Session 3.5

Git For Version Control System

Git Workflow

Purwadhika
Startup and Coding School

What is a successful Git workflow?

There are so many workflows in case of collaboration using git & GitHub, but in this chapter we will only discus two of them

- Does this workflow scale with team size?
- Is it easy to undo mistakes and errors with this workflow?
- Does this workflow impose any new unnecessary cognitive overhead to the team?

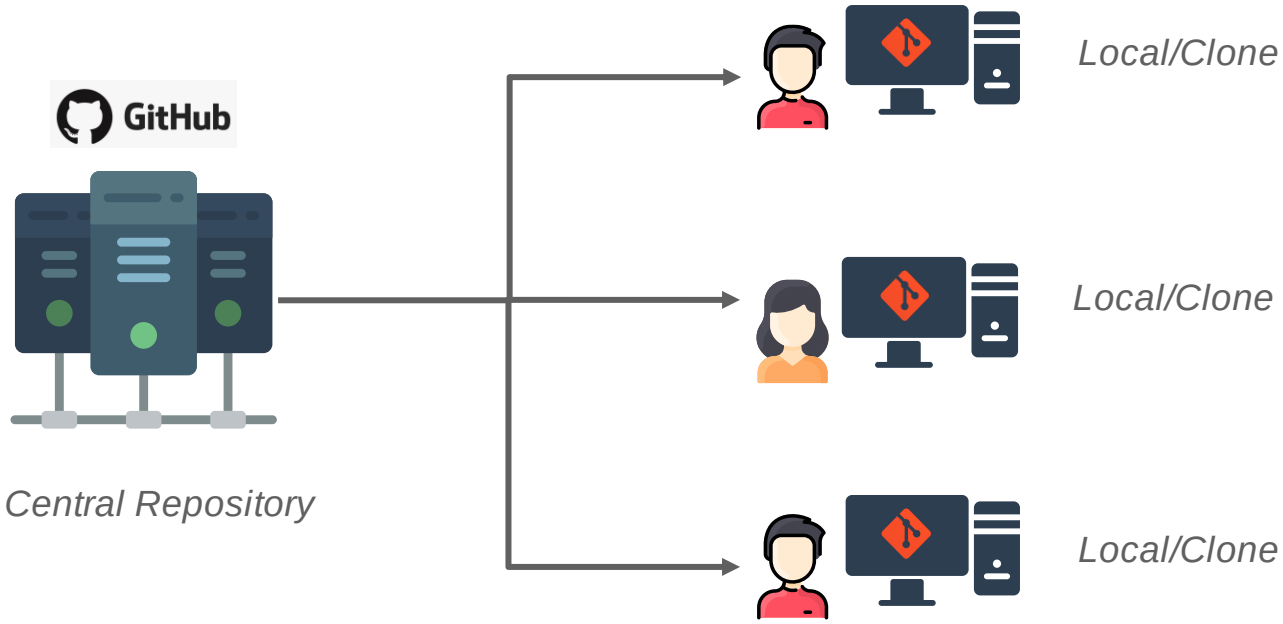
Git Workflow

There are so many workflows in case of collaboration using git & GitHub. In this chapter, we will only discuss some of them that commonly used by other developers in teams

- Centralize Workflow
- Feature Branch Workflow
- Gitflow & GitHub Workflow
- Forking Workflow

Centralize Workflow

This workflow is great for teams transitioning from SVN, this workflow use a central repository to serve as the single point-of-entry for all changes that happened in the projects. The central repository can be hosted through 3rd party git hosting services like Bitbucket Cloud or GitHub

