clearvision

Module 7

Collaboration Strategies

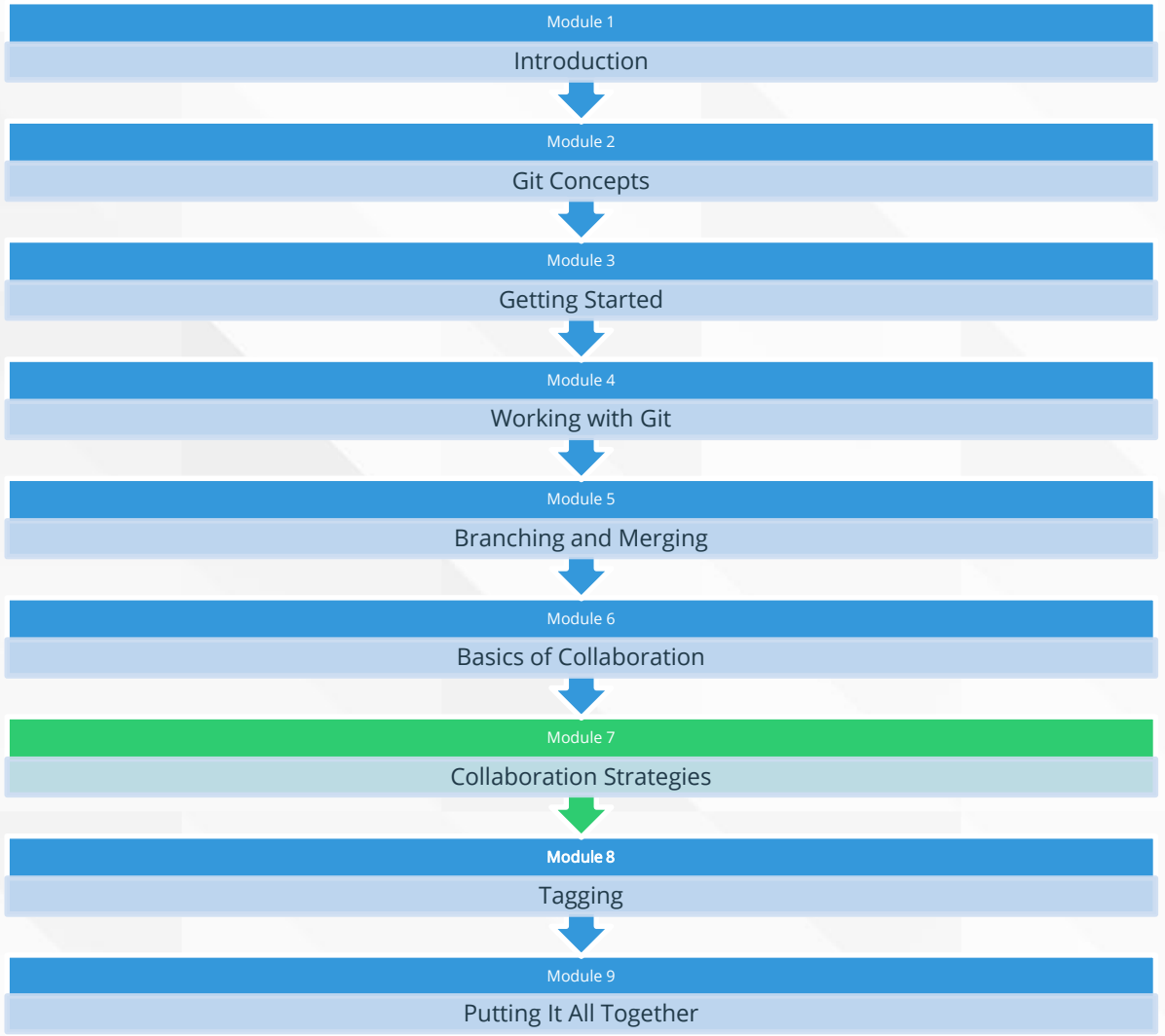
git

Module Contents

clearvision

Module Contents

- Sharing Changes
- Cloning Repositories
- Working with Remotes
- Remote Branches
- Collaboration Strategies



