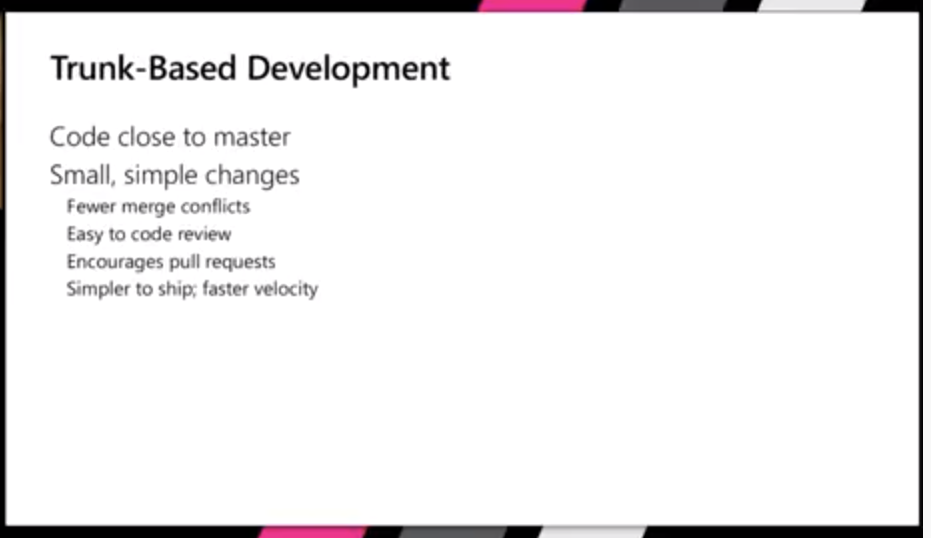
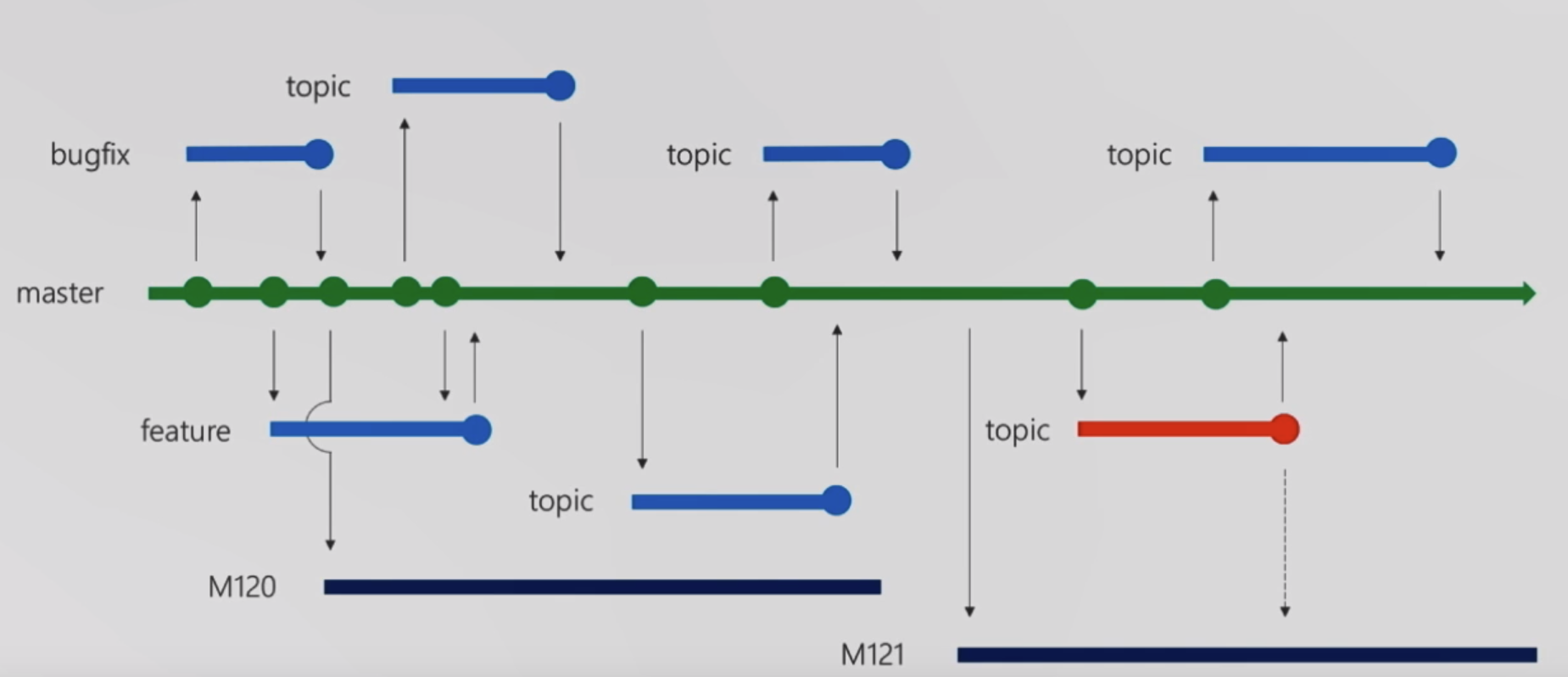
GIT Branching Strategy

