Gitflow - Fix production release

Agile Pod Services - CoP

Exported on 02/11/2024

Table of Contents

1

Situation

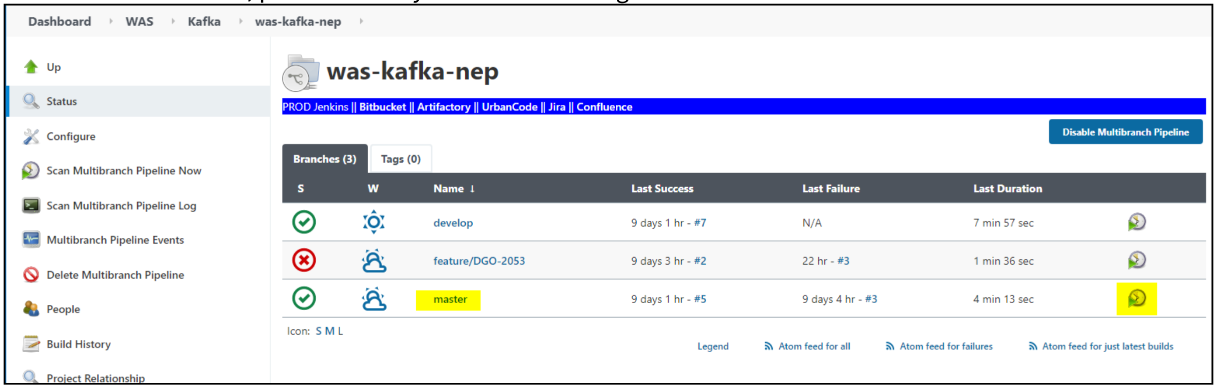
A production incident requires code change, and needs to go live before the next regular UCC release.

The starting point is the currently deployed application release in production, which corresponds to a specific Git tag.

2 Create hotfix branch

The Jenkins pipeline job is used to create a hotfix branch that is based on a specific tag.

1. Go to the pipeline job of the application. In this **example** we work with the application NEP.
2. On the branch **master**, press the "Play" button on the right side:



Dashboard › WAS › Kafka › was-kafka-nep ›

Up
Status
Configure
Scan Multibranch Pipeline Now
Scan Multibranch Pipeline Log
Multibranch Pipeline Events
Delete Multibranch Pipeline
People
Build History
Project Relationship

was-kafka-nep

PROD Jenkins || Bitbucket || Artifactory || UrbanCode || Jira || Confluence

Disable Multibranch Pipeline

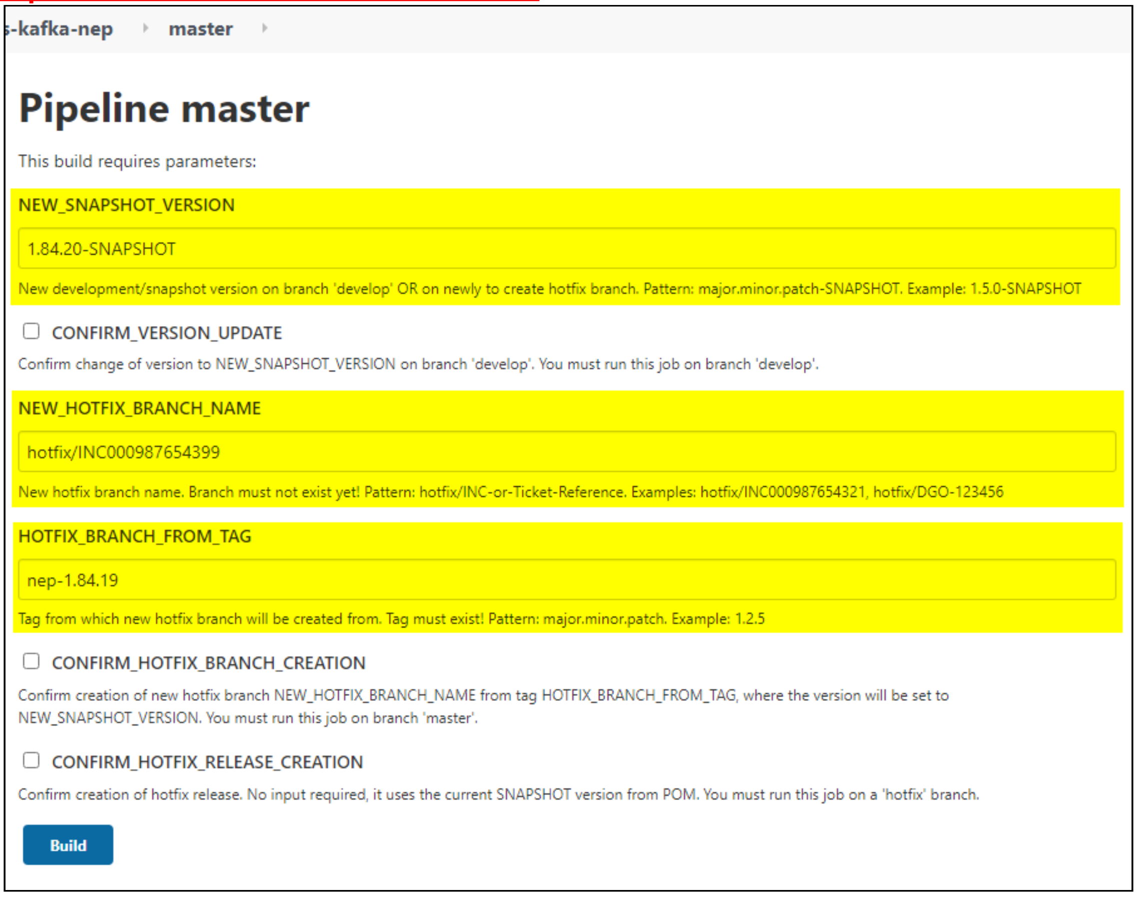
Branches (3) | Tags (0)

