Jenkins environment variables

Mukesh Singal Mar 3, 2018 · 3 min read

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

Jenkins exposes characteristics of components using Environment variables. This blog will tell you everything about Environment variables.

1. Creating Global environment variables
2. Access Global environment variables.
3. Creating local environment variables during build.

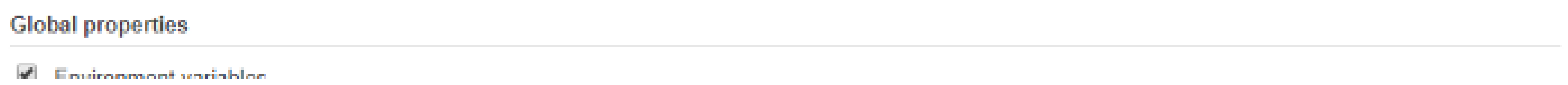
1. Creating global environment variables:

1.1 Using Jenkins Console

we can easily create global environment variables.

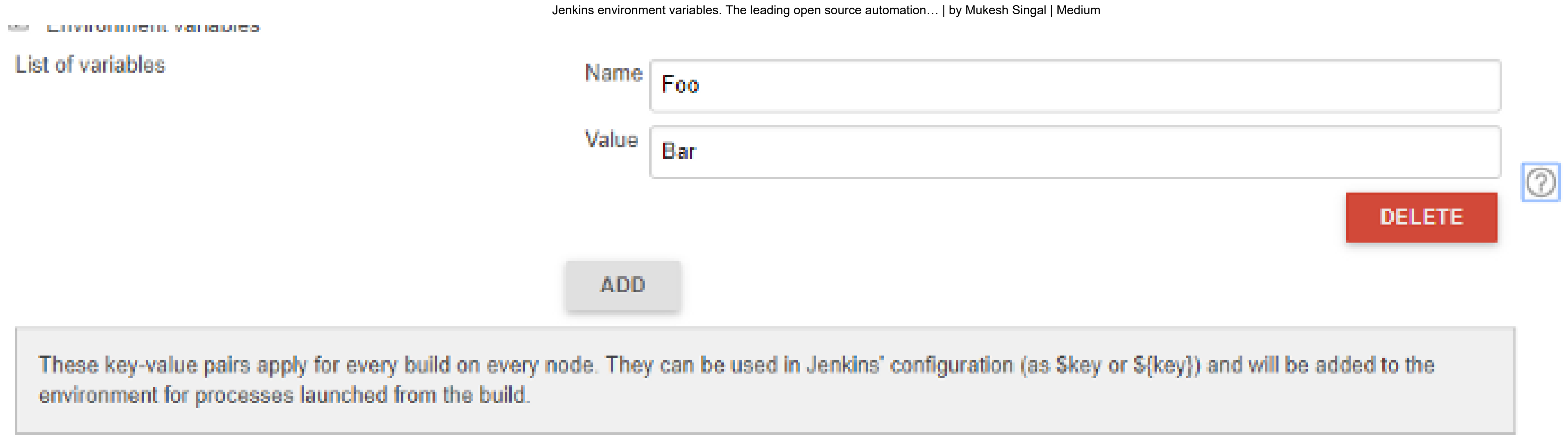
> *Manage Jenkins => Configure*

Under section **Global Properties,** check **Environment variables** checkbox. Now Jenkins allow us to add key and value pairs which will be accessible with in each node and each build.



Global properties

☑ Environment variables



Environment variables

List of variables

Name Foo

Value Bar

These key-value pairs apply for every build on every node. They can be used in Jenkins' configuration (as $key or ${key}) and will be added to the environment for processes launched from the build.

