Advancing in open source

Manaswini Das

Solving merge conflicts

- Sending pull requests to GitHub
- Fetching remote changes from GitHub
- Attempting to merge and rebase

Cherry-picking commits

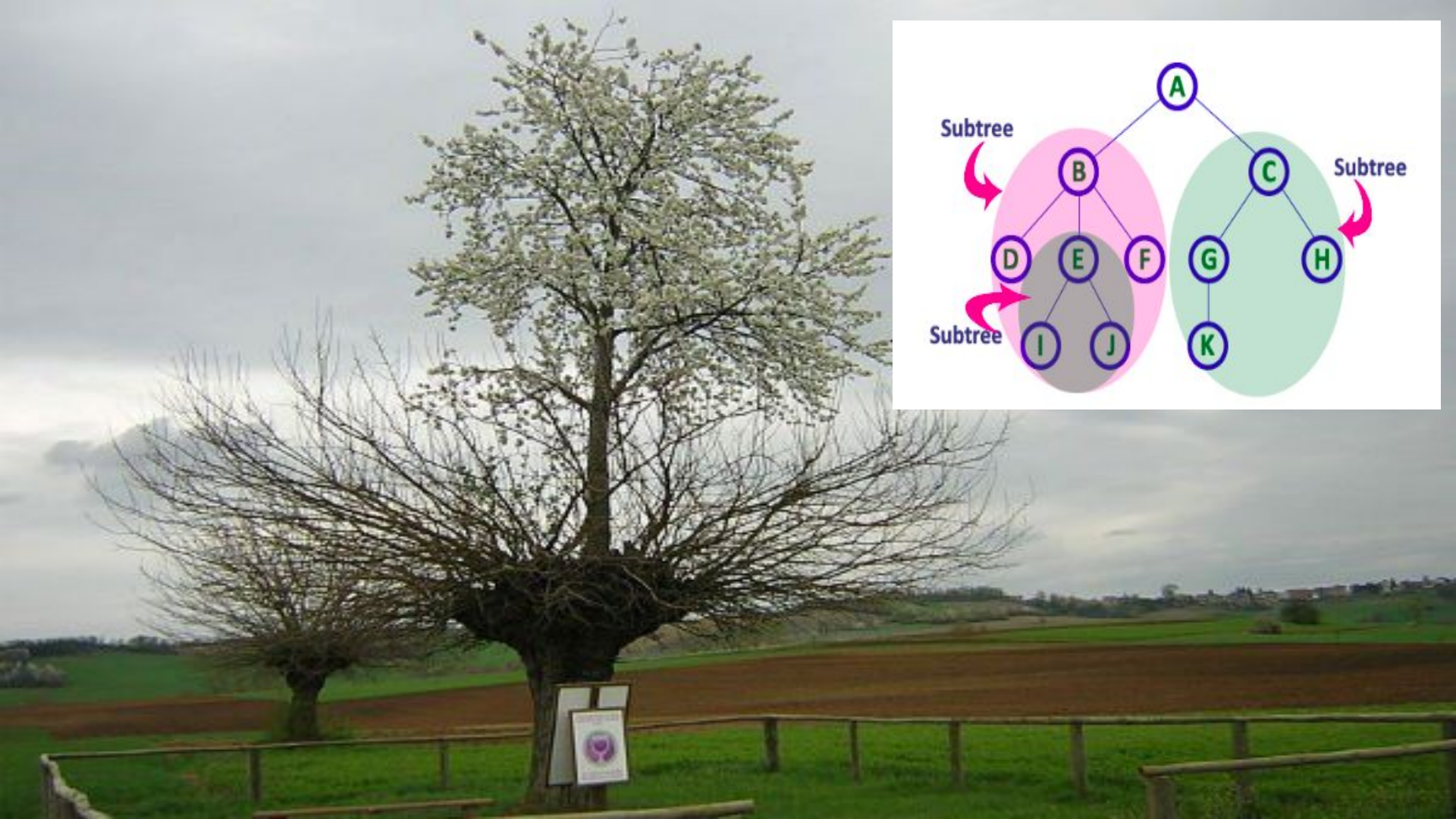
Squashing commits

Dependency management using git

Need

- Redundancy
- Cumbersome to edit
- Inability to build and test from bleeding edge checkouts

The Solution


A
Subtree
B
C
Subtree
D
E
F
G
H
Subtree
I
J
K

Git submodules

- Nest git repositories within another git repository
- Points to a specific commit reference of the child repository
- Ability to build and test from bleeding edge checkouts
- Update specific submodules

Demo time

git subtree

Introduced in 1.7.11

Alternative to git submodule to handle external dependencies.