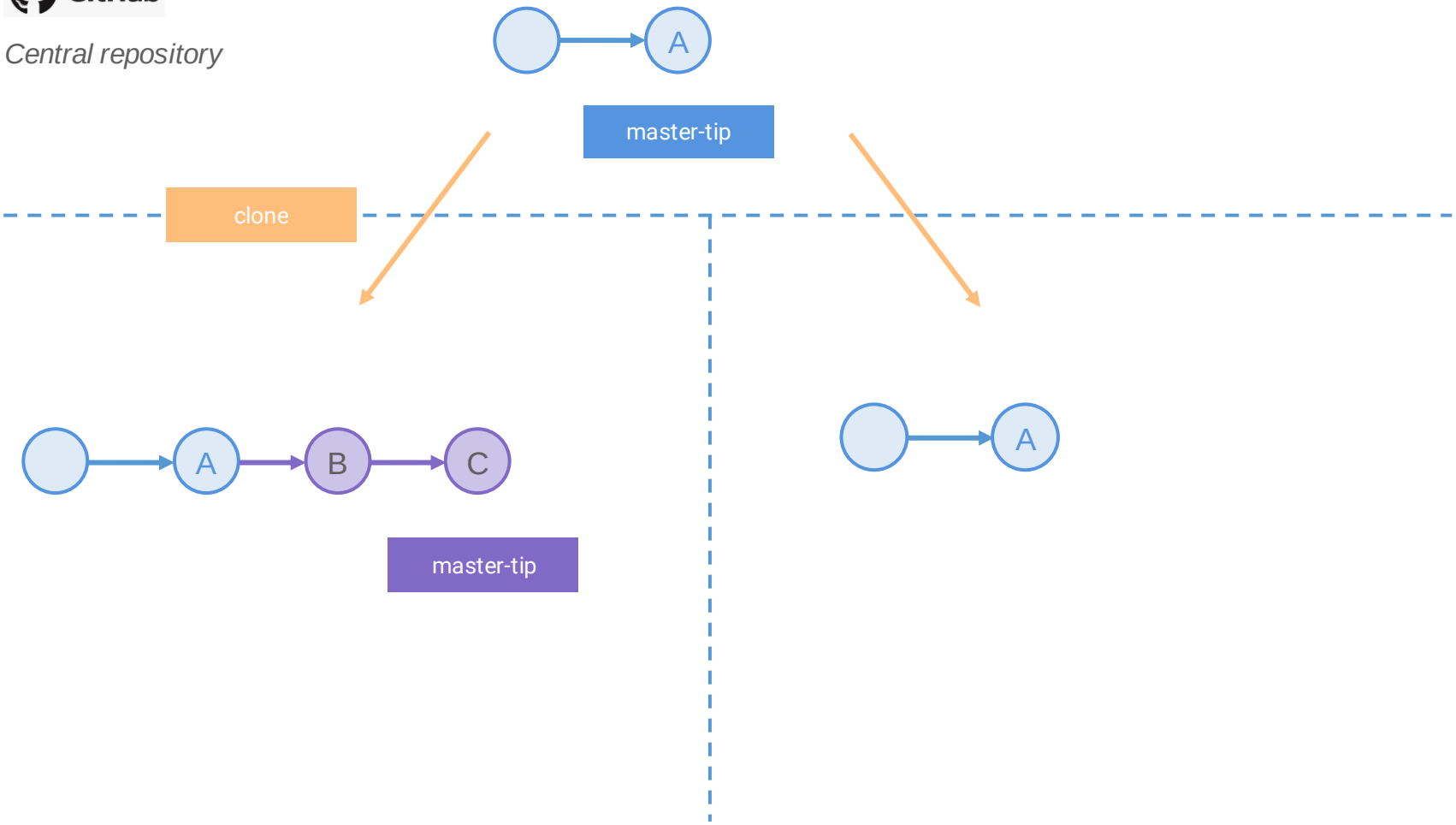


Purwadhika
Startup and Coding School

How it work?

GitHub

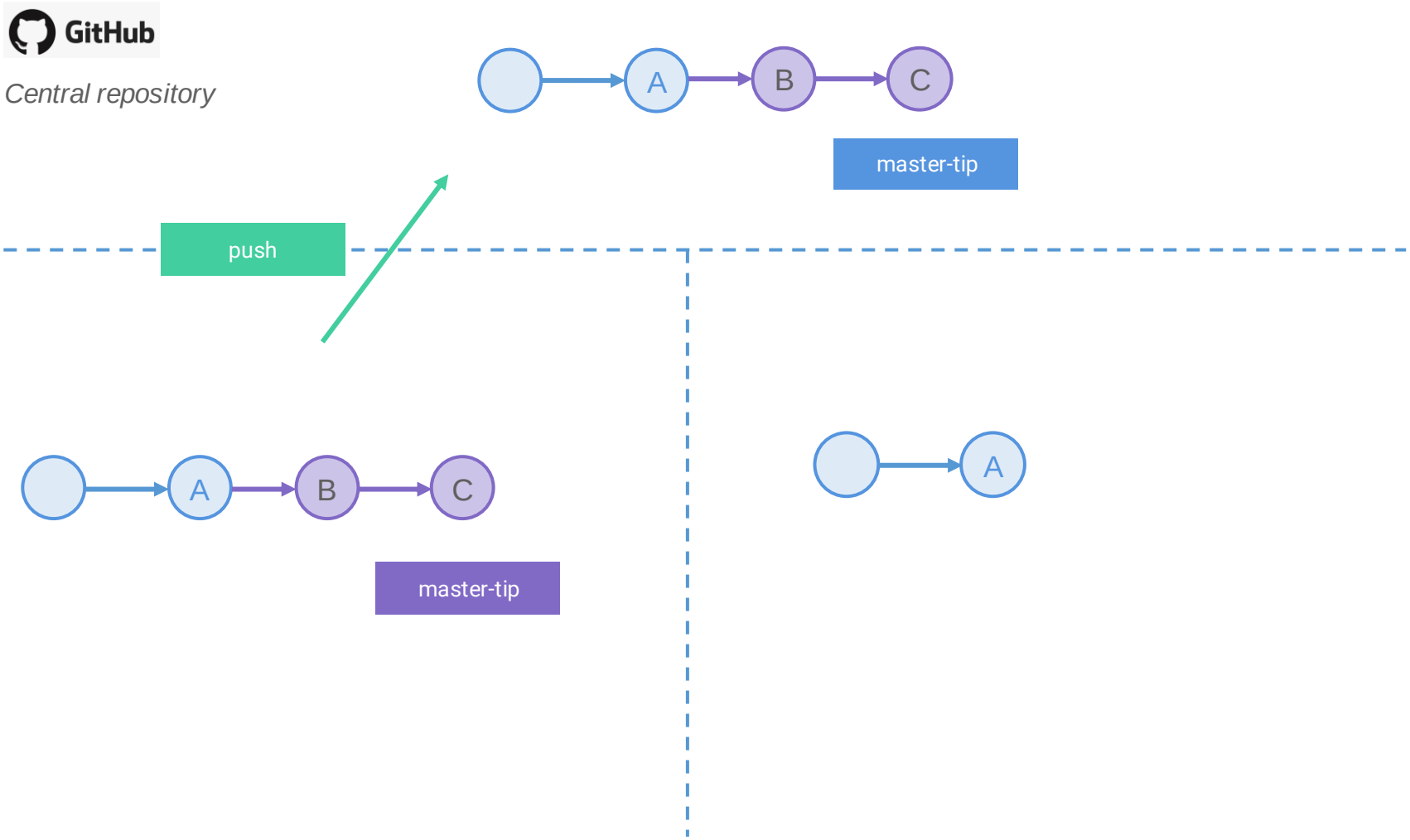
Central repository



Local repository

Purwadhika
Startup and Coding School

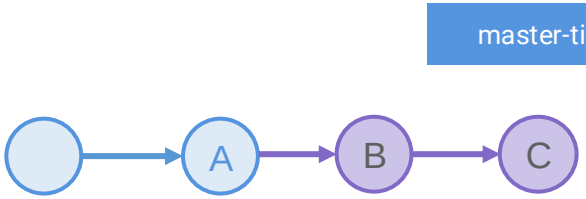
How it work?