1.2 Using Groovy/Java

To create global environment variable, we need to use below code.

```
import hudson.EnvVars;
import hudson.slaves.EnvironmentVariablesNodeProperty;
import hudson.slaves.NodeProperty;
import hudson.slaves.NodePropertyDescriptor;
import hudson.util.DescribableList;
import jenkins.model.Jenkins;


public createGlobalEnvironmentVariables(String key, String value){

        Jenkins instance = Jenkins.getInstance();

        DescribableList<NodeProperty<?>, NodePropertyDescriptor>
globalNodeProperties = instance.getGlobalNodeProperties();
        List<EnvironmentVariablesNodeProperty>
envVarsNodePropertyList =
globalNodeProperties.getAll(EnvironmentVariablesNodeProperty.class);

        EnvironmentVariablesNodeProperty newEnvVarsNodeProperty =
null;
        EnvVars envVars = null;

        if ( envVarsNodePropertyList == null ||
envVarsNodePropertyList.size() == 0 ) {
            newEnvVarsNodeProperty = new
hudson.slaves.EnvironmentVariablesNodeProperty();
            globalNodeProperties.add(newEnvVarsNodeProperty);
            envVars = newEnvVarsNodeProperty.getEnvVars();
        } else {
            envVars = envVarsNodePropertyList.get(0).getEnvVars();
        }
        envVars.put(key, value)
        instance.save()
}


createGlobalEnvironmentVariables('Var1','DummyValue')
```

2. Access Global Environment variables

2.1 In jenkins Pipeline

To use above defined variable in pipeline we can use env.Key give us access to out jenkins global variables.

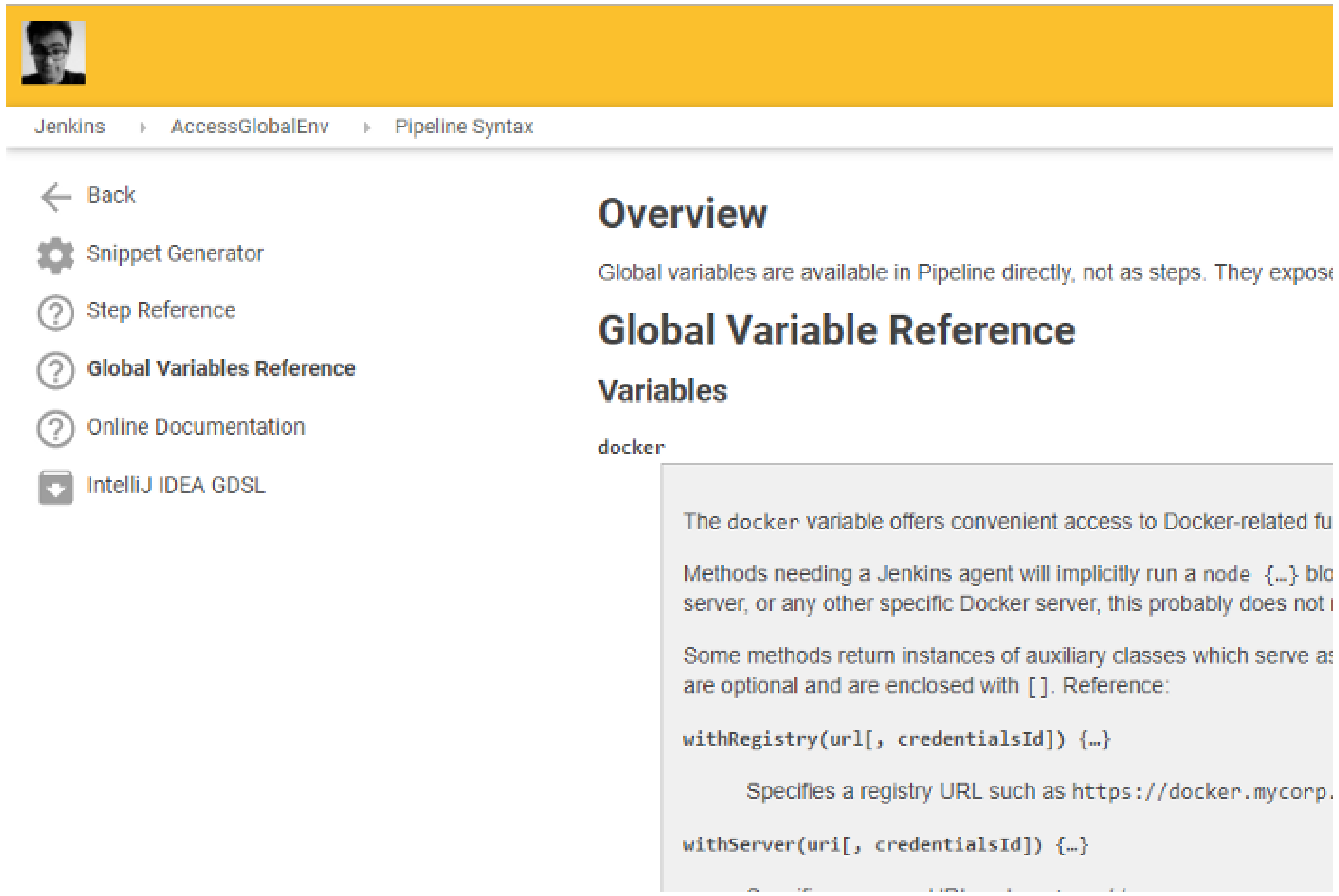
```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                echo env.Var1
            }
        }
    }
}
```