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☀	develop	9 days 1 hr - #7	N/A	7 min 57 sec
✗	⛅	feature/DGO-2053	9 days 3 hr - #2	22 hr - #3	1 min 36 sec
✓	⛅	master	9 days 1 hr - #5	9 days 4 hr - #3	4 min 13 sec

Icon: S M L

Legend   Atom feed for all   Atom feed for failures   Atom feed for just latest builds

3. You need to provide three pieces of information:
   a. the **new SNAPSHOT version name** (the pom.xml will be updated with this version on the to-be created hotfix branch). Most likely this is the next patch version after the version of the tag.
   b. the **name of the hotfix branch** to create. Must always start with **hotfix/.** You can use the SmartIT incident id, or the JIRA ticket that was created for this hotfix code change.
   c. the **tag from which the hotfix branch will start**. **This must correspond to the tag that is deployed in production. Double-check this information!**



-kafka-nep › master ›

Pipeline master

This build requires parameters:

NEW_SNAPSHOT_VERSION

1.84.20-SNAPSHOT

New development/snapshot version on branch 'develop' OR on newly to create hotfix branch. Pattern: major.minor.patch-SNAPSHOT. Example: 1.5.0-SNAPSHOT

☐ CONFIRM_VERSION_UPDATE

Confirm change of version to NEW_SNAPSHOT_VERSION on branch 'develop'. You must run this job on branch 'develop'.

NEW_HOTFIX_BRANCH_NAME

hotfix/INC000987654399

New hotfix branch name. Branch must not exist yet! Pattern: hotfix/INC-or-Ticket-Reference. Examples: hotfix/INC000987654321, hotfix/DGO-123456

HOTFIX_BRANCH_FROM_TAG

nep-1.84.19

Tag from which new hotfix branch will be created from. Tag must exist! Pattern: major.minor.patch. Example: 1.2.5