Inject dependency

It allows you to inject an external project into a sub-folder

Extract project

It can be also used to extract a sub-folder as separate project

Demo time

Other applications/powers

- Managing project dependencies (https://github.com/manaswinidas/droolsjbpm-build-bootstrap/tree/maven-repo-test)
- Eliminates redundancy
- Publish in various platforms like gitbook(documentation)
- Makes maintenance easier

git submodule vs git subtree

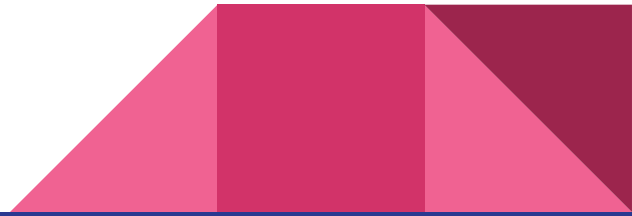
- Harder for beginners
- Just a link to a commit reference in another repository
- Requires submodule to be accessible in a server(like GitHub)
- Smaller repository size
- Component-based development

- Easier for beginners
- Code is merged in parent repository history(entire copy of child repository in parent repository)
- Decentralized
- Bigger repository size
- System-based development

Submodules vs subtrees cheat sheet:
https://github.github.com/training-kit/downloads/submodule-vs-subtree-cheat-sheet/

Caveats

- Git subtree - not a direct replacement to git submodule
- External repository you own and are likely to push code back to - use git submodule(easier to push)
- Third party code that you are unlikely to push to - use git subtree(easier to pull)

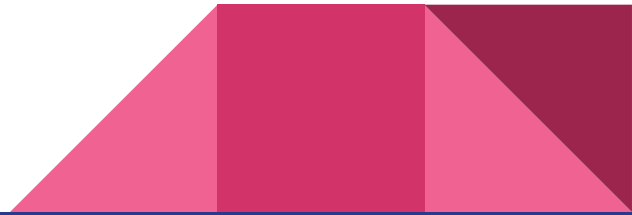


Set up CI/CD

- Development approaches
- Integrate/Deploy more often to alleviate surprises and challenges during production
- CD is a step further
- Thrusts automation and reduces human intervention

Example:

https://github.com/manaswinidas/kogito-docs



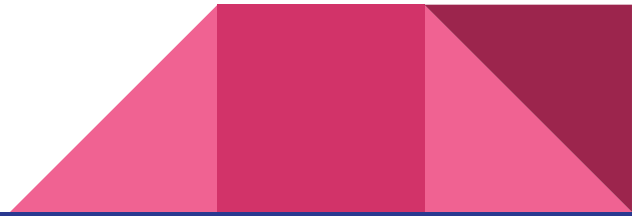
Building GitHub actions

GitHub Actions (as for other CI/CD services/platforms):

- Triggers on events
- Prepare environment(s)
- Runs workflow(s)
- Report results

Types of actions

- Dockerfile action:
  - Environment defined by container layers
  - Workflow defined by entry point bash script
- JavaScript action:
  - Environment defined in the package.json, as NPM modules
  - Workflow is defined by the JavaScript/TypeScript code



Demo time

References:

https://medium.com/@michaelekpang/creating-a-ci-cd-pipeline-using-github-actions-b65bb248edfe

Building GitHub bots

- Apps that runs automation on GitHub
- Uses GitHub WebHooks and APIs
- Enables one to focus on real collaboration

References: https://developer.github.com/v3/activity/events/types/ ,

https://github-bot-tutorial.readthedocs.io/en/latest/

Examples:

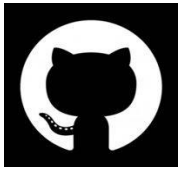
https://github.com/TimonVS/epic-generator, https://github.com/manaswinidas/github-bot-example

“In real open source, you have the right to control your own destiny”

- Linus Torvalds

You can find me

dasmanaswini10@gmail.com



@manaswinidas

@ManaswiniDas4