In jenkins pipeline there are lot of other useful environment variables which can be accessed during build execution.

Most useful environment variables. Highly recommended to check out.

1. **env**: Environment variables are accessible from Groovy code as `env.VARNAME` or simply as `VARNAME`. You can write to such properties as well (only using the `env.` prefix):
2. **currentBuild:** The `currentBuild` variable may be used to refer to the currently running build.
3. **params**: Exposes all parameters defined in the build as a read-only map with variously typed values.
4. **docker** : The `docker` variable offers convenient access to Docker-related functions from a Pipeline script.

In pipeline job -> pipeline Syntax -> Global Variable Reference



Jenkins › AccessGlobalEnv › Pipeline Syntax

Overview

Global variables are available in Pipeline directly, not as steps. They expose

Global Variable Reference

Variables

docker

The docker variable offers convenient access to Docker-related fu

Methods needing a Jenkins agent will implicitly run a node {…} blo
server, or any other specific Docker server, this probably does not

Some methods return instances of auxiliary classes which serve a
are optional and are enclosed with []. Reference:

withRegistry(url[, credentialsId]) {…}

Specifies a registry URL such as https://docker.mycorp.

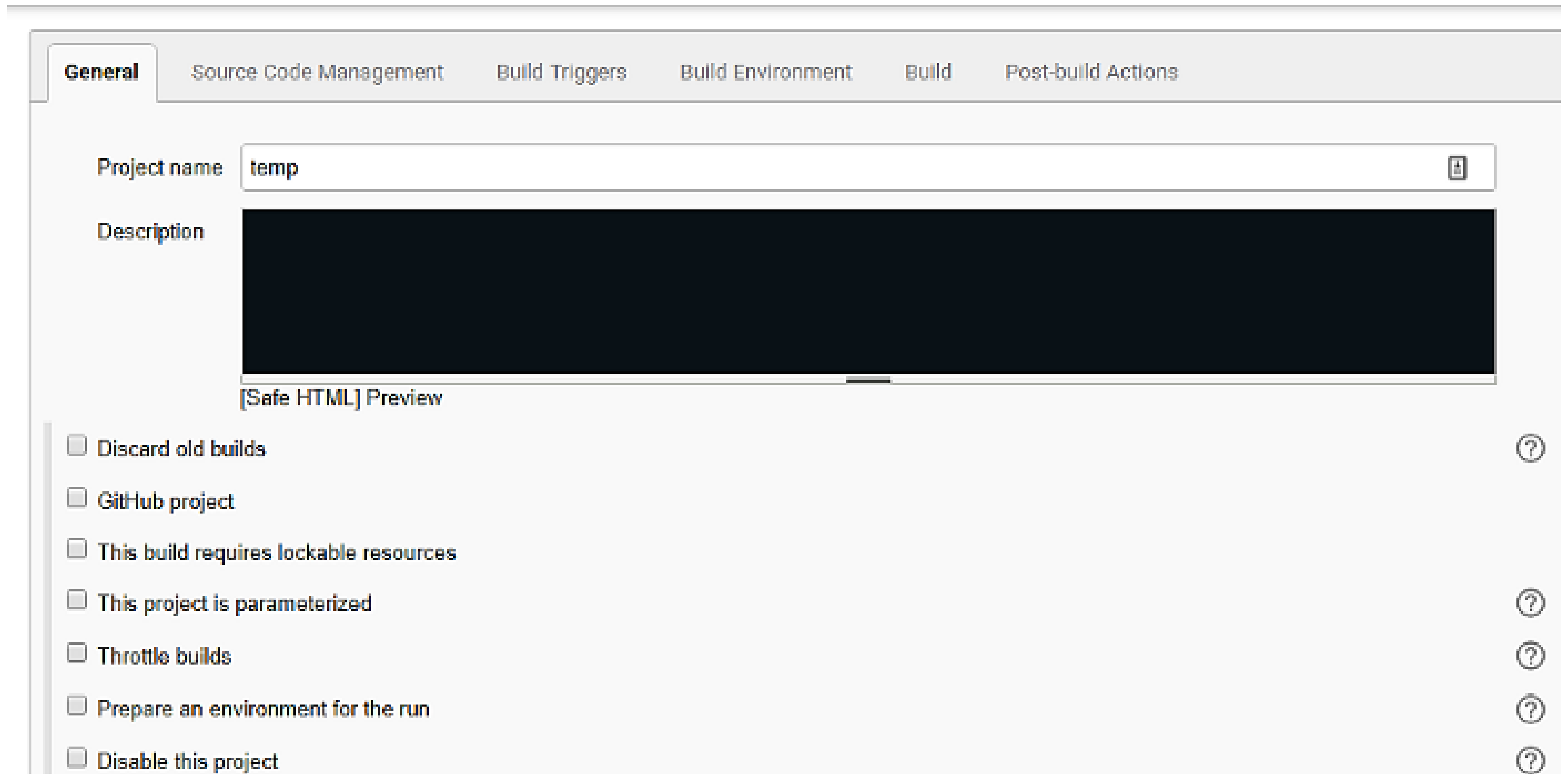
withServer(uri[, credentialsId]) {…}

2.2 In Shell/Batch Script

In Shell script we can use environment variables using $Key or ${Key}. Similarly in batch we can use %Key% to access Environment Variables.

2.3 In FreeStyle projects

In freestyle project we can use EnvEnject Plugin.

