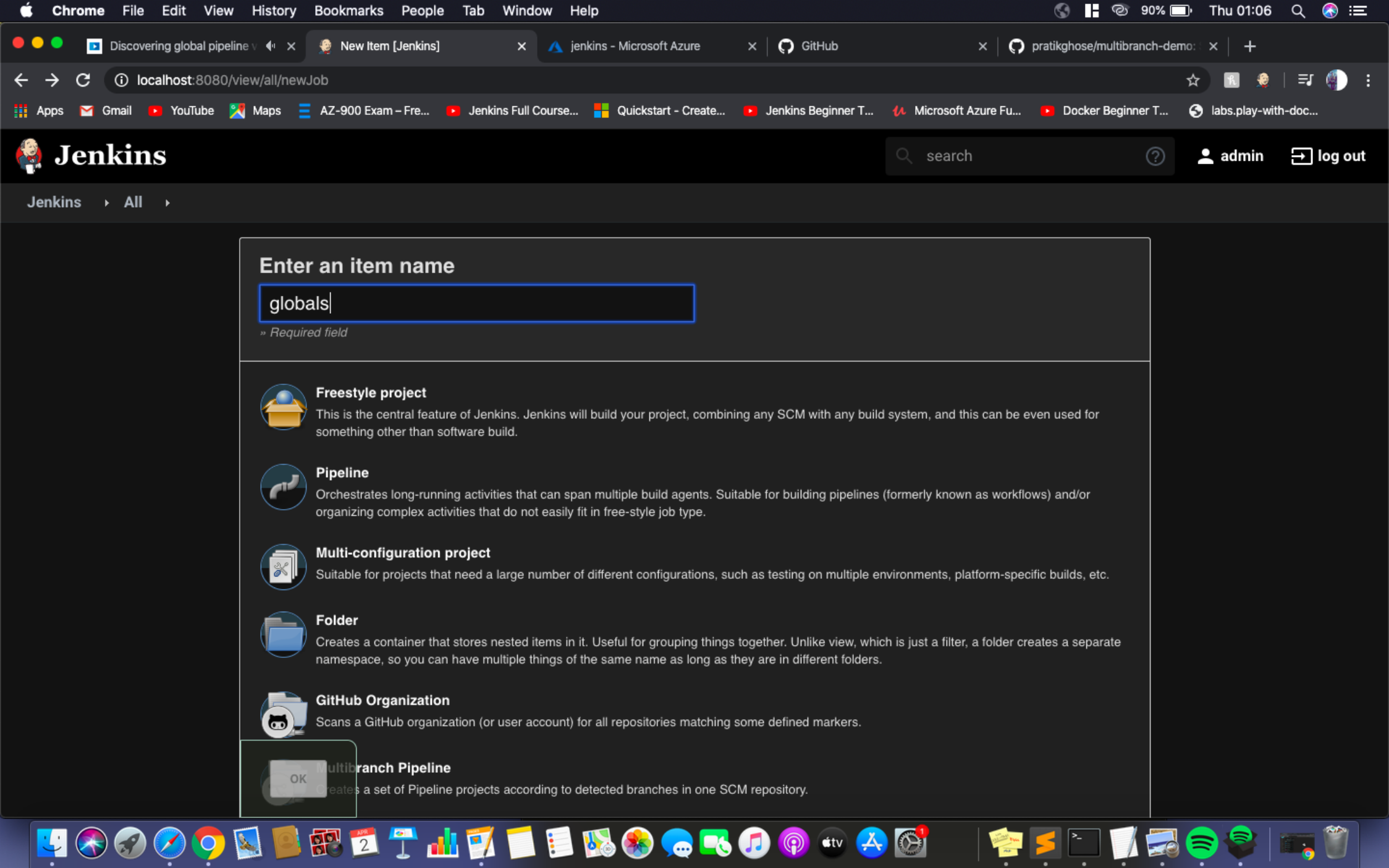