clearvision

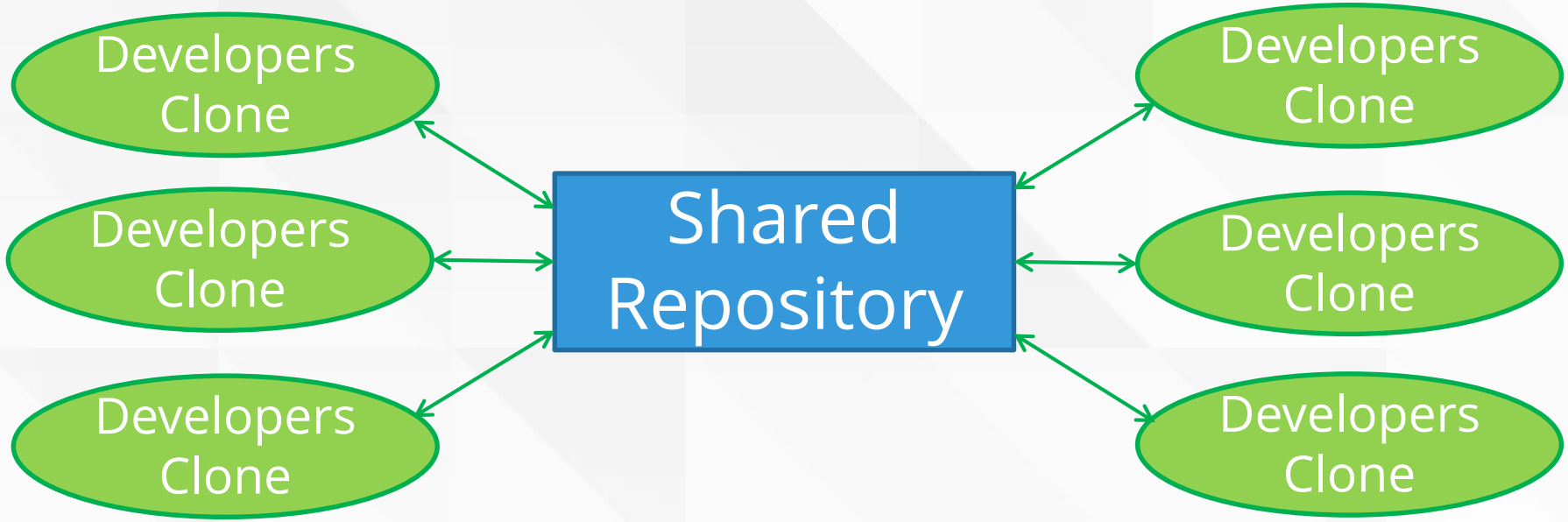
Collaboration Strategies

git

Central Repository

clearvision

- The Central Repository Model will be the model that most people recognize if they have used other SCM tools before.
- This model features a single repository on a centralized server which acts as a hub.
- All developers have a clone of the central repository and push changes to and pull changes from the central server.