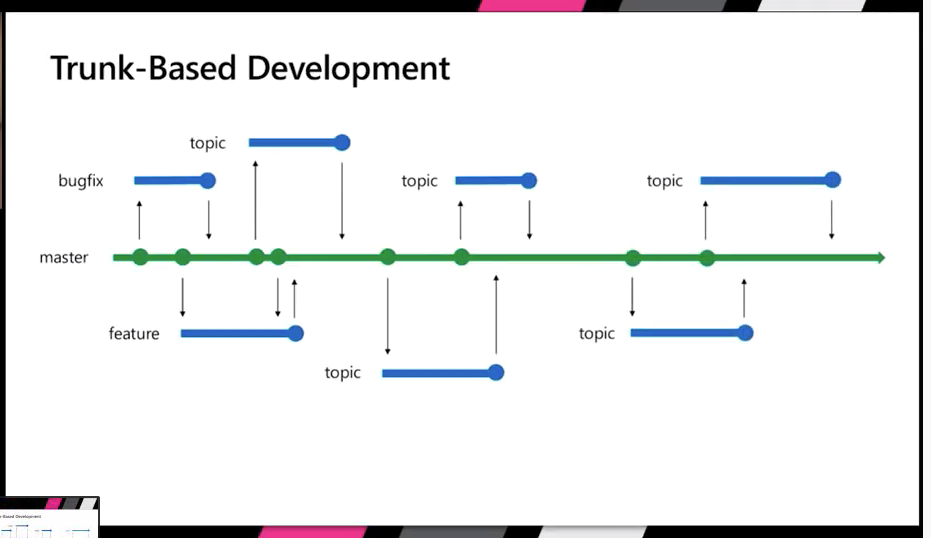
**Release Workflow Microsoft:**

Use feature flags

1. Create a branch off of master and name it based off the release, and deploy that to production
2. Hot Fix: Create a topic branch and merge it into master first. Once the code is in master, we’ll cherry pick it into production branch i.e. release branch.
3. We do not bring new features/changes into production branch unless it’s a hot fix/live side incident.
4. Never check into the release branches, always cherry pick through Master







https://docs.microsoft.com/en-us/azure/devops/learn/devops-at-microsoft/release-flow

<https://docs.microsoft.com/en-us/azure/devops/learn/devops-at-microsoft/release-flow>

**Git Flow (Doesn’t work for Multiple projects with independent releases)**

Branching Strategy 1:

1. Master at origin is the main branch, source always reflects production release state
2. Develop branch/Integration branch at origin always reflect the latest delivered development changes for next release.
   1. Merge develop branch into master and tag it with release, and push master into production
3. Feature Branch:
   1. Developed new features for upcoming releases: We can use it as release branch.
   2. Make branch from Develop branch and must be merged into develop branch
   3. Naming Conventions anything except master, develop, release-\* or hotfix-\*
4. Release Branch:
   1. Use it for minor bug, meta data, last minute fixes
   2. Merge it into Develop and Master branch
5. Hot Fix Branch:
   1. Use it for Production issues and hot fixes for prod code.
   2. Merge it back into develop and master.

**Trunk Based development:**

Custom Branching Strategy:

* Version branches: Are only needed if you plan to maintain multiple releases at once. If not, just use tags to pin your releases.
* Development & master branch: Only needed if you have to actually maintain the stable release in parallel to the development release. If not, just use tags to pin your releases.
* Use Tags

**branches should be as short-lived as possible.**

Merging Strategy:

Merge

Fast-Fwd

Rebase

Misc:

Git Hook

Pre push pipeline

Critical Path tests : resiliency gauntlet

Make develop as main branch for checkout