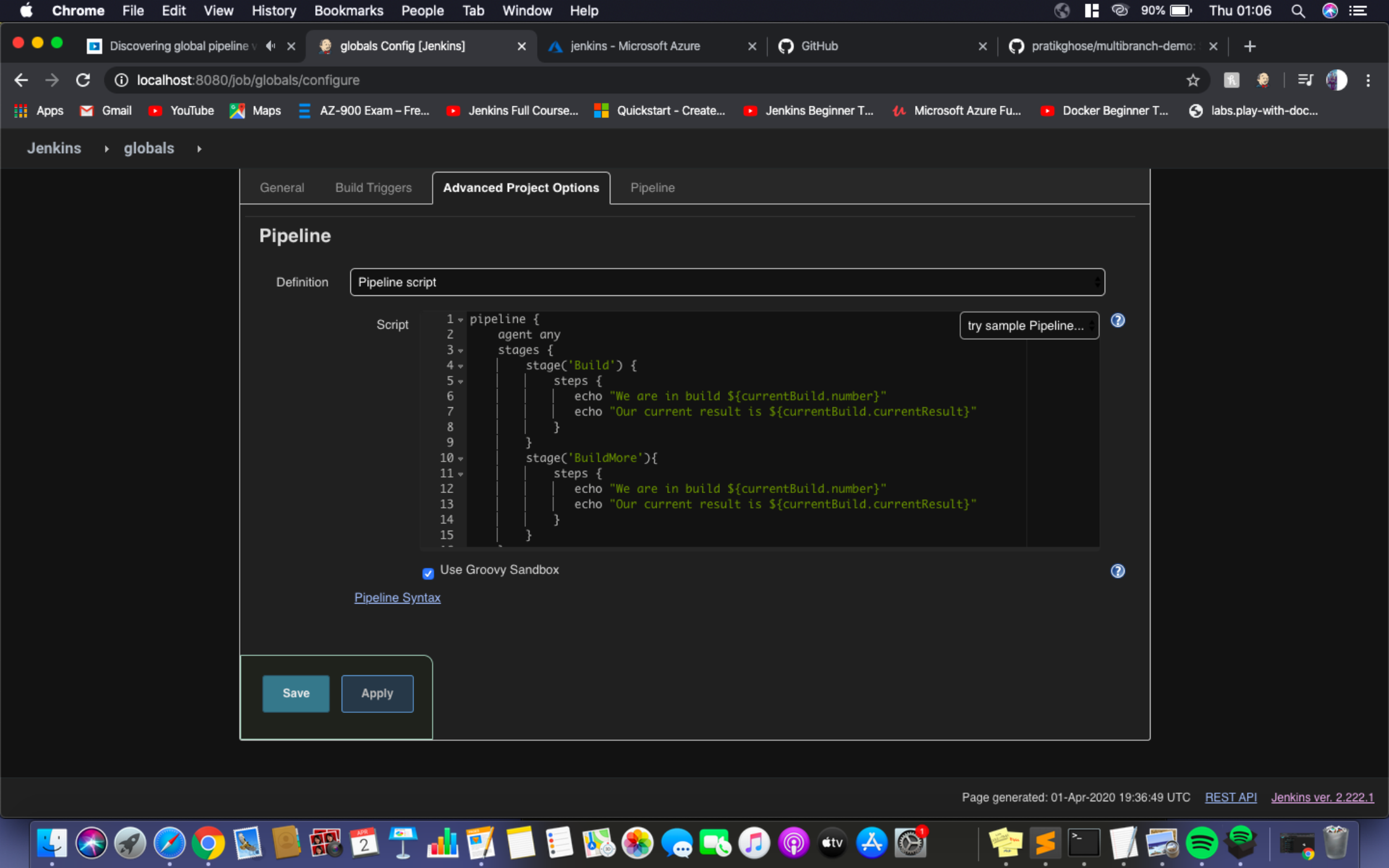
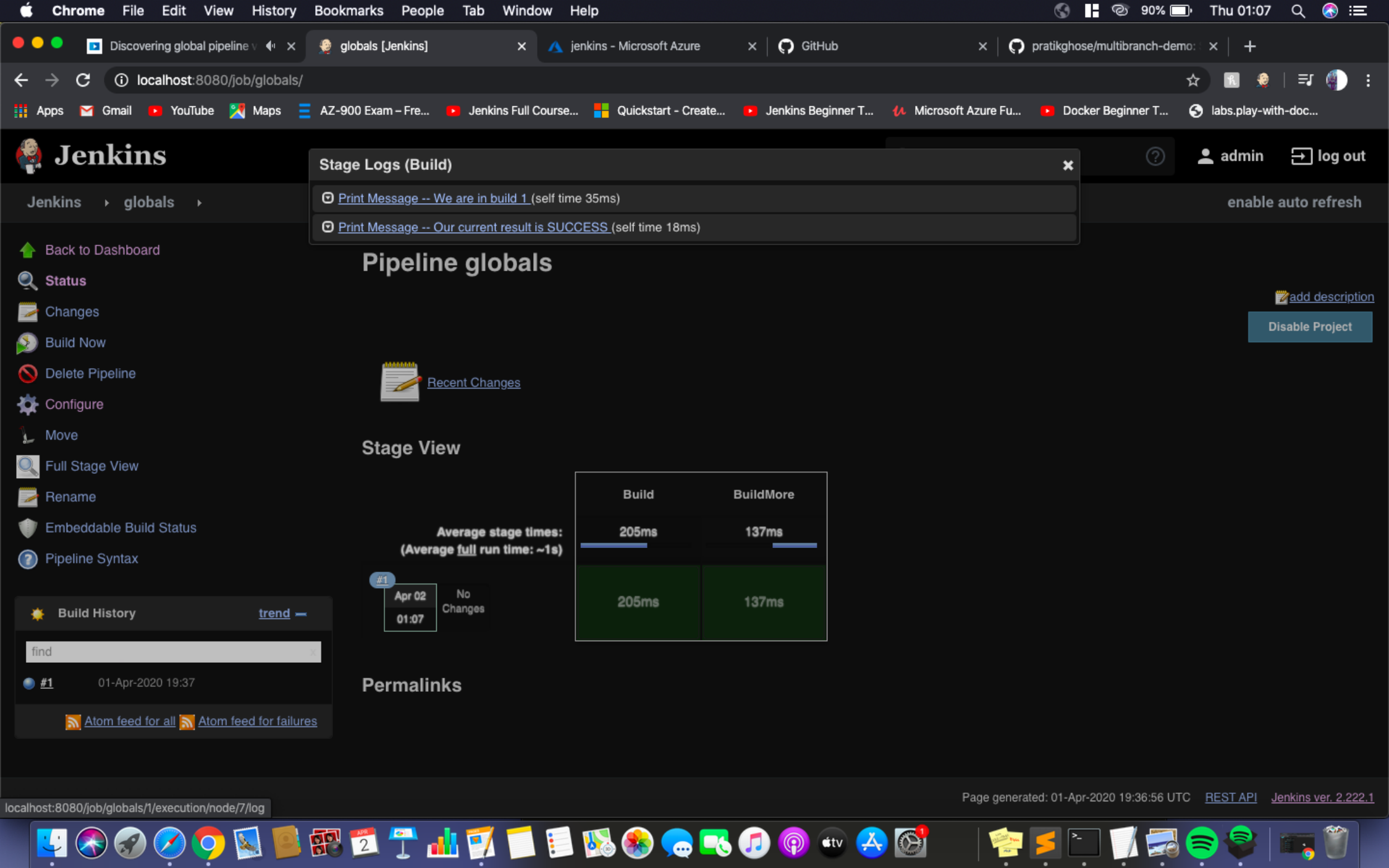
