

# How to Customize Checkout for Pipeline Multibranch?

Yesterday at 19:39

## Issue

- I am using Pipeline Multibranch and/or GitHub Organization and/or BitBucket Team Project - i.e. any project item that leverage MultiBranch Pipelines.
- I want to define additional behaviors to my SCM for specific branches - like I would do with the checkout step in a simple Pipeline job.
- I want a pipeline-as-code solution to customize SCM checkout for Multibranch Pipeline Projects.

## Environment

- CloudBees Jenkins Enterprise
- Jenkins
- Pipeline Plugin
- Pipeline Multibranch Plugin
- CloudBees BitBucket Branch Source Plugin
- GitHub Branch Source Plugin

## Related Issues

JENKINS-37658  
JENKINS-33022  
JENKINS-31924

## Resolution

**IMPORTANT:** The following approach is a workaround and should not be used once JENKINS-37658 has been fixed

For MultiBranch Pipelines, the source code can be checked out with a simple checkout `scm` in the `Jenkinsfile`. In some cases, a different behavior needs to be defined for specific branches. For that, there is a **pipeline-as-code** solution.

The checkout step accepts the `scm` variable. This variable references an **Object** which means that attributes of this object can be accessed via `scm.$attributeName`. Therefore the checkout `scm` can be customized.

In brief, for the following MultiBranch configuration:

Branch Sources

Git

Project Repository

https://github.beescloud.com/support-team/private-repo

Credentials

support-team/\*\*\*\*\* (support-team-up) 

Add

Ignore on push notifications

☐

Include branches

\*

Exclude branches

Property strategy

All branches get the same properties

Add property

Delete source

the following Jenkinsfile:

```
node {
    checkout scm
}
```

is equivalent to something like this:

```
node {
    checkout([
        $class: 'GitSCM',
        branches: scm.branches,
        doGenerateSubmoduleConfigurations: scm.doGenerateSubmoduleConfigurations,
        extensions: scm.extensions,
        userRemoteConfigs: scm.userRemoteConfigs
    ])
}
```

With one difference though: in the latter **the access to scm.\* attributes need to be whitelisted by an administrator in Jenkins**

## Some Explanation

In the context of Pipeline Multibranch, for each branch the SCM configuration is “*injected*”in the variable scm.

This variable is an object implementation of an SCM. For example with Git, the scm references a GitSCM Object. You can see that in the source code and/or when using the Snippet Generator. By default, the snippet generator for **checkout > git** will generate something like the following:

```
checkout([$class: 'GitSCM',
    branches: [[name: '*/master']],
    doGenerateSubmoduleConfigurations: false,
    extensions: [],
    submoduleCfg: [],
    userRemoteConfigs: []])
```

This object contains attributes, for example branches, extensions, userRemoteConfigs. Inside the Jenkinsfile, these attributes can be accessed via scm.branches, scm.extensions, scm.userRemoteConfigs. Some attributes are mandatory, some are optional or have a default value and therefore don’t need to be specified.

*Note: While the Snippet Generator gives a pretty good idea of the attributes accessible, it is sometimes required to have a look at the source code to understand how to map them to the constructor.*

## Implications

When the Pipeline is read from SCM - like for Multibranch Pipelines - it is **sandboxed** (see Script Security Plugin). This means that any access to scm attributes need to be approved by an Administrator in Jenkins. This can be controlled under **Manage Jenkins > In-process Script Approval**.

## Example

So we have seen that it is possible to customize our checkout scm step. Here is a simple scenario: for all my branches, I want to checkout the code and carry on with my workflow whereas for the release/\* branches I want to *Clean before Checkout*- which is a Git additional Behavior:

For all branches, my Jenkinsfile looks like this:

```
node {
    checkout scm

    //Build, Test, Stage, Deploy
    [...]
}
```

In my feature/\* branches, I want to *Clean After Checkout* so I am gonna change the Jenkinsfile. I use the Snippet Generator for **checkout > Git** and add the the Additional Behavior *Clean After Checkout*.

Steps

Sample Step checkout: General SCM

SCM Git

Repositories

Repository URL https://github.com/user/repo.git

Credentials dohbedoh (git-credentials) Add

Advanced...

Add Repository

Branches to build

Branch Specifier (blank for 'any') \*/master

Add Branch

Delete Branch

Repository browser (Auto)

Additional Behaviours

Clean after checkout

Add

Include in polling?

Include in changelog?

