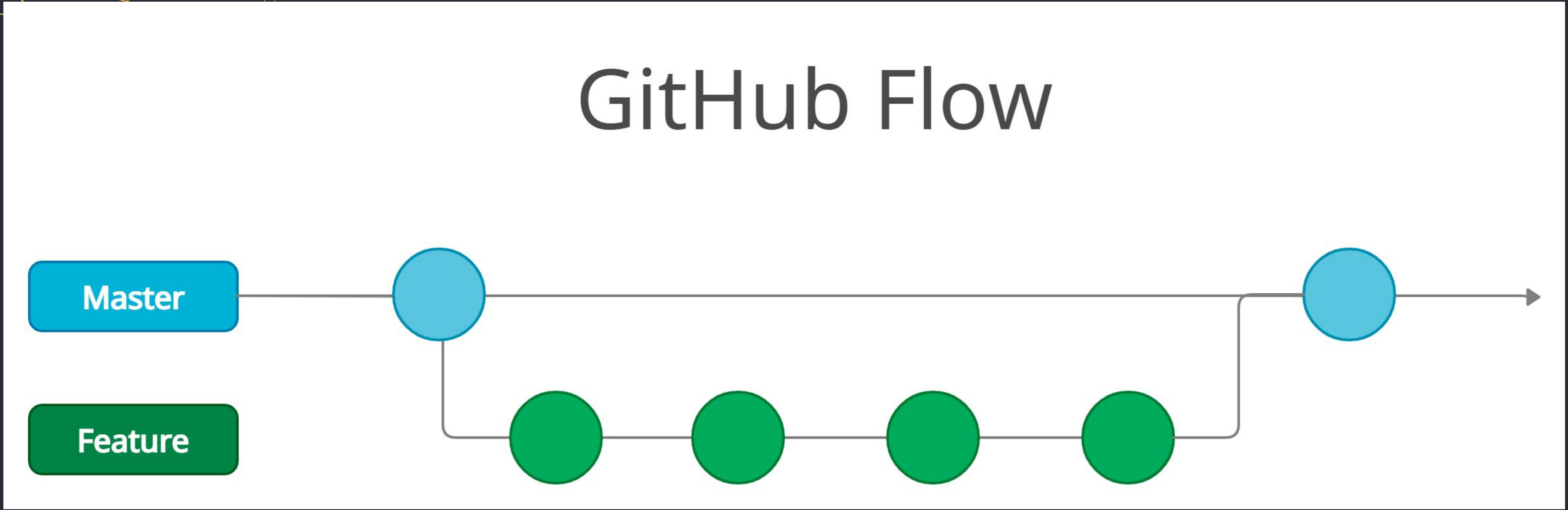


Single Branch - Features

- Simple
- Agile
- Teams need to trust each other
- All the commits are based on the use of Feature Flags

*Feature Flags that allow you to enable or disable a new feature, often used to avoid conflicts when merging.