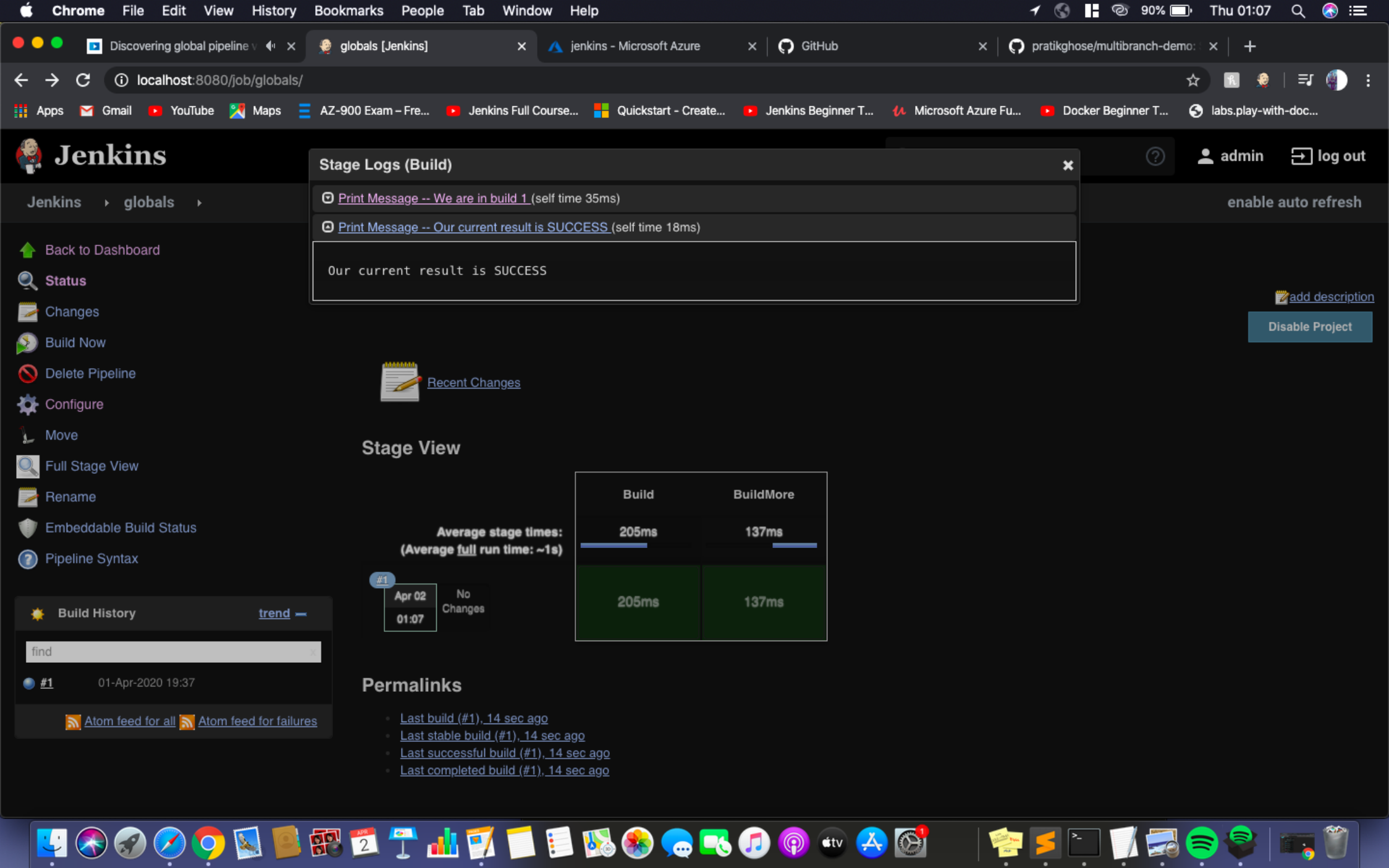