Let’s see which attribute I need to add:

```
checkout([$class: 'GitSCM',
    branches: [[name: '*/master']],
    doGenerateSubmoduleConfigurations: false,
    extensions: [[$class: 'CleanCheckout']],
    submoduleCfg: [],
    userRemoteConfigs: [[credentialsId: 'git-credentials', url: 'https://github.com/user/repo.git']]
])
```

I need to add the `[[ $class: 'CleanCheckout' ]]` to extensions. Attributes other than branches and userRemoteConfigs don't need to be specified in my case. So I modify the Jenkinsfile and append this attribute to the scm.extensions that are passed to the flow:

```
node {
    checkout([
        $class: 'GitSCM',
        branches: scm.branches,
        extensions: scm.extensions + [[ $class: 'CleanCheckout' ]],
        userRemoteConfigs: scm.userRemoteConfigs
    ])

    //Build, Test, Stage, Deploy
    [...]
}
```

The Jenkins Administrator needs to approve the following methods:

```
method hudson.plugins.git.GitSCM getBranches
method hudson.plugins.git.GitSCM getUserRemoteConfigs
method hudson.plugins.git.GitSCMBackwardCompatibility getExtensions
staticMethod org.codehaus.groovy.runtime.DefaultGroovyMethods plus java.util.Collection java.lang.Object
```

## Further Information

Since the release of the **version 2.5.2 of the Git Plugin**, it is now possible to specify Advanced Behaviors in the Git SCM configuration of MultiBranch Project:



## Documentation

- Pipeline SCM Step Plugin - Generic SCM step
- Pipeline Plugin Compatibility - SCMs
- Snippet Generator
- Pipeline Multibranch
- GitHub Organization
- BitBucket Team Project
- Pipeline-as-code with Multibranch Workflows in Jenkins
- Script Security Plugin

[multibranch-simple-git-config.png-4276ac6](#) (50 KB)

[sgenerator-git-advanced.png-31275c63](#) (80 KB)

[multibranch-git-scm.png-b1414120](#) (50 KB)

Have more questions? Submit a request

7 Comments


Date

Votes

^

0

v



Giorgio Sironi

January 24, 2017 05:05

Interesting to see how this can be done with pipeline code (which is not very clear at <https://jenkins.io/doc/pipeline/steps/workflow-scm-step/#checkout-general-scm>).

It seems however we are replicating information about the build configuration inside the pipeline definition, replicating attributes like:

```
userRemoteConfigs: scm.userRemoteConfigs
```

Is there a way to just specify what changes, altering the scm object or making a full copy of it with some customization?

⚙️

v

^

0

v



Rbabzba 123

April 08, 2017 05:48

I was trying to achieve the **SparseCheckout** but failing so far, your help would be highly appreciated. It checkout the whole branch instead of specified folder.

```
checkout([
    $class: 'GitSCM',
    branches:[[name: '*/branchName']],
    extensions: scm.extensions + [[ $class: 'SparseCheckoutPaths' ], path:'FolderName'],
    userRemoteConfigs: scm.userRemoteConfigs
])
```

Edited by Rbabzba 123 10 months ago


⚙️

v

^

0

v



Viacheslav Dubrovskiy

May 11, 2017 11:56

Hi.

I would like use reference repository with credential. For example from Google Cloud source. Is it possible? Help me please.

⚙️

v

^

0

v



Christian Schulz

August 21, 2017 01:55

Not all additional behaviors seems to be supported.


⚙️

v

^

0

v



Denys Digtar

August 21, 2017 18:21

Hi Christian,

Your comment cannot really be addressed since it lacks any context and information about your environment. If you are CloudBees customer feel free to raise a support ticket, if not please approach the Jenkins community on regular channels (e.g. IRC, user mailing list, Stack Overflow).


⚙️

v

^

0

v



Patrick Callahan

Tuesday at 19:45

I would like to do something like this because I would rather encode as much information about how the build is to be done in the Jenkinsfile as possible-- the only thing I really want to specify in the job itself is the location of the repository.

My concern is not branch-specific behavior, but certain behaviors (like checking out to a local branch) that we'll need for all of our projects because our build tools expect it.


Is it still considered a 'workaround' to be 'avoided' in my case? Should I prefer configuring the SCM through the web interface, for some reason? Is it 'not safe' to access the properties of the `scm` variable in this way?



^

0

v



Allan Burdajewicz

Patrick,

Since GitHub Branch Source 2.2.0 / Bitbucket Branch Source 2.2.0, such advanced behaviors - now known as *SCM Traits* are configurable at the organization / multibranch level in the UI. You should not need to fiddle with the *checkout* step anymore.

Yesterday at 19:39

Please sign in to leave a comment.