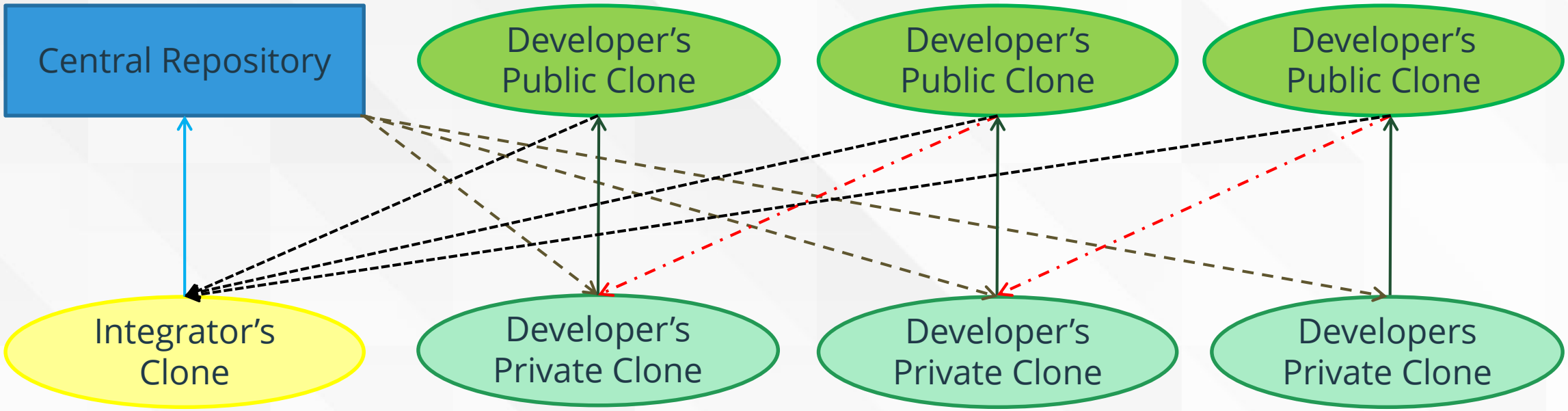


git

Integrator Model

clearvision

- In this model, only one individual (the “Integrator”) has access to the central repository
- Each developer has two repository clones. A private repository where they write code and a public repository which they push changes to and everyone else has read access.

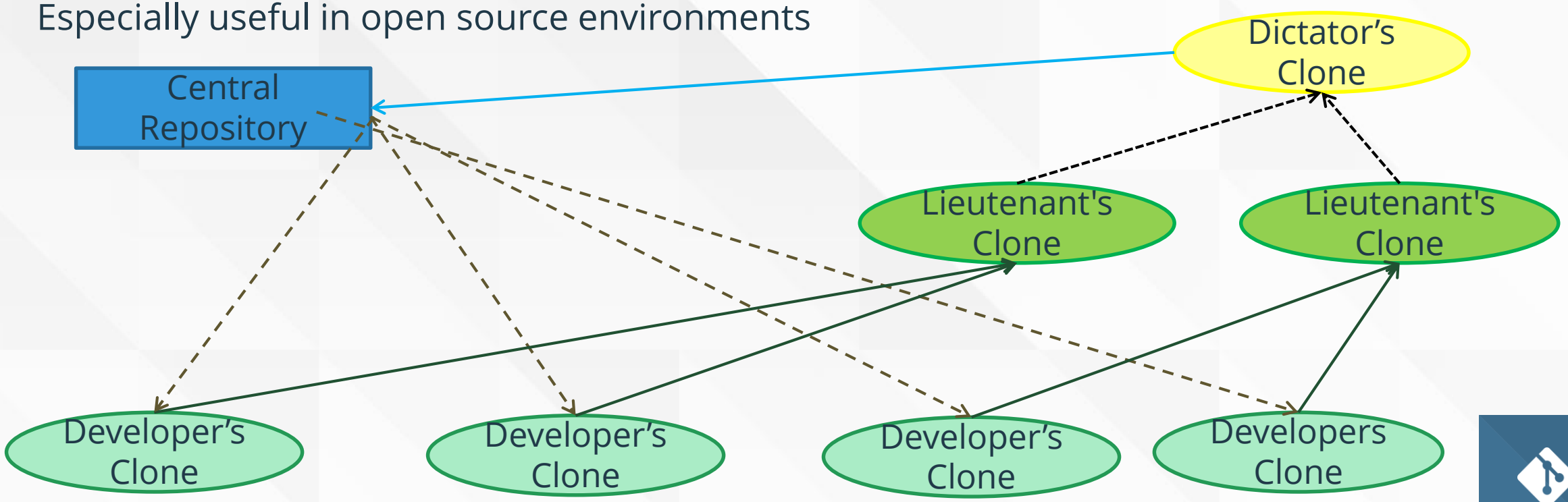


git

Hierarchical Model

clearvision

- This is a very structured model which is also referred to as the "Dictator and Lieutenant" model.
- This model is highly scalable and more suitable for larger, or more complex teams.
- Especially useful in open source environments