☐ CONFIRM_HOTFIX_BRANCH_CREATION

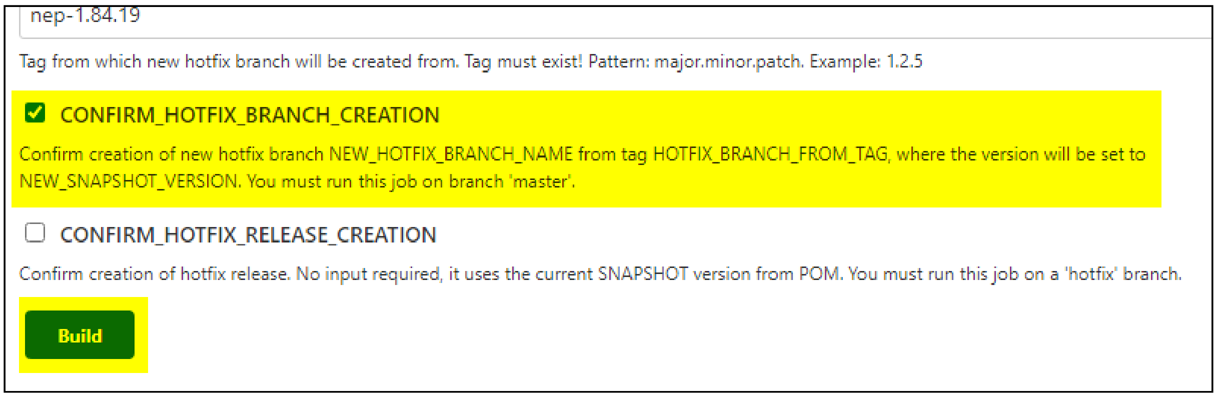
Confirm creation of new hotfix branch NEW_HOTFIX_BRANCH_NAME from tag HOTFIX_BRANCH_FROM_TAG, where the version will be set to NEW_SNAPSHOT_VERSION. You must run this job on branch 'master'.

☐ CONFIRM_HOTFIX_RELEASE_CREATION

Confirm creation of hotfix release. No input required, it uses the current SNAPSHOT version from POM. You must run this job on a 'hotfix' branch.

Build

4. If the three values look sound, you click the "CONFIRM_HOTFIX_BRANCH_CREATION" checkbox, and press the "Build" button.



nep-1.84.19

Tag from which new hotfix branch will be created from. Tag must exist! Pattern: major.minor.patch. Example: 1.2.5

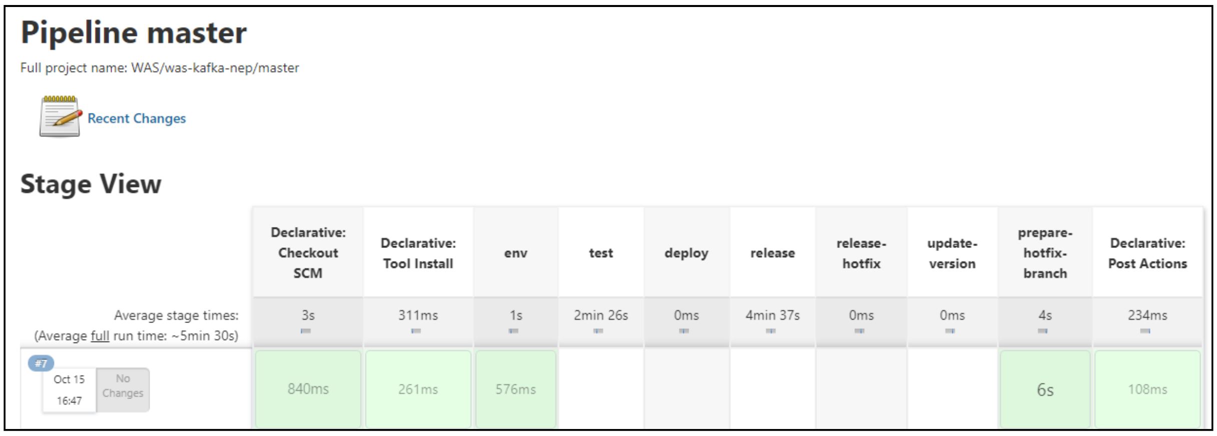
☑ CONFIRM_HOTFIX_BRANCH_CREATION

Confirm creation of new hotfix branch NEW_HOTFIX_BRANCH_NAME from tag HOTFIX_BRANCH_FROM_TAG, where the version will be set to NEW_SNAPSHOT_VERSION. You must run this job on branch 'master'.

☐ CONFIRM_HOTFIX_RELEASE_CREATION

Confirm creation of hotfix release. No input required, it uses the current SNAPSHOT version from POM. You must run this job on a 'hotfix' branch.