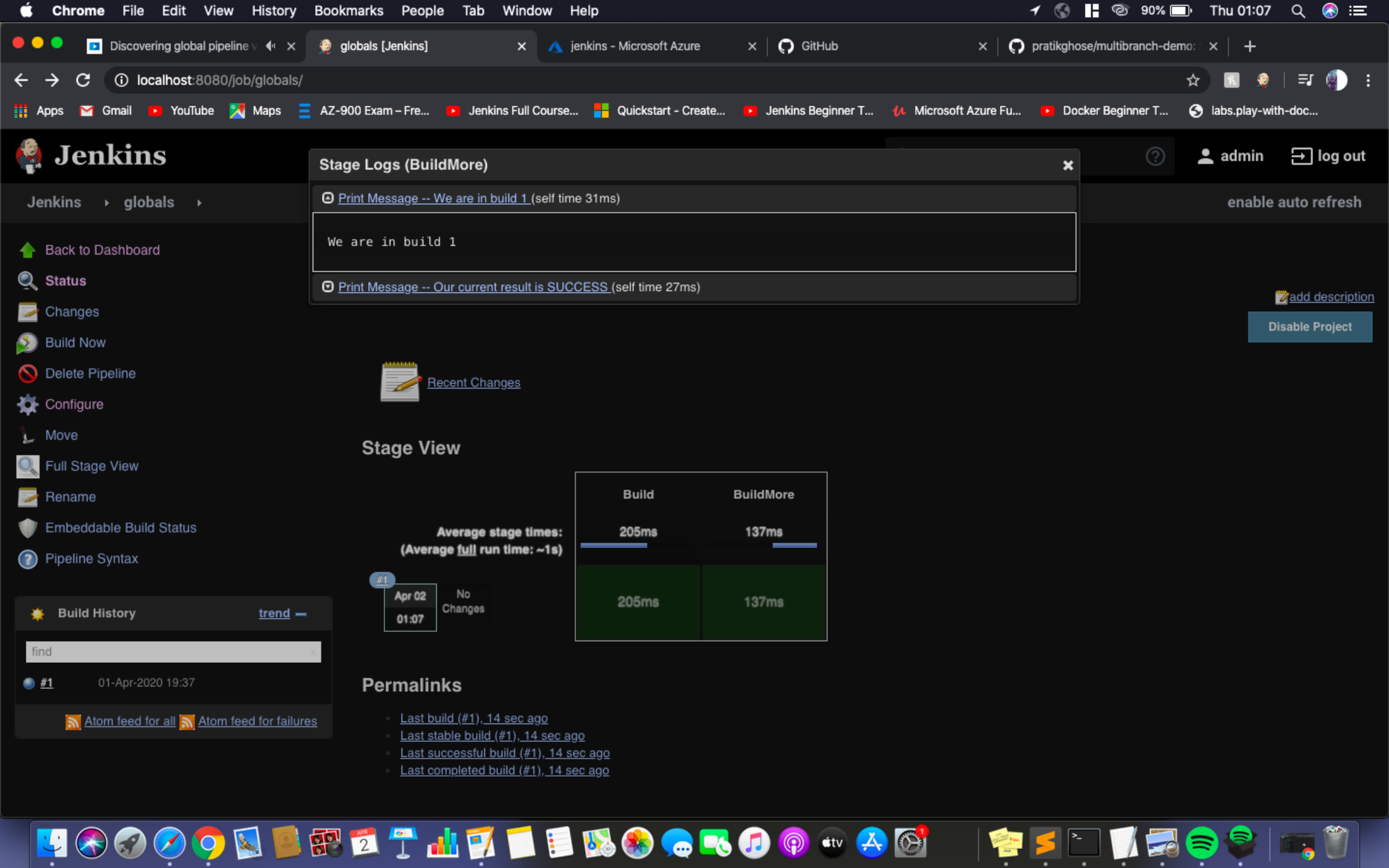