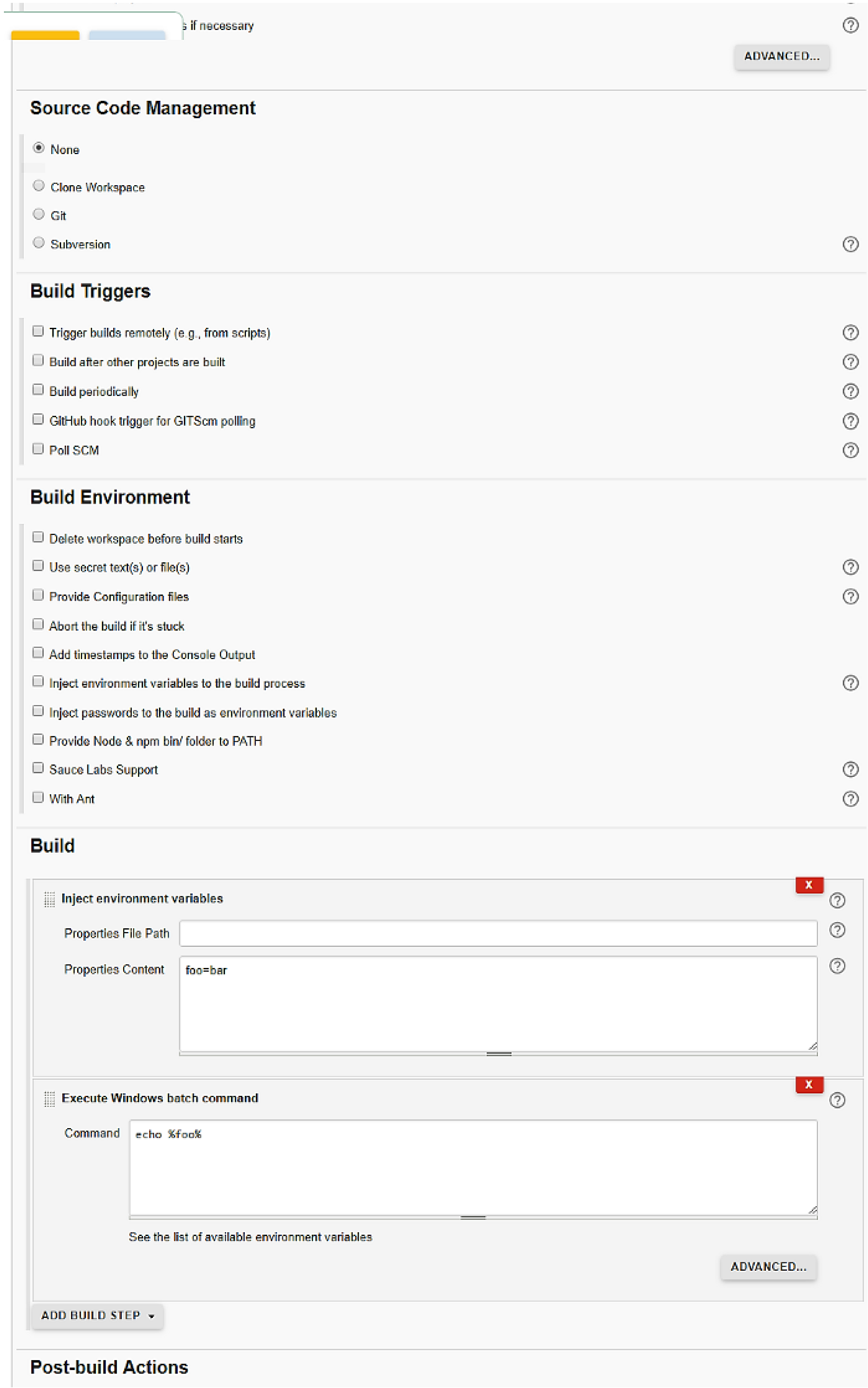


Project name: temp

Description: [redacted]

[Safe HTML] Preview

□ Discard old builds
□ GitHub project
□ This build requires lockable resources
□ This project is parameterized
□ Throttle builds
□ Prepare an environment for the run
□ Disable this project



s if necessary

ADVANCED...

Source Code Management

● None

○ Clone Workspace

○ Git

○ Subversion

Build Triggers

□ Trigger builds remotely (e.g., from scripts)

□ Build after other projects are built

□ Build periodically

□ GitHub hook trigger for GITScm polling

□ Poll SCM

Build Environment

□ Delete workspace before build starts

□ Use secret text(s) or file(s)

□ Provide Configuration files

□ Abort the build if it's stuck

□ Add timestamps to the Console Output

□ Inject environment variables to the build process

□ Inject passwords to the build as environment variables