git

clearvision

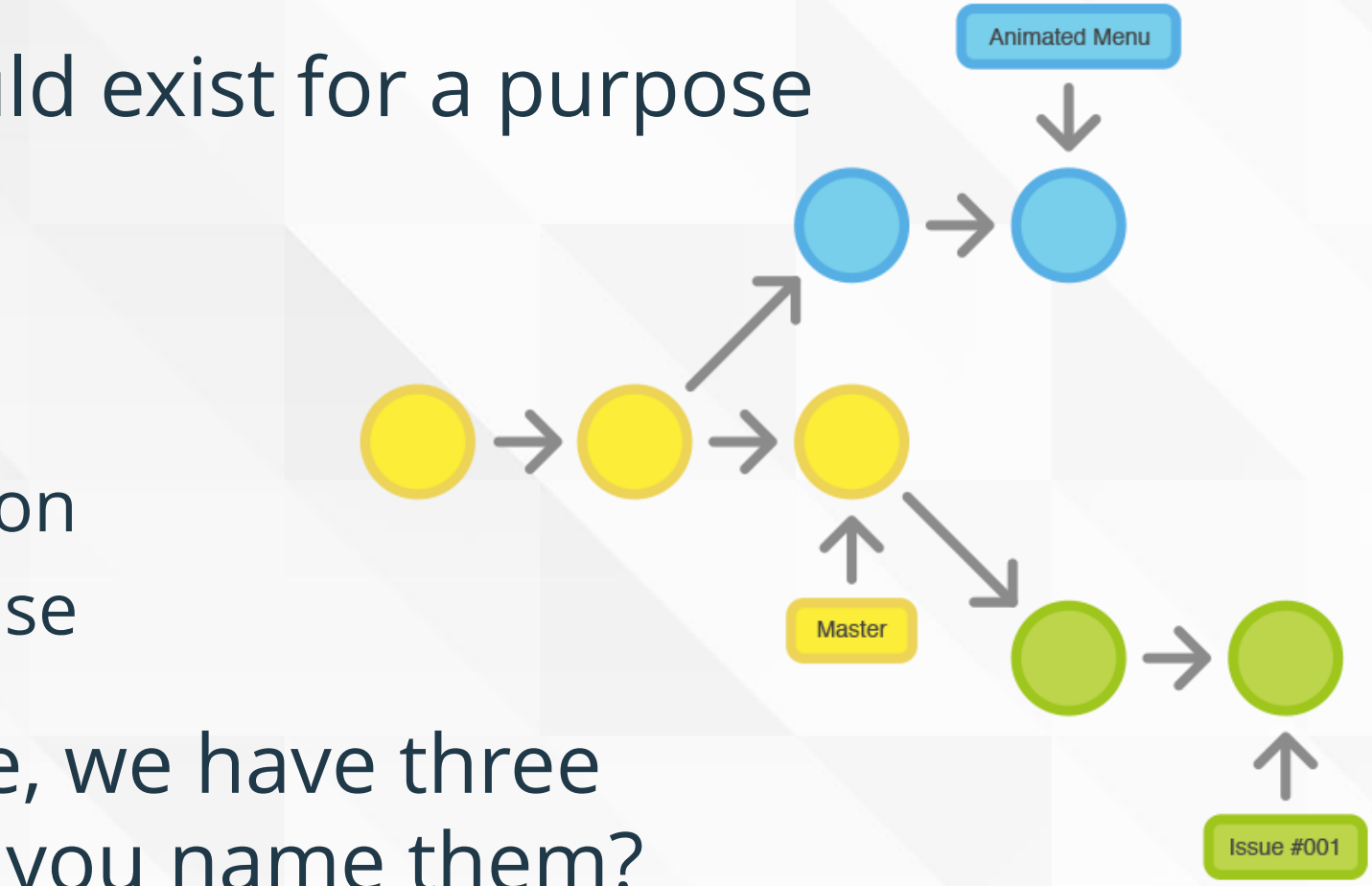
Branching Strategies

git

Branches

clearvision

- Branches should exist for a purpose
  - Version
  - Feature
  - Hotfix
  - Experimentation
  - Stable Codebase
- In this example, we have three branches. Can you name them?