5. Wait until the pipeline is finished. This should only take a few seconds.



Pipeline master

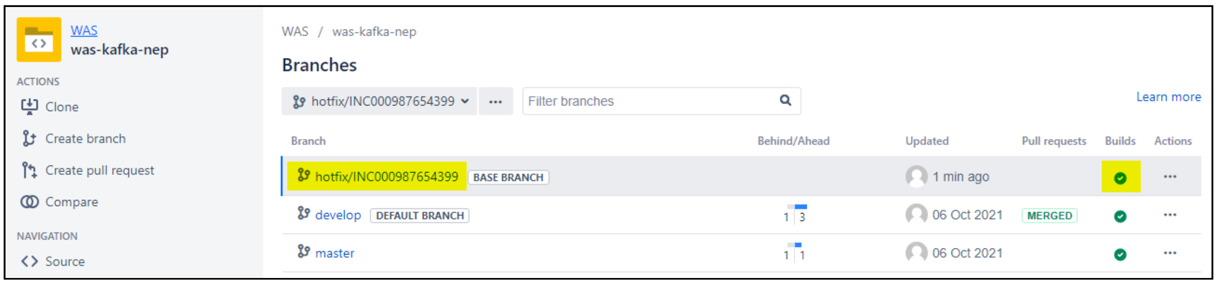
Full project name: WAS/was-kafka-nep/master

Recent Changes

Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	env	test	deploy	release	release-hotfix	update-version	prepare-hotfix-branch	Declarative: Post Actions
Average stage times: (Average full run time: ~5min 30s)	3s	311ms	1s	2min 26s	0ms	4min 37s	0ms	0ms	4s	234ms
#7 Oct 15 16:47 No Changes	840ms	261ms	576ms						6s	108ms

6. The hotfix branch was created in Bitbucket, and a first pipeline run was automatically executed in Jenkins. You are now ready to work with the hotfix branch.

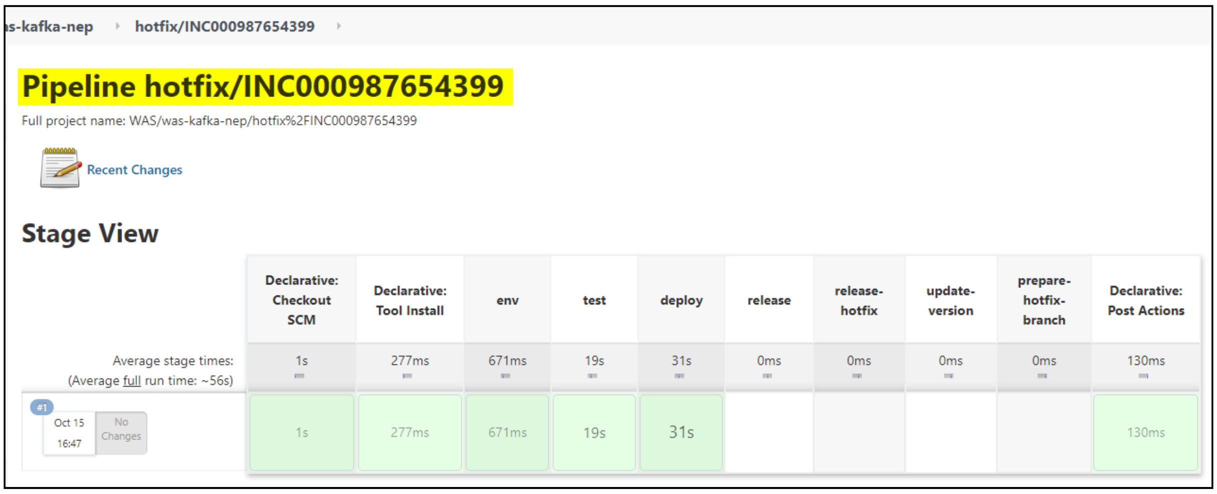


WAS
was-kafka-nep

WAS / was-kafka-nep

Branches

Branch	Behind/Ahead	Updated	Pull requests	Builds	Actions
hotfix/INC000987654399 BASE BRANCH		1 min ago		✓	···
develop DEFAULT BRANCH	1 3	06 Oct 2021	MERGED	✓	···
master	1 1	06 Oct 2021		✓	···



s-kafka-nep › hotfix/INC000987654399 ›

Pipeline hotfix/INC000987654399

Full project name: WAS/was-kafka-nep/hotfix%2FINC000987654399

Recent Changes

Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	env	test	deploy	release	release-hotfix	update-version	prepare-hotfix-branch	Declarative: Post Actions
Average stage times: (Average full run time: ~56s)	1s	277ms	671ms	19s	31s	0ms	0ms	0ms	0ms	130ms
#1 Oct 15 16:47 No Changes	1s	277ms	671ms	19s	31s					130ms