Chrome File Edit View History Bookmarks People Tab Window Help
90% Thu 01:06
Discovering global pipeline v
New Item [Jenkins]
jenkins - Microsoft Azure
GitHub
pratikghose/multibranch-demo:
localhost:8080/view/all/newJob
Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...
Jenkins
search
admin
log out
Jenkins ▸ All ▸
Enter an item name
globals
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.
OK
ranch Pipeline
a set of Pipeline projects according to detected branches in one SCM repository.


Chrome File Edit View History Bookmarks People Tab Window Help
90% Thu 01:06
Discovering global pipeline
globals Config [Jenkins]
jenkins - Microsoft Azure
GitHub
pratikghose/multibranch-demo:
localhost:8080/job/globals/configure
Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...
Jenkins ▸ globals ▸
General
Build Triggers
Advanced Project Options
Pipeline
Pipeline
Definition
Pipeline script
Script
try sample Pipeline...
1 pipeline {
2 agent any
3 stages {
4 stage('Build') {
5 steps {
6 echo "We are in build ${currentBuild.number}"
7 echo "Our current result is ${currentBuild.currentResult}"
8 }
9 }
10 stage('BuildMore'){
11 steps {
12 echo "We are in build ${currentBuild.number}"
13 echo "Our current result is ${currentBuild.currentResult}"
14 }
15 }
Use Groovy Sandbox
Pipeline Syntax
Save
Apply
Page generated: 01-Apr-2020 19:36:49 UTC
REST API
Jenkins ver. 2.222.1