FLOW MODELS - GitHub Flow

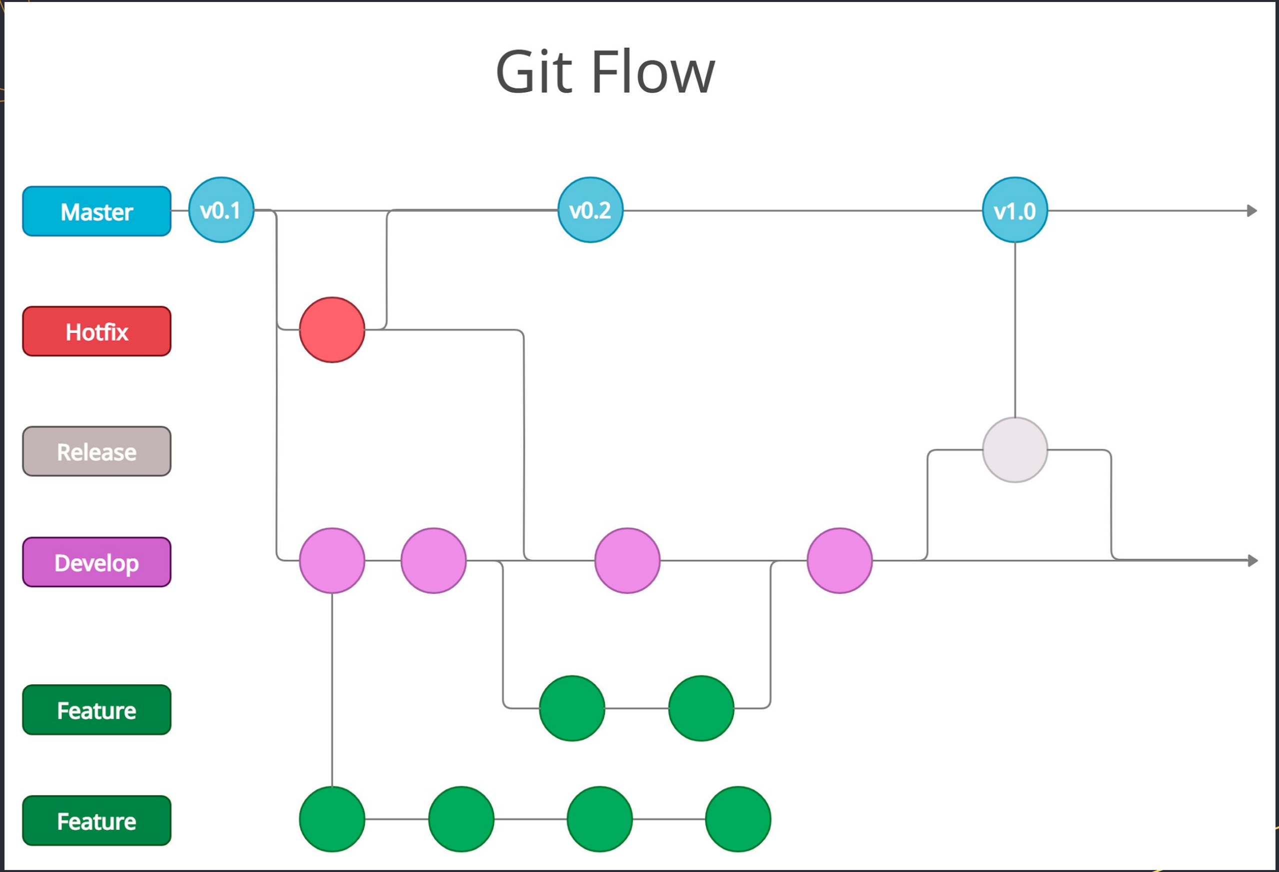


GitHub Flow - Features

- Everything is done in the Feature branch
- Master is always deployable and acts like a safety measure
- Works very well with Continuous Deployment
- Clean Commit History

*Continuous Deployment is a software release process that uses automated testing.