3 Work on hotfix branch

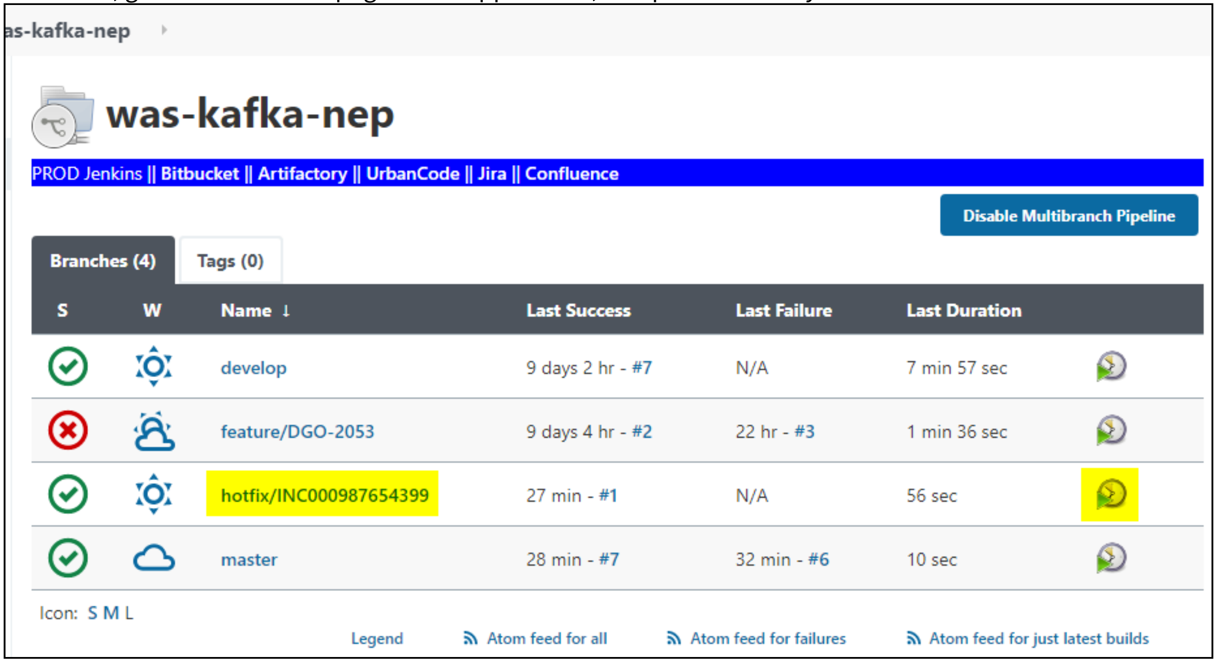
Again, in this **example** we work with the application NEP, and continue with the created hotfix branch from the previous section.

1. Depending on your local state, you might already have the application's repository, or not
   a. if you already have the repo, pull the new content, and switch to the hotfix branch (assuming you have a clean working tree):
      `git pull; git switch hotfix/INC000987654399`
   b. if you don't have the repo yet, then might want to follow the workspace setup instructions for that application as you don't seem to have developed for it yet.
      Alternatively you can just clone the repository and switch to the hotfix branch
      `git clone ssh://git@git.ucc:7999/was/was-kafka-nep.git; git switch hotfix/INC000987654399`
2. Work as usual with the local repository, i.e. add and commit your work.
3. When you reach the point where a new build is appropriate, maybe combined with a first deployment in the development environment, then push your commits:
   `git push`
   Again the pipeline of the hotfix branch will then run, and install a new revision of the snapshot version in Artifactory which you can deploy viaUCD.
4. When you are done with coding, and you have tested your changes sufficiently; both inthe local as well as in the development environment, you are ready to create a release version.

4 Create release for hotfix

The creation of a release version is again done with manual run of the pipeline.