Chrome File Edit View History Bookmarks People Tab Window Help
90% Thu 01:07
Discovering global pipeline
globals [Jenkins]
jenkins - Microsoft Azure
GitHub
pratikghose/multibranch-demo:
localhost:8080/job/globals/
Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...
Jenkins
admin
log out
Stage Logs (Build)
Print Message -- We are in build 1 (self time 35ms)
Print Message -- Our current result is SUCCESS (self time 18ms)
Jenkins ▸ globals ▸
enable auto refresh
Back to Dashboard
Status
Changes
Build Now
Delete Pipeline
Configure
Move
Full Stage View
Rename
Embeddable Build Status
Pipeline Syntax
Build History
trend
find
#1 01-Apr-2020 19:37
Atom feed for all
Atom feed for failures
Pipeline globals
add description
Disable Project
Recent Changes
Stage View
Average stage times:
(Average full run time: ~1s)
#1
Apr 02
01:07
No Changes
Build
BuildMore
205ms
137ms
205ms
137ms
Permalinks
Page generated: 01-Apr-2020 19:36:56 UTC
REST API
Jenkins ver. 2.222.1
localhost:8080/job/globals/1/execution/node/7/log


Chrome File Edit View History Bookmarks People Tab Window Help
90% Thu 01:07
Discovering global pipeline
globals [Jenkins]
jenkins - Microsoft Azure
GitHub
pratikghose/multibranch-demo:
localhost:8080/job/globals/
Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...
Jenkins
admin
log out
Jenkins › globals ›
enable auto refresh
Stage Logs (Build)
Print Message -- We are in build 1 (self time 35ms)
Print Message -- Our current result is SUCCESS (self time 18ms)
Our current result is SUCCESS
Back to Dashboard
Status
Changes
Build Now
Delete Pipeline
Configure
Move
Full Stage View
Rename
Embeddable Build Status
Pipeline Syntax
add description
Disable Project
Recent Changes
Stage View
Build
BuildMore
Average stage times:
(Average full run time: ~1s)
205ms
137ms
#1
Apr 02
01:07
No Changes
205ms
137ms
Build History
trend
find
#1
01-Apr-2020 19:37
Atom feed for all
Atom feed for failures
Permalinks
Last build (#1), 14 sec ago
Last stable build (#1), 14 sec ago
Last successful build (#1), 14 sec ago
Last completed build (#1), 14 sec ago