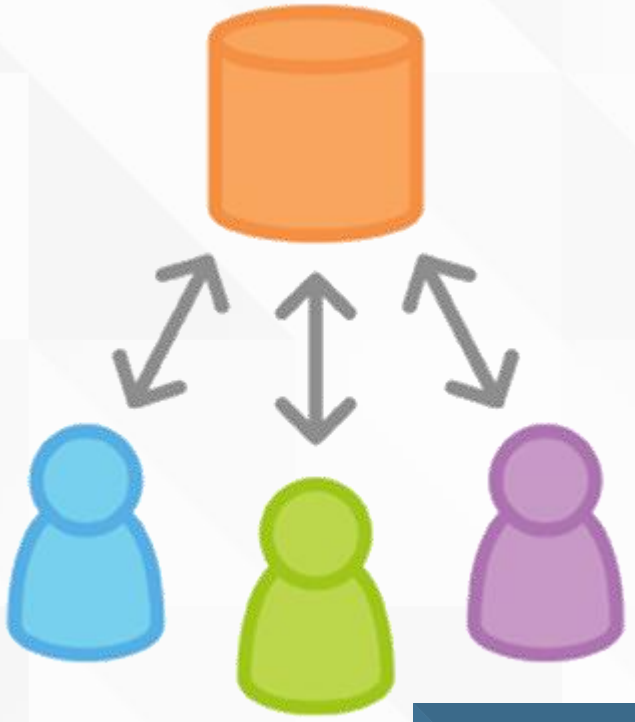


git

Centralized Workflow

clearvision

- The simplest workflow to manage these is the centralized workflow
- Requires the central repository model.
- The default development branch is the master
- All changes are committed and pushed to this branch
- This workflow doesn't require any other branches



git

Centralized Workflow

clearvision

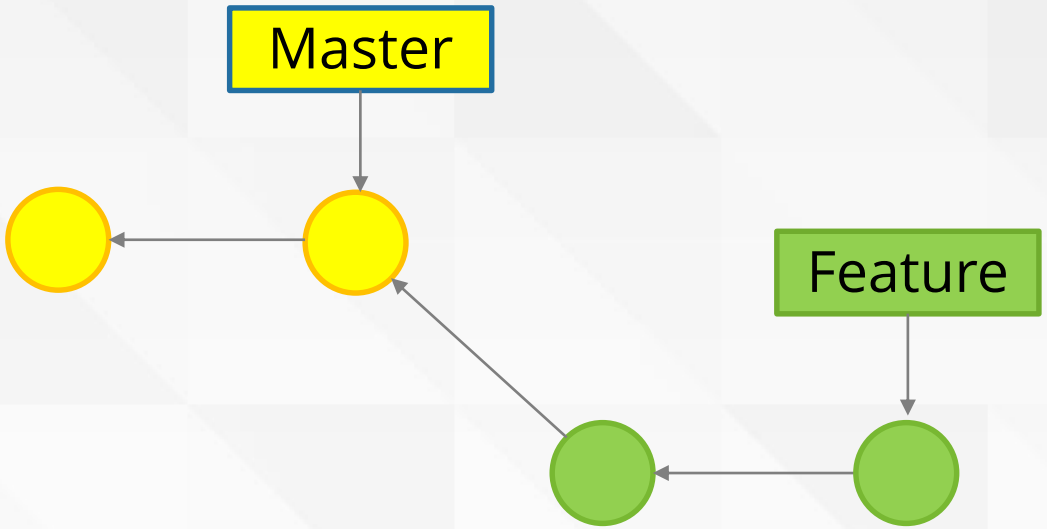
- This workflow is identical to basic centralized systems such as Subversion (using trunk), typically, it doesn't scale very well