1. In Jenkins, go to the overview page of the application, and press the "Play" button of the hotfix branch:



as-kafka-nep

was-kafka-nep

PROD Jenkins || Bitbucket || Artifactory || UrbanCode || Jira || Confluence

Disable Multibranch Pipeline

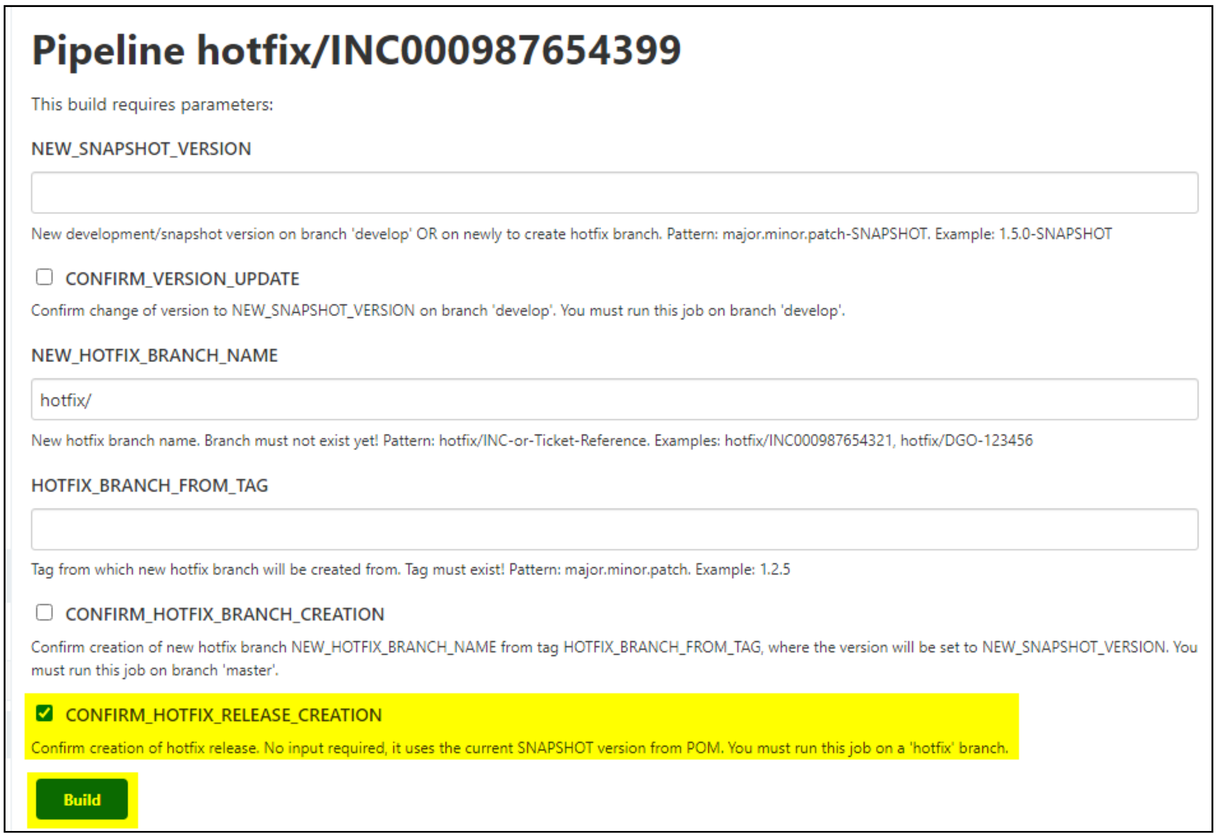
Branches (4) Tags (0)

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	develop	9 days 2 hr - #7	N/A	7 min 57 sec	▶
✗	⛅	feature/DGO-2053	9 days 4 hr - #2	22 hr - #3	1 min 36 sec	▶
✓	☀	hotfix/INC000987654399	27 min - #1	N/A	56 sec	▶
✓	☁	master	28 min - #7	32 min - #6	10 sec	▶

Icon: S M L

Legend Atom feed for all Atom feed for failures Atom feed for just latest builds

2. All you have to do now is to tick the checkbox "CONFIRM_HOTFIX_RELEASE_CREATION", and to press the "Build" button.



Pipeline hotfix/INC000987654399

This build requires parameters:

NEW_SNAPSHOT_VERSION

New development/snapshot version on branch 'develop' OR on newly to create hotfix branch. Pattern: major.minor.patch-SNAPSHOT. Example: 1.5.0-SNAPSHOT

☐ CONFIRM_VERSION_UPDATE

Confirm change of version to NEW_SNAPSHOT_VERSION on branch 'develop'. You must run this job on branch 'develop'.