Chrome File Edit View History Bookmarks People Tab Window Help
90% Thu 01:07
Discovering global pipeline
globals [Jenkins]
jenkins - Microsoft Azure
GitHub
pratikghose/multibranch-demo:
localhost:8080/job/globals/
Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...
Jenkins
admin
log out
Jenkins ▸ globals ▸
enable auto refresh
Stage Logs (BuildMore)
Print Message -- We are in build 1 (self time 31ms)
We are in build 1
Print Message -- Our current result is SUCCESS (self time 27ms)
Back to Dashboard
Status
Changes
Build Now
Delete Pipeline
Configure
Move
Full Stage View
Rename
Embeddable Build Status
Pipeline Syntax
add description
Disable Project
Recent Changes
Stage View
Average stage times:
(Average full run time: ~1s)
Build
BuildMore
205ms
137ms
205ms
137ms
#1
Apr 02
01:07
No Changes
Build History
trend
find
#1
01-Apr-2020 19:37
Atom feed for all
Atom feed for failures
Permalinks
Last build (#1), 14 sec ago
Last stable build (#1), 14 sec ago
Last successful build (#1), 14 sec ago
Last completed build (#1), 14 sec ago

Chrome File Edit View History Bookmarks People Tab Window Help 90% Thu 01:07

Discovering global pipeline v | globals [Jenkins] | jenkins - Microsoft Azure | GitHub | pratikghose/multibranch-demo:

localhost:8080/job/globals/

Apps Gmail YouTube Maps AZ-900 Exam – Fre... Jenkins Full Course... Quickstart - Create... Jenkins Beginner T... Microsoft Azure Fu... Docker Beginner T... labs.play-with-doc...

Jenkins

admin log out

Jenkins › globals ›

enable auto refresh

- Back to Dashboard
- Status
- Changes
- Build Now
- Delete Pipeline
- Configure
- Move
- Full Stage View
- Rename
- Embeddable Build Status
- Pipeline Syntax

Build History trend

find

#1 01-Apr-2020 19:37

Atom feed for all Atom feed for failures

Stage Logs (BuildMore)

Print Message -- We are in build 1 (self time 31ms)

Print Message -- Our current result is SUCCESS (self time 27ms)

```
Our current result is SUCCESS
```

add description

Disable Project

Recent Changes

Stage View

	Build	BuildMore
Average stage times: (Average full run time: ~1s)	205ms	137ms
#1 Apr 02 01:07 No Changes	205ms	137ms

Permalinks

- Last build (#1), 14 sec ago
- Last stable build (#1), 14 sec ago
- Last successful build (#1), 14 sec ago
- Last completed build (#1), 14 sec ago

Back
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL

Overview

Global variables are available in Pipeline directly, not as steps. They expose methods and variables to be accessed within your Pipeline script.

Global Variable Reference

Variables

`docker`

The `docker` variable offers convenient access to Docker-related functions from a Pipeline script.

Methods needing a Jenkins agent will implicitly run a `node {…}` block if you have not wrapped them in one. It is a good idea to enclose a block of steps which should all run on the same node in such a block yourself. (If using a Swarm server, or any other specific Docker server, this probably does not matter, but if you are using the default server on localhost it likely will.)

Some methods return instances of auxiliary classes which serve as holders for an ID and which have their own methods and properties. Methods taking a body return any value returned by the body itself. Some method parameters are optional and are enclosed with `[]`. Reference:

`withRegistry(url[, credentialsId]) {…}`

Specifies a registry URL such as `https://docker.mycorp.com/`, plus an optional credentials ID to connect to it.

`withServer(uri[, credentialsId]) {…}`

Specifies a server URI such as `tcp://swarm.mycorp.com:2376`, plus an optional credentials ID to connect to it.

`withTool(toolName) {…}`

Specifies the name of a Docker installation to use, if any are defined in Jenkins global configuration. If unspecified, `docker` is assumed to be in the `$PATH` of the Jenkins agent.