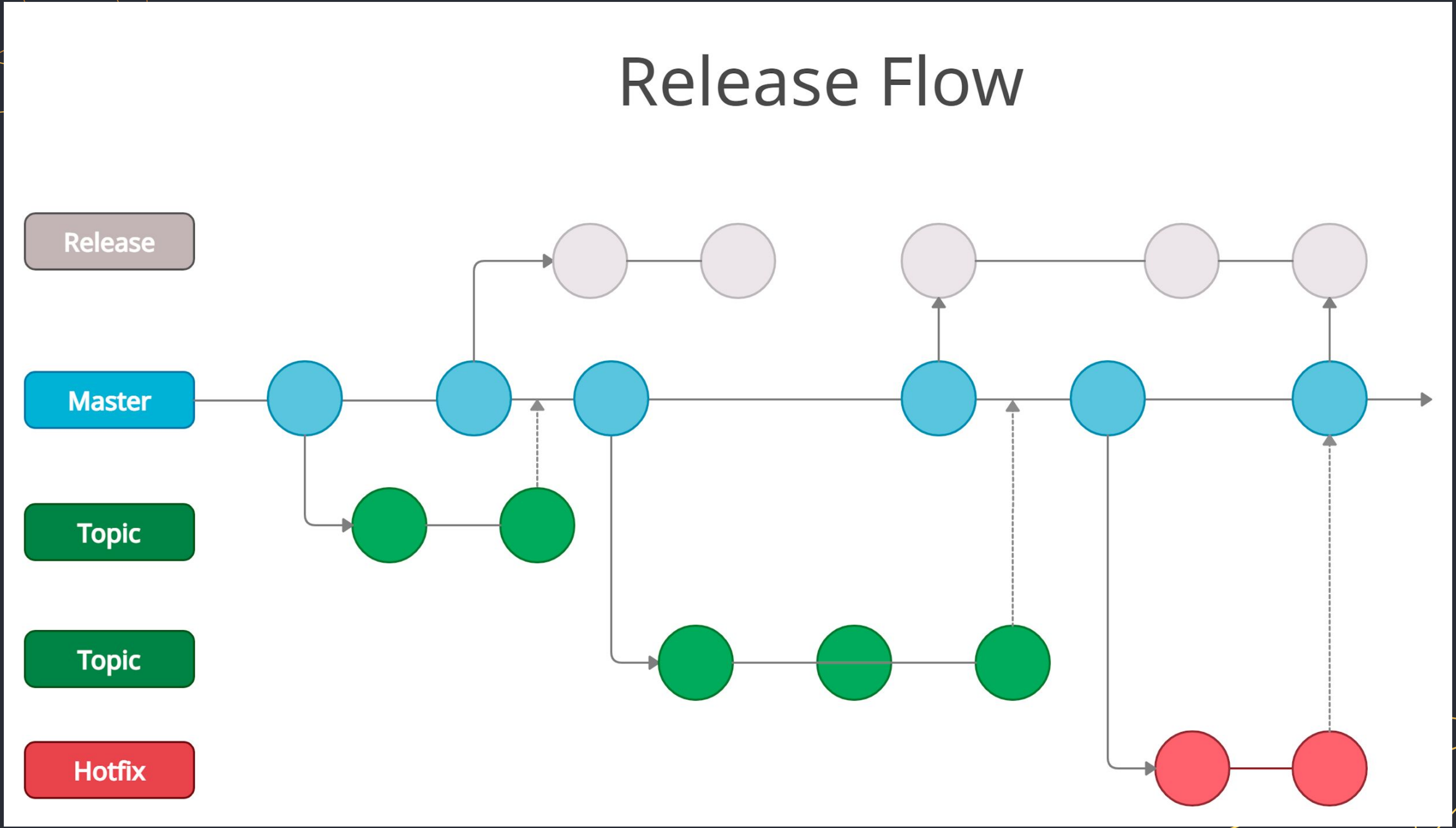
FLOW MODELS - Git Flow



Git Flow - Features

- Many branches (Master + Develop + Feature Branch + Release Branch + Hotfix Branch)
- Complex
- Great for release-based workflow
- The most famous model
- Not recommended for small projects
- Messy Commit History