NEW_HOTFIX_BRANCH_NAME

hotfix/

New hotfix branch name. Branch must not exist yet! Pattern: hotfix/INC-or-Ticket-Reference. Examples: hotfix/INC000987654321, hotfix/DGO-123456

HOTFIX_BRANCH_FROM_TAG

Tag from which new hotfix branch will be created from. Tag must exist! Pattern: major.minor.patch. Example: 1.2.5

☐ CONFIRM_HOTFIX_BRANCH_CREATION

Confirm creation of new hotfix branch NEW_HOTFIX_BRANCH_NAME from tag HOTFIX_BRANCH_FROM_TAG, where the version will be set to NEW_SNAPSHOT_VERSION. You must run this job on branch 'master'.

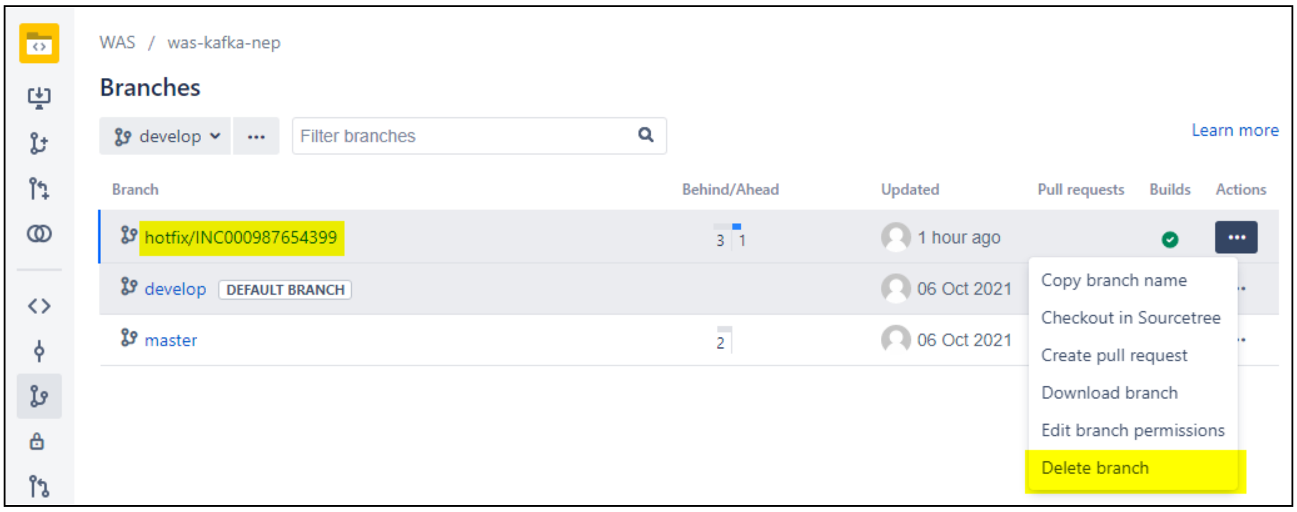
☑ CONFIRM_HOTFIX_RELEASE_CREATION

Confirm creation of hotfix release. No input required, it uses the current SNAPSHOT version from POM. You must run this job on a 'hotfix' branch.

3. Wait for the pipeline to finish. The result is a new tag - in our example that would be 1.84.20 - in the repository, and the built modules in Artifactory.
   The Maven Release plugin also prepared the next snapshot version, in case you need to add more changes to the hotfix branch.
4. Regarding the hotfix release, you're all settled. But wait, what about the code changes which fix a super critical bug in the application?
   Unfortuately, there is no automated process that merges the relevant changes from the hotfix branch to the develop branch. The pom.xml files that were changed are most likely not valid for the current state of the develop branch, which is already one minor version ahead. So a direct pull request from the hotfix to the develop branch won't work.
   Therefore, you have to **manually apply the necessary code changes on the develop branch. This is part of the hotfix DoD.** You have several options:
   a. create a feature branch from the current develop branch and try to merge the hotfix branch changes into the feature branch. If only the pom.xml files give you merge conflicts, then you're in luck.
   b. create a feature branch from the develop branch and manually apply the code changes. This might be the fastest, although not safest, way when only few changes occured.

5 Cleanup hotfix branch

Since no pull requests are used, you have to **manually delete the hotfix branch in Bitbucket.**



Branch	Behind/Ahead	Updated	Pull requests	Builds	Actions
hotfix/INC000987654399	3 1	1 hour ago		✓	•••
develop DEFAULT BRANCH		06 Oct 2021			
master	2	06 Oct 2021			

It is recommend to wait until the production deployment of the fix release is over, and you have confirmation that the changes solved the problem.