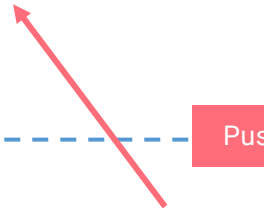


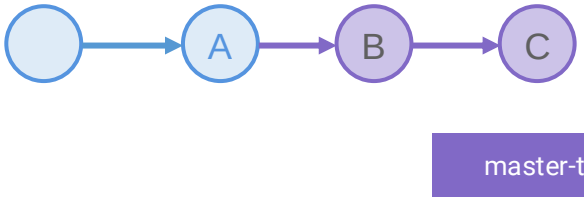
Purwadhika
Startup and Coding School

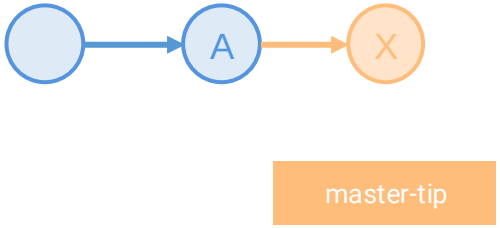
How it work?

Central repository







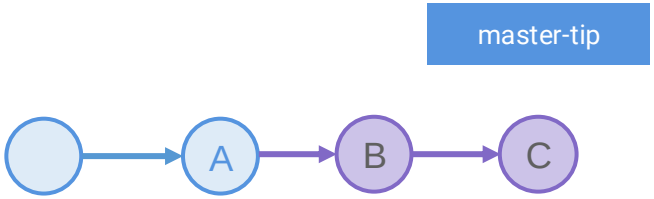


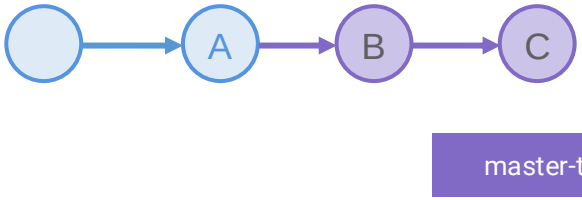
Local repository

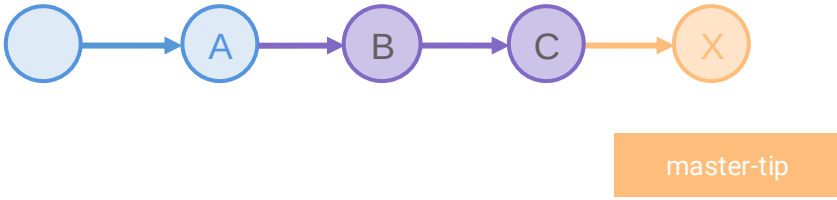
Purwadhika
Startup and Coding School

How it work?

Central repository







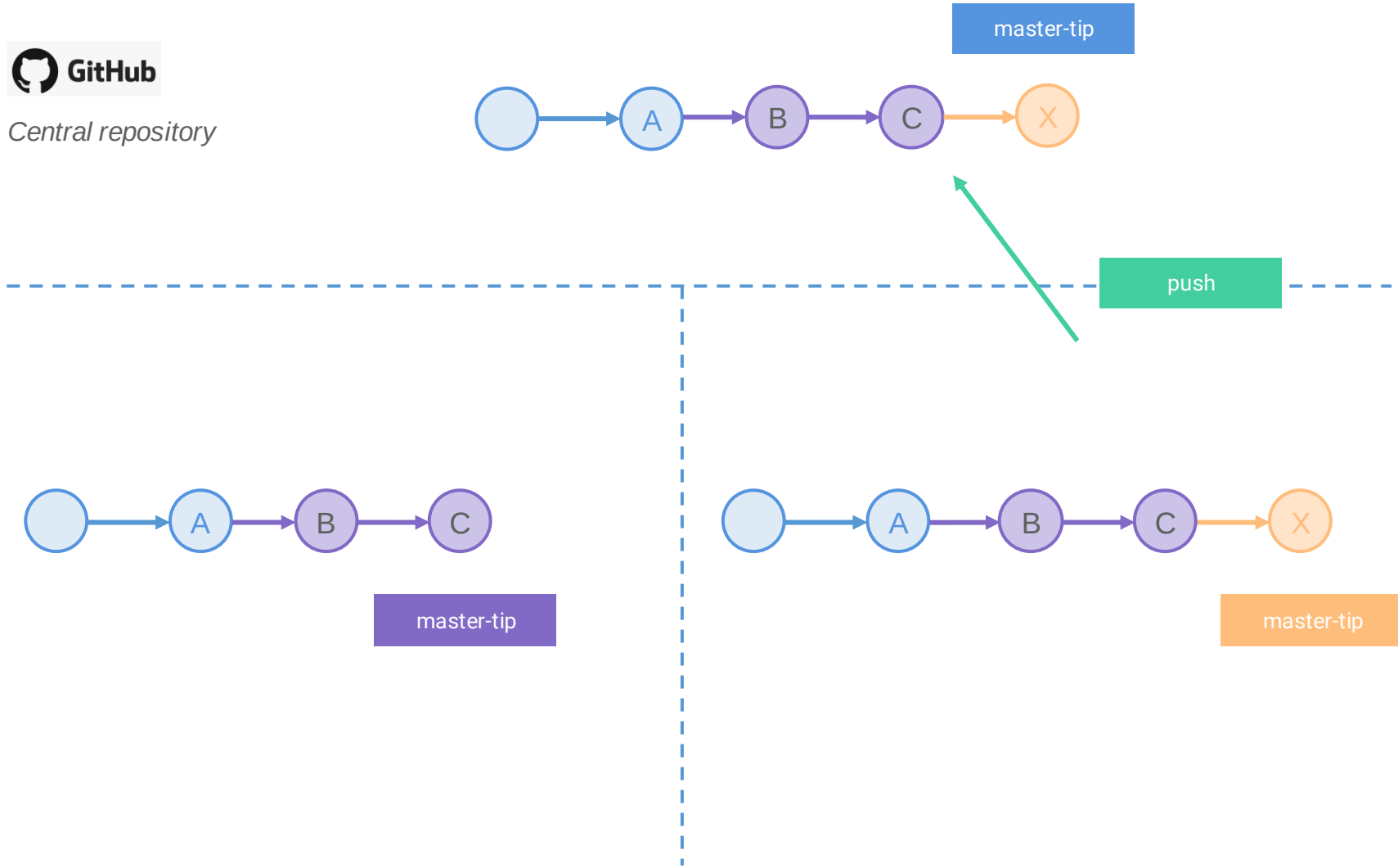
Local repository

Purwadhika
Startup and Coding School

How it work?