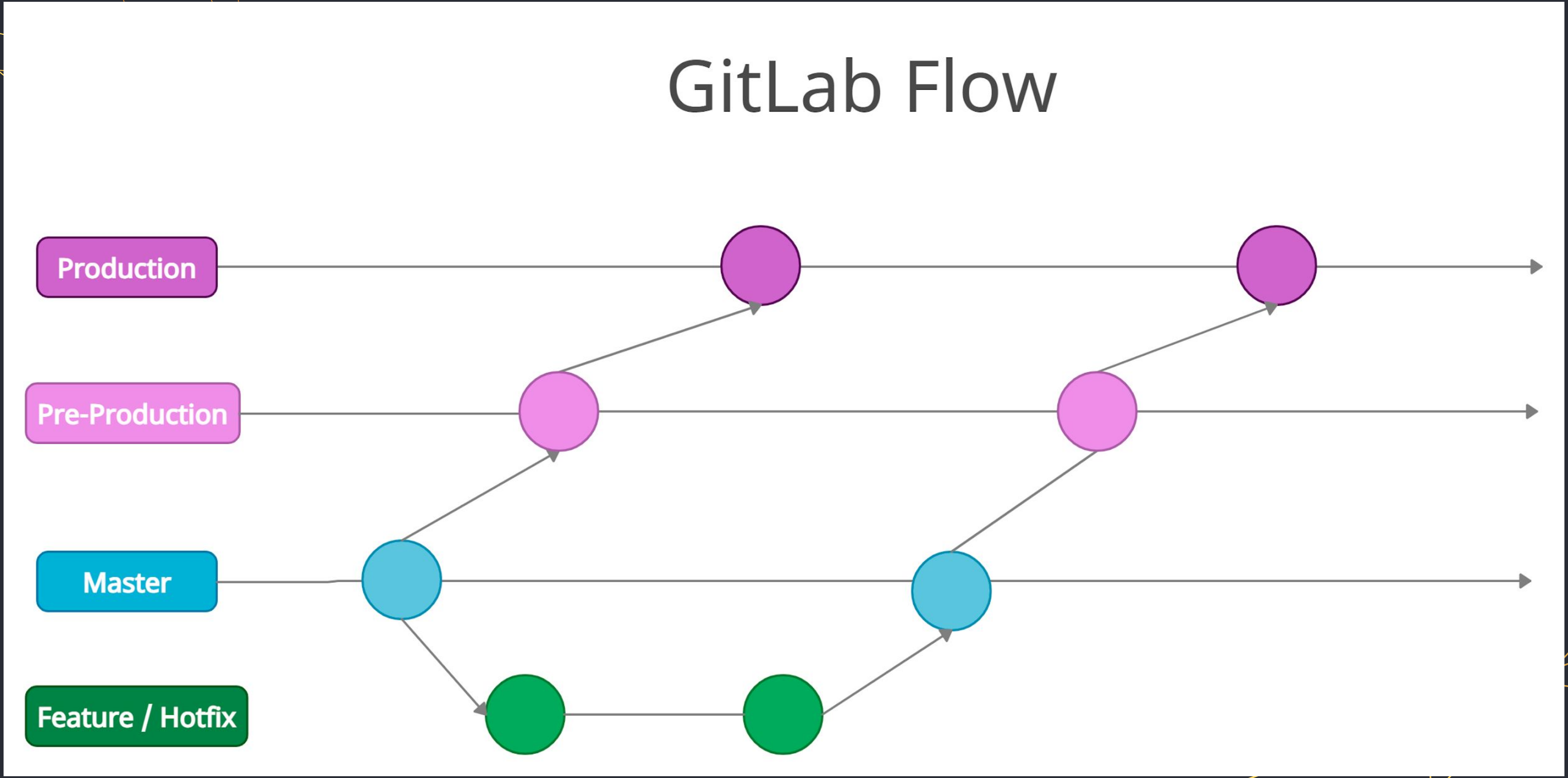
FLOW MODELS - Release Flow



Release Flow - Features

- System developed by Microsoft
- Topics are like mini features
- Releases every 3 weeks
- Model used to manage massive development teams