git

Feature Branch Workflow

clearvision

- Features a single repository
- Each feature is created on an isolated branch



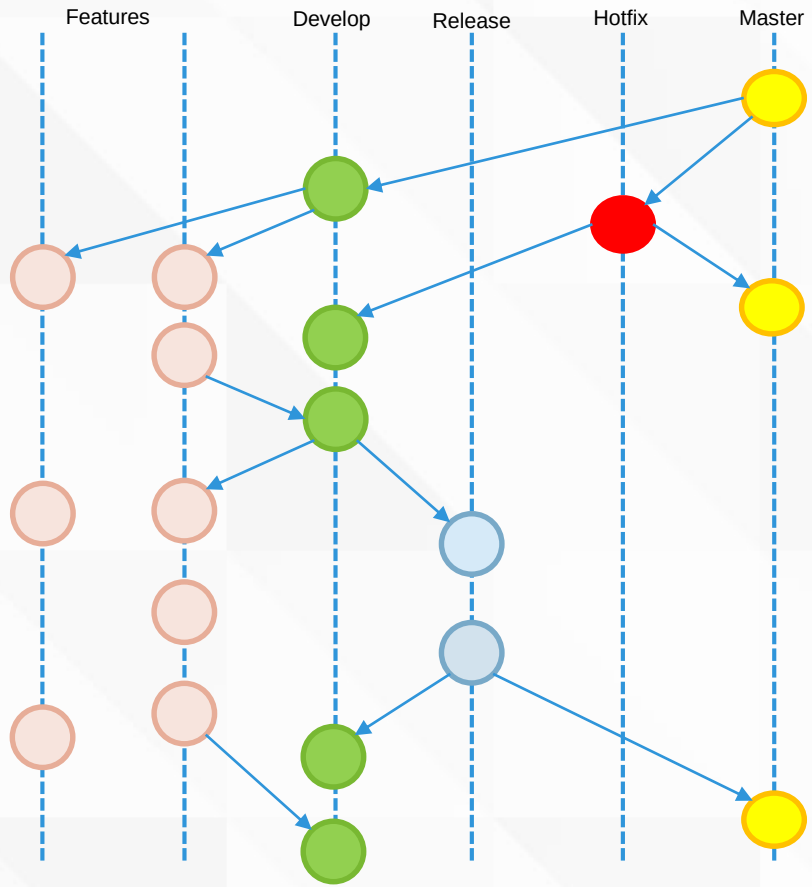
- Users are free to push to their feature branch as often as possible
- Feature should pass code review before being merged back into master

git

Gitflow Workflow

clearvision

- Created by Albert Dressed at Nvie
- Defines a strict branching model
  - Features
  - Integration
  - Releases
  - Master
  - Hotfixes
- Not as scary as it looks!