GitHub

Central repository

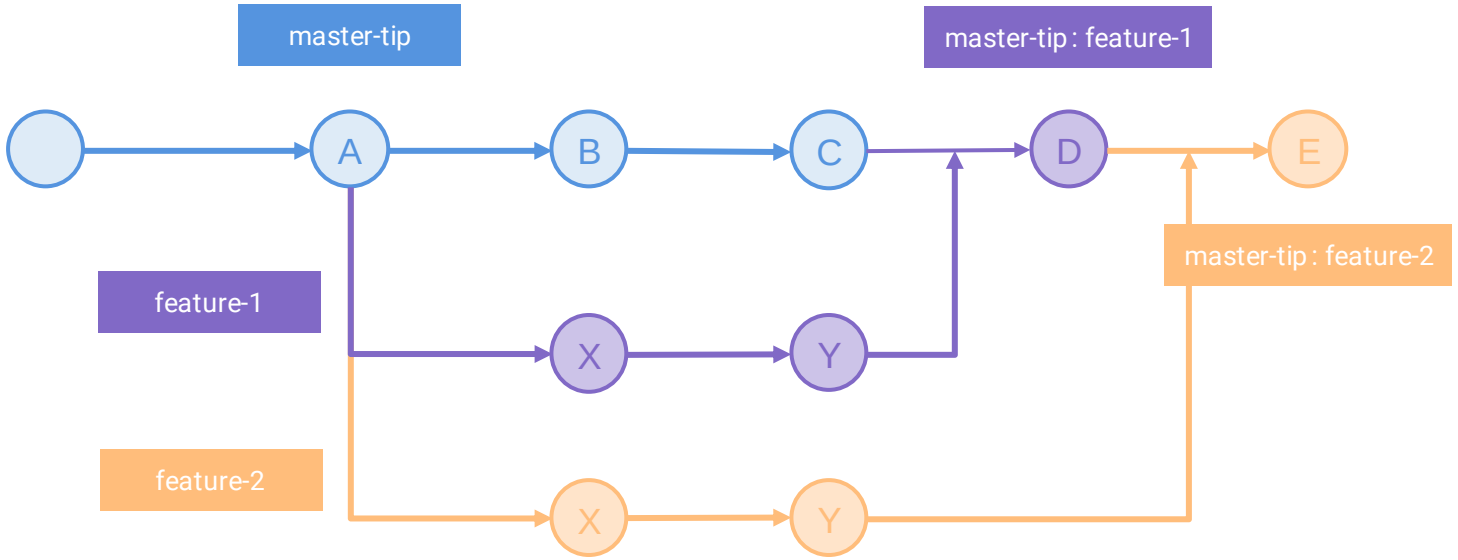


Local repository

Purwadhika
Startup and Coding School

Feature Branch Workflow

The main idea of this workflow is each feature development must be created in separated branch. If the feature is ready, developer will merge their feature branch to master branch. This mean that multiple developer can work on a particular feature without disturbing the main code base (master branch) and the master branch will never contain broken code, which is a huge advantage for continuous integration development

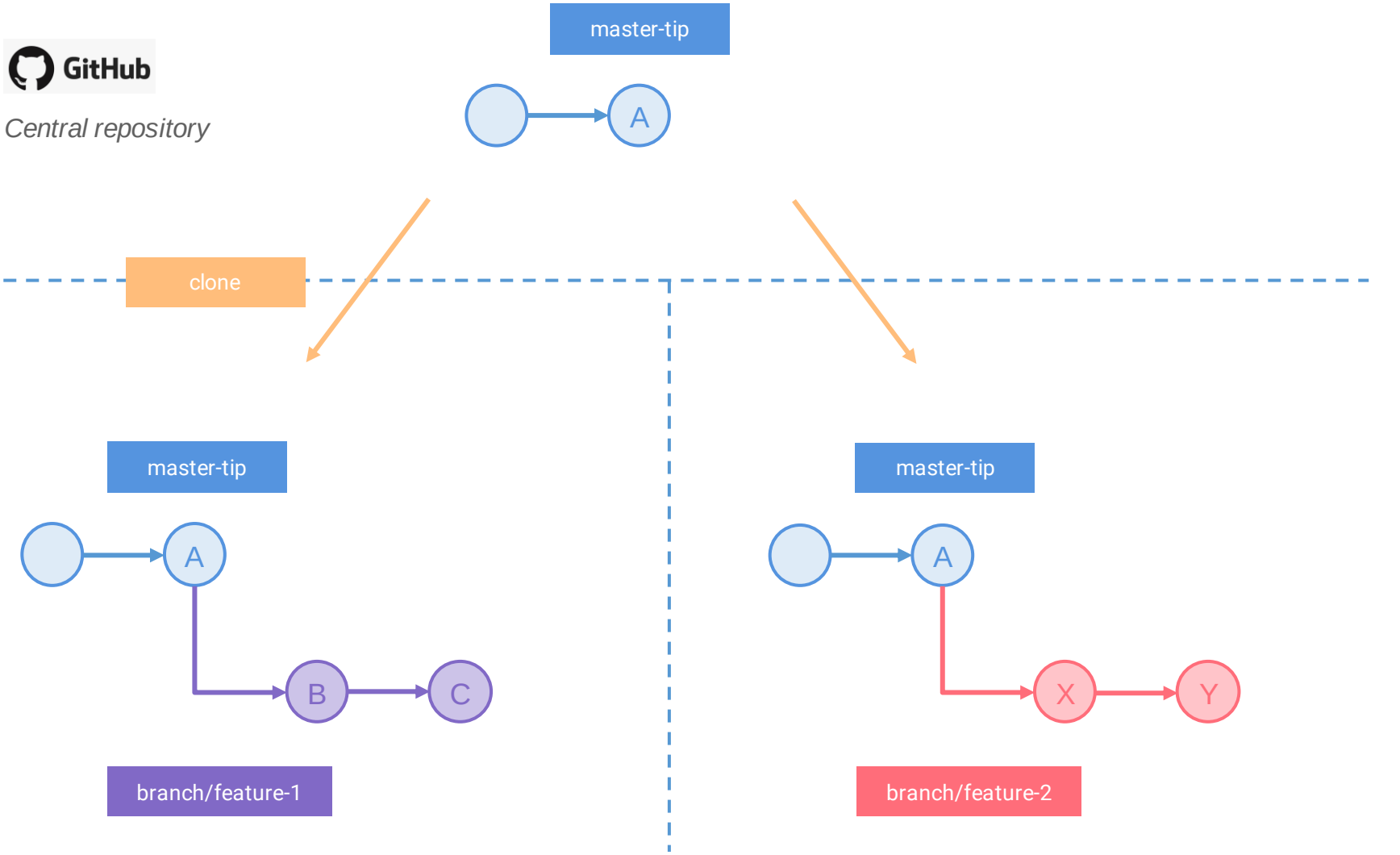


Purwadhika
Startup and Coding School

How it work?