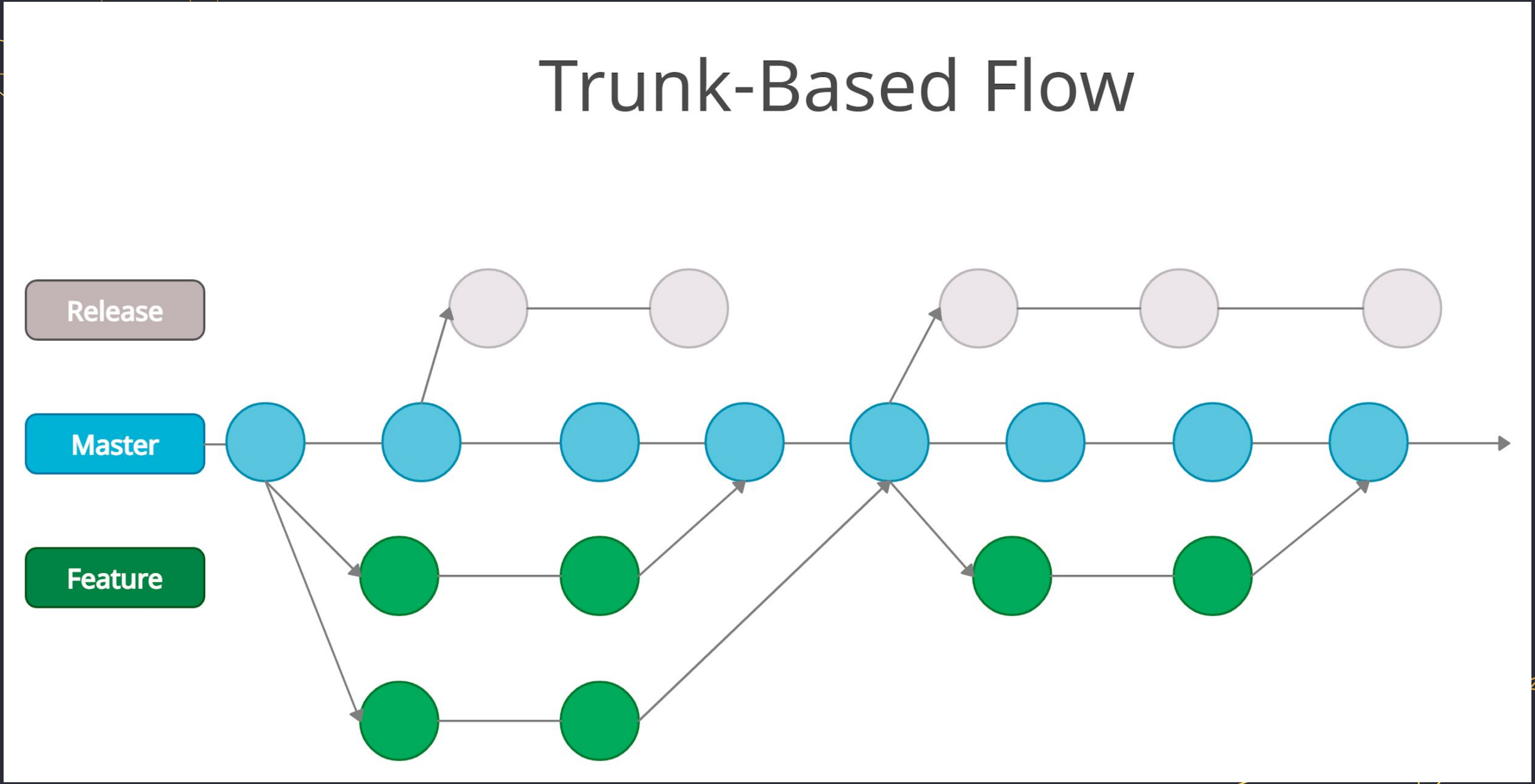
FLOW MODELS - GitLab Flow



GitLab Flow - Features

- Similar to GitHub Flow but more complex
- Not as Agile as other methods
- When a feature arrives to production its more reliable
- Forbidden to commit in master directly
- Recommended for projects where you can't allow yourself to fail

FLOW MODELS - Trunk-Based Flow



Trunk-Based Flow - Features

- Similar to GitHub Flow
- Master branch can develop with the use of Feature Flags
- Can have Release branches
- Master is always deployable

6

CONCLUSION

CONCLUSIONS

- Large number of Pros VS the small amount of Cons
- Try to always use branching, but don’t feel forced to use it
- Create a branch in your repository and see what happens
- Test the different models and find the one suitable for your project

RECOMMENDATION

- **Single Branch:** You probably already used, but try extracting its true potential. Focus on Feature Flags, so everyone can work at the same time.

- **GitHub Flow:** It's easier to manage conflicts than a Single Branch flow. Good model to start branching. Not needing Feature Flags.

- **Trunk-Based Development:** Same advantages from GitHub Flow.
It's a matter of preference, in GitHub you deploy from the Feature branch, while in Trunk-Based you first commit to Master and then do the deployment.

Time to create your first branch

- Try creating a branch in your repository without looking the step by step guide
- Commit something to the branch and then merge it to main by using a pull request
- If you get lost visit https://osvak.github.io/Branching-Policies/

Recommendation: do it in a repository that you don't use, or create a new one to avoid any problems

RESOURCES

LINKS

- GitHub Repository
- Topic Web Page

REFERENCES

- Git Flow
- Main branch information
- Explanation of branching use
- Concept explanation
- Branch models (Spanish)
- Creative Branching Models for Multiple Release Streams
- Creately

THANKS!

DO YOU HAVE ANY QUESTION?

oscarcanales2001@gmail.com