□ Provide Node & npm bin/ folder to PATH

□ Sauce Labs Support

□ With Ant

Build

Inject environment variables

Properties File Path

Properties Content

```
foo=bar
```

Execute Windows batch command

Command

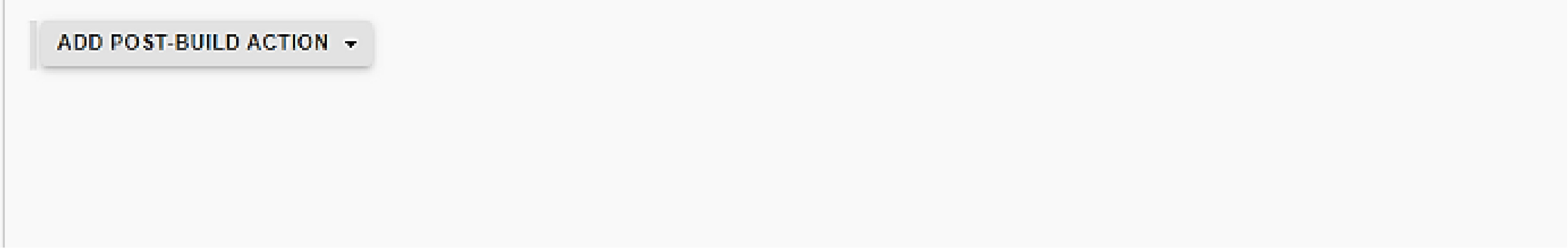
```
echo %foo%
```

See the list of available environment variables

ADVANCED...

ADD BUILD STEP ▾

Post-build Actions



3. Creating local environment variables during build.

3.1 Using Declarative pipeline

Jenkinsfile (Declarative Pipeline)

```
pipeline {
    agent any
    environment {
        DISABLE_AUTH = 'true'
    }
    stages {
        stage('Build') {
            steps {
                echo env.DISABLE_AUTH
            }
        }
    }
}
```

3.2 Using Scripted pipeline.

```
node{
    stage('Build') {
        withEnv(["DISABLE_AUTH=true"]) {
          echo env.DISABLE_AUTH
        }
    }
}
```