GitHub

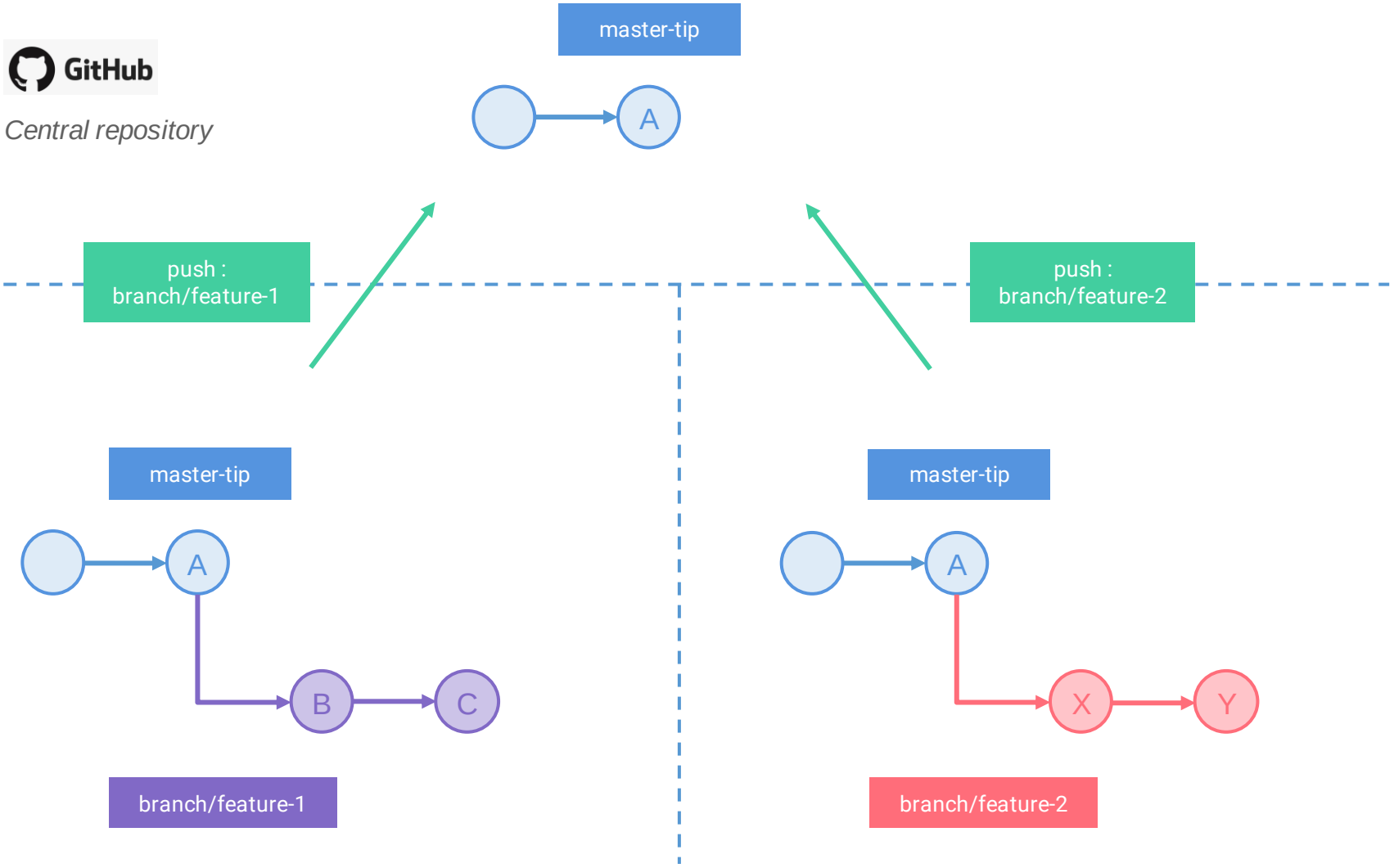
Central repository



Local repository

Purwadhika
Startup and Coding School

How it work?