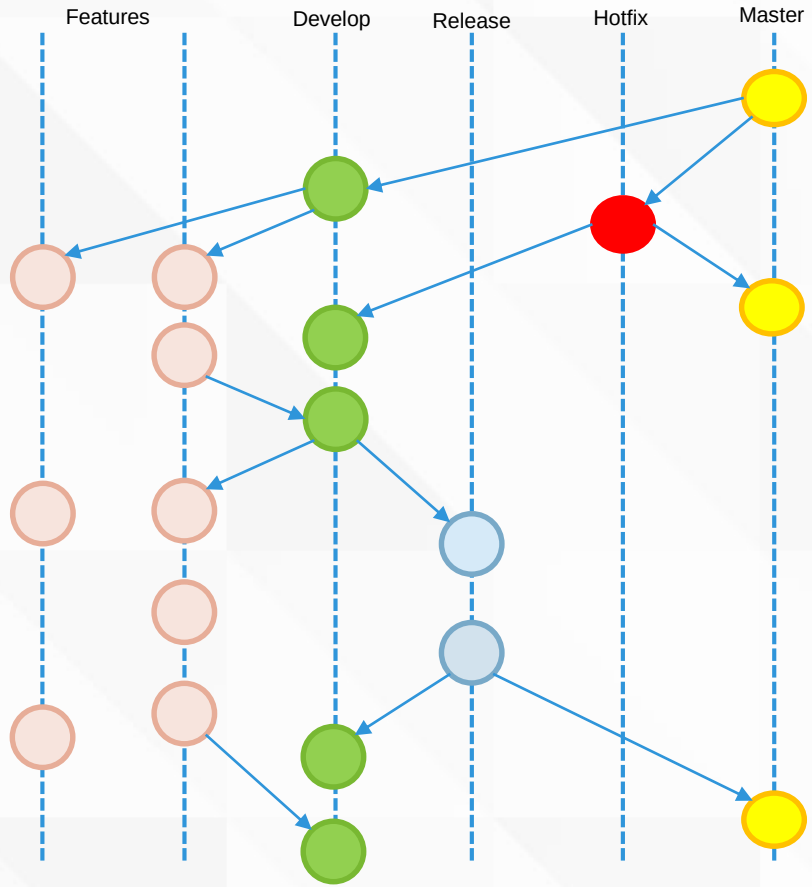


git

Gitflow Workflow

clearvision

- Master – a stable branch. Only released code lives on this branch. It is tagged as a release when merged.
- Develop – the integration branch, features are built from the latest code here and integrated back in when finished
- Feature – small short lived branches to create features. Based on the latest code from the develop branch. These often exist only on the developers machine (unless shared)



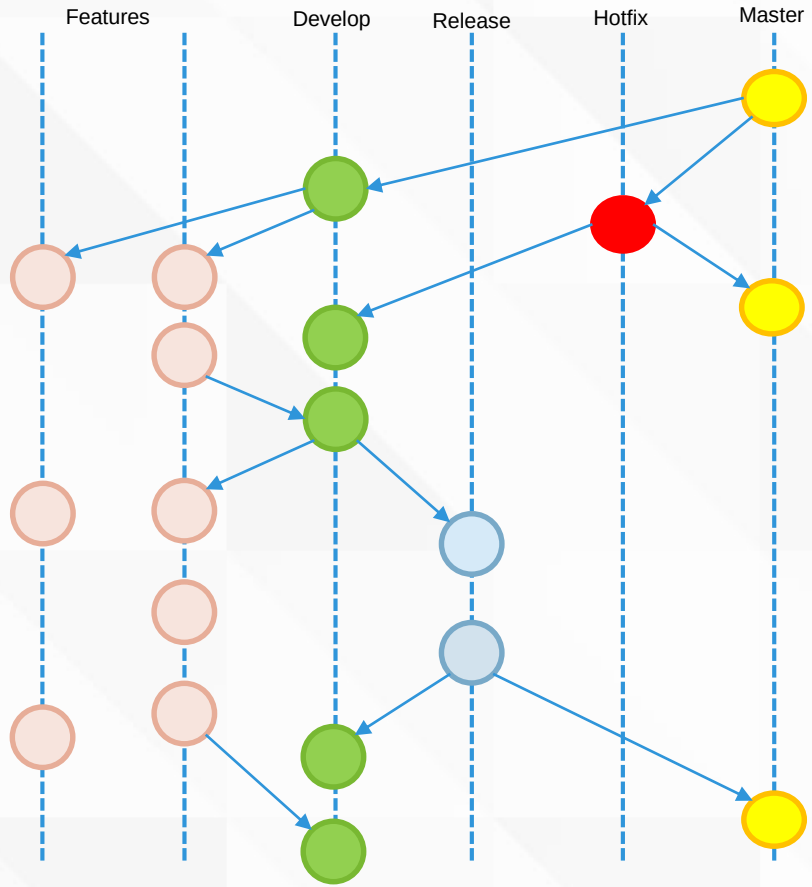
git

Gitflow Workflow

clearvision

- Release – a potentially releasable version is given its own release branch, this allows minor changes such as documentation to occur before release. This allows the develop branch to continue to evolve without affecting the release.
  Once complete, they should be merged into the master and tagged, and all changes should go back into the develop branch

- Hotfix – a short lived branch for an emergency fix on a released version. When complete, merge back into master (and create a new tag), and merge the changes back into the develop branch.



git

clearvision

Lab Exercise

git

clearvision

End of Module

git