

Publishing maven artifacts to Jfrog

- Follow the steps
 - create an access token in jfrog cloud
 - Create a credential in jenkins using secret text to store above created token
 - install artifactory plugin
 - configure artifactory (Manage Jenkins => System => JFrog)
 - Now create a pipeline
- Note: For jf cli usage refer classroom video
- [Refer Here](#) for the changes done to publish artifacts to artifactory

The screenshot shows the Jenkins web interface. The browser address bar indicates the URL is 20.102.112.13:8080/job/spc-declarative/1/. The Jenkins logo is visible in the top left, and a search bar is in the top right. The main content area displays the status of 'Build #1' (Apr 14, 2024, 3:02:36 AM) with a green checkmark icon. A sidebar on the left lists various build actions: Status, Changes, Console Output, Edit Build Information, Delete build '#1', Git Build Data, Maven Artifacts, Test Result, Artifactory Build Info, Restart from Stage, Replay, and Pipeline Steps. The main area shows build details: 'Started by user qtdevops', 'Revision: b22f0bc557a6ecdd453b8d3fe8b1d16a99b9957c', and 'Repository: https://github.com/dummyrepos/spring-petclinic-apr24.git'. It also lists 'Deployed Maven Artifacts' as 'spring-petclinic-3.2.0-SNAPSHOT.jar' and 'spring-petclinic-3.2.0-SNAPSHOT.pom'. A 'Test Result' link shows '(no failures)'. A 'Keep this build' button is in the top right corner.

- Break till 8:50 AM

Upstream and Downstream Projects

- This concept is used mostly in free style projects

The image illustrates the concept of project dependencies in Jenkins. It features a diagram and a screenshot of the Jenkins web interface.

Diagram: A vertical flowchart on a dark grid background shows three boxes: 'Project A' at the top, 'Project B' in the middle, and 'Project C' at the bottom. Arrows point from 'Project A' to 'Project B' and from 'Project B' to 'Project C', indicating a sequential dependency.

Jenkins UI Screenshot: The screenshot shows the Jenkins web interface for 'project B'. The breadcrumb navigation is 'Dashboard > project B >'. The left sidebar contains links: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area shows 'project B' with a list of 'Upstream Projects' (project A) and 'Downstream Projects' (project c). Both 'project A' and 'project c' are circled in green. Below this, there are 'Permalinks' for 'Build History' (No builds) and 'Atom feed for all' / 'Atom feed for failures'. The browser address bar shows '20.102.112.13:8080/job/project%20B/'.

Pull request based workflow in CI/CD pipelines

- Our jenkins jobs are triggered when the developer pushes the changes, if the build is failed then this commit has to be reverted from git

- To make this better lets try understanding the feature branches workflow
- Developer creates a feature branch
- makes changes
- pushes the changes
- now creates a pull request to consider feature branch into developer branch
- Now CI/CD should run build tests static code, Quality Gate and then the other peer will approve the merge/pull request

Try setting up the Pull request workflow in jenkins

- Option 1: [Refer Here](#) try finding a pull request plugin for github
- option 2: Web hook