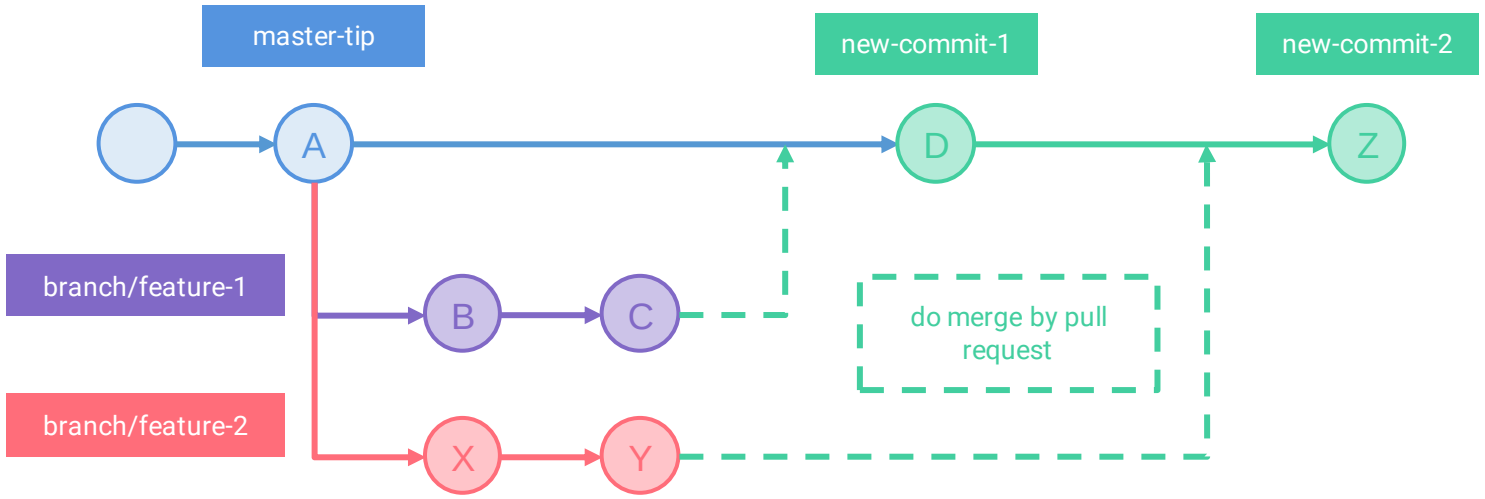


Purwadhika
Startup and Coding School

How it work?

GitHub

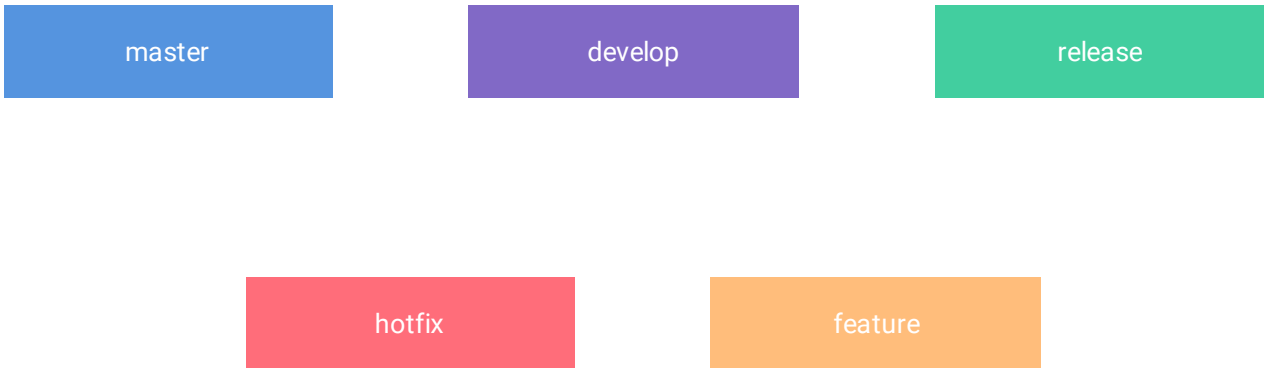
Central repository



Purwadhika
Startup and Coding School

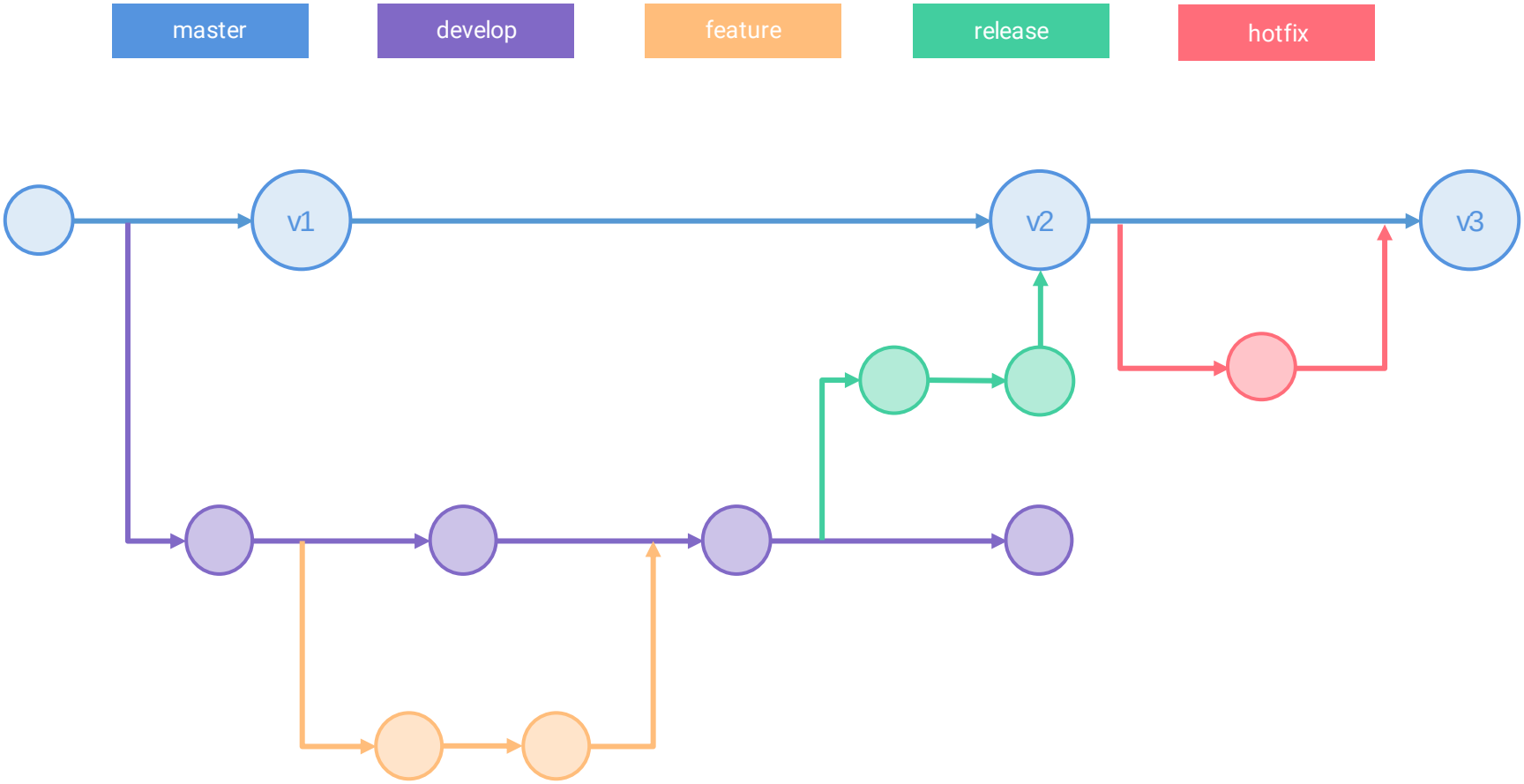
Gitflow Workflow

it a branch-based workflow, mean that each feature will be develop in a separated branch by each developer with some branch rule to make the workflow more effective. In Gitflow there five branch category, master, develop, features, release, and hotfix branch. Each individual branch has their own function and rules.



Purwadhika
Startup and Coding School

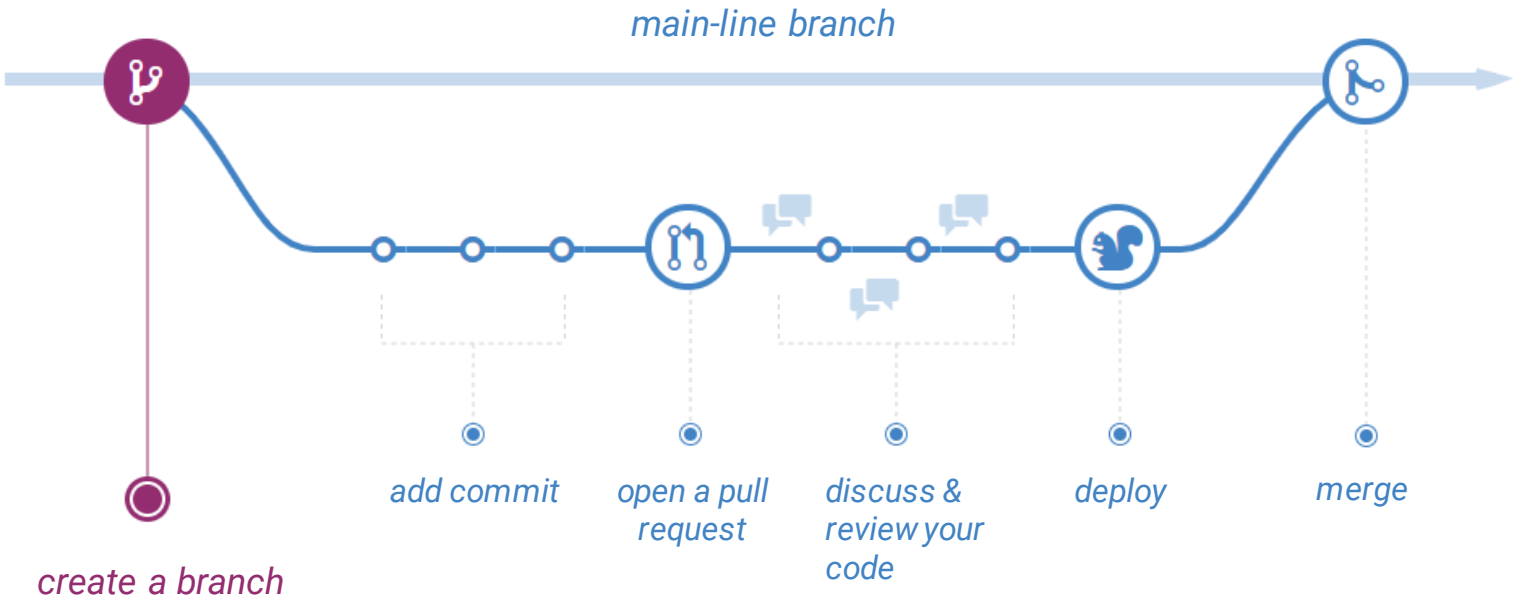
How it work?



Purwadhika
Startup and Coding School

GitHub Workflow

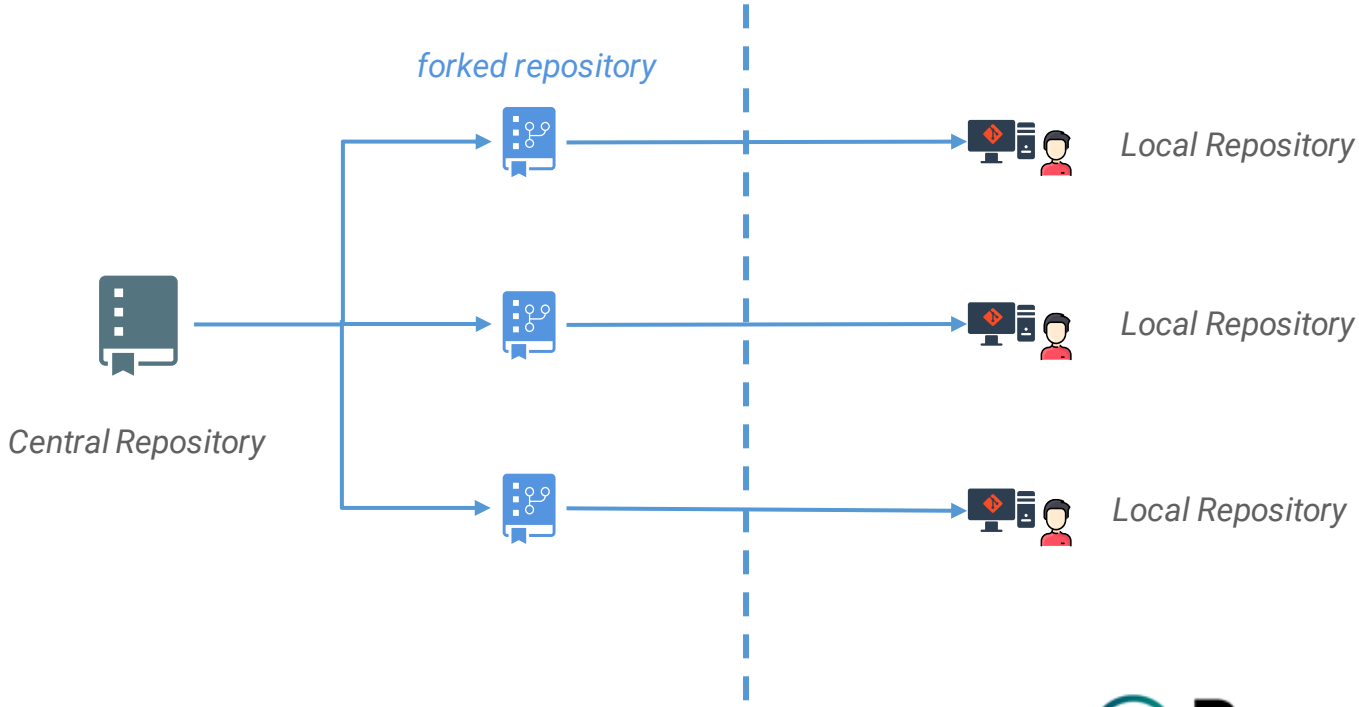
It similar with gitflow except it doesn't have "release" branch. Deployment happens frequently, almost every day, even multiples times a day. This workflow is good for teams and projects focused primarily on continuous delivery. There's only one rule: anything in the main branch is always deployable.



Purwadhika
Startup and Coding School

Forking Workflow

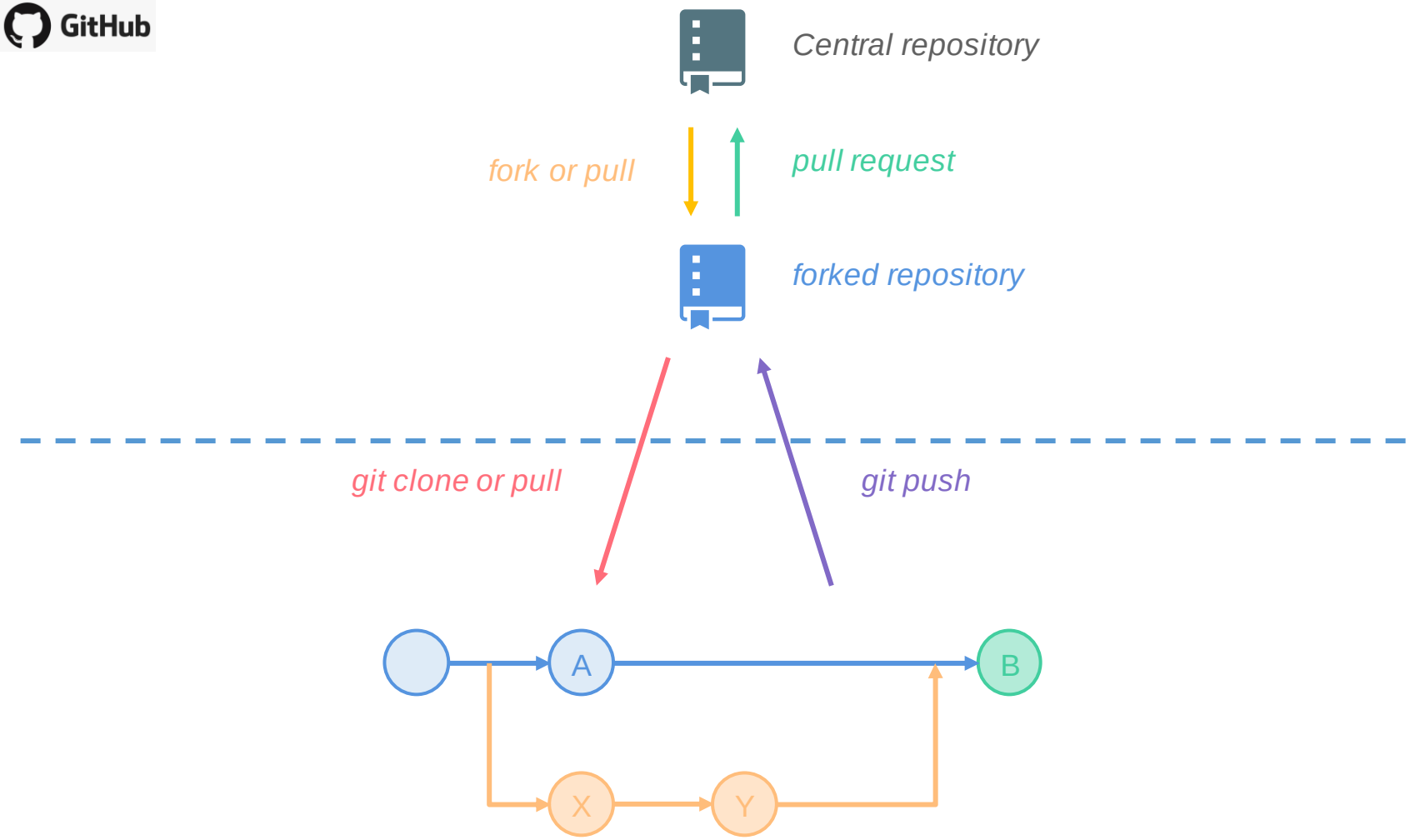
It fundamentally different with others workflow. Instead of using single-side repository act as the "central repository" for the codebase, it gives every developer their own repository that fork from main repository. So, contributions can be integrated without the need for everybody to push to a single central repository. The result is a distributed workflow that provides a flexible way for large, organic teams (including untrusted third-parties) to collaborate securely.



Purwadhika
Startup and Coding School

How it work?

GitHub



Local Repository

Purwadhika
Startup and Coding School

Session 3

Git For Version Control System

Git Log

Purwadhika
Startup and Coding School

- Create & show all available Branches

```
$ git branch --list
```

```
$ git branch bug-fixed
```

- Create a remote branch in GitHub

```
$ git branch new-branch
$ git remote add remote-repo http://github.com/....
$ git push remote-repo new-branch
```

Purwadhika
Startup and Coding School

Git Checkout

The git checkout command lets you navigate between the branches created by git branch.

```
$ git checkout new-branch
```

- Switching & create a new branch

```
$ git checkout -b new-feature
```

- Checkout from remote branch

```
// get all git content from remote repository using fetch
$ git fetch --all

// swicth branch from remote repository using checkout
$ git checkout branch-name

// Additionally you can checkout a new local branch and reset it to the remote branches last
commit.
$ git checkout -b  git reset --hard origin/
```

Purwadhika
Startup and Coding School