`image(id)`

Creates an `Image` object with a specified name or ID. See below.

`build(image[, args])`

Runs `docker build` to create and tag the specified image from a `Dockerfile` in the current directory. Additional `args` may be added, such as `'-f Dockerfile.other --pull --build-arg http_proxy=http://192.168.1.1:3128 .'`. Like `docker build`, `args` must end with the build context.

Container.id

Hexadecimal ID of a running container.

Container.stop

Runs `docker stop` and `docker rm` to shut down a container and remove its storage.

Container.port(port)

Runs `docker port` on the container to reveal how the port `port` is mapped on the host.

pipeline

The `pipeline` step allows you to define your Pipelines in a more structured way. See the wiki for more information.

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):

```
env.MYTOOL_VERSION = '1.33'
node {
  sh '/usr/local/mytool-$MYTOOL_VERSION/bin/start'
}
```

These definitions will also be available via the REST API during the build or after its completion, and from upstream Pipeline builds using the `build` step.

However any variables set this way are global to the Pipeline build. For variables with node-specific content (such as file paths), you should instead use the `withEnv` step, to bind the variable only within a `node` block.

A set of environment variables are made available to all Jenkins projects, including Pipelines. The following is a general list of variable names that a

BRANCH_NAME
For a multibranch project, this will be set to the name of the branch being built, for example in case you wish to deploy to production from ma
feature branches; if corresponding to some kind of change request, the name is generally arbitrary (refer to `CHANGE_ID` and `CHANGE_TARGE`

CHANGE_ID

```
}
```

These definitions will also be available via the REST API during the build or after its completion, and from upstream Pipeline builds using the `build` step.

However any variables set this way are global to the Pipeline build. For variables with node-specific content (such as file paths), you should instead use the `withEnv` step, to bind the variable only within a `node` block.

A set of environment variables are made available to all Jenkins projects, including Pipelines. The following is a general list of variable names that are available.

BRANCH_NAME
: For a multibranch project, this will be set to the name of the branch being built, for example in case you wish to deploy to production from `master` but not from feature branches; if corresponding to some kind of change request, the name is generally arbitrary (refer to `CHANGE_ID` and `CHANGE_TARGET`).

CHANGE_ID
: For a multibranch project corresponding to some kind of change request, this will be set to the change ID, such as a pull request number, if supported; else unset.

CHANGE_URL
: For a multibranch project corresponding to some kind of change request, this will be set to the change URL, if supported; else unset.

CHANGE_TITLE
: For a multibranch project corresponding to some kind of change request, this will be set to the title of the change, if supported; else unset.

CHANGE_AUTHOR
: For a multibranch project corresponding to some kind of change request, this will be set to the username of the author of the proposed change, if supported; else unset.

CHANGE_AUTHOR_DISPLAY_NAME
: For a multibranch project corresponding to some kind of change request, this will be set to the human name of the author, if supported; else unset.

CHANGE_AUTHOR_EMAIL
: For a multibranch project corresponding to some kind of change request, this will be set to the email address of the author, if supported; else unset.

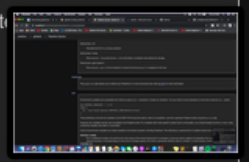
CHANGE_TARGET
: For a multibranch project corresponding to some kind of change request, this will be set to the target or base branch to which the change could be merged, if supported; else unset.

CHANGE_BRANCH
: For a multibranch project corresponding to some kind of change request, this will be set to the name of the actual head on the source control system which may or may not be different from `BRANCH_NAME`. For example in GitHub or Bitbucket this would have the name of the origin branch whereas `BRANCH_NAME` would be something like `PR-24`.

CHANGE_FORK
: For a multibranch project corresponding to some kind of change request, this will be set to the name of the forked repo if the change originat
unset.

BUILD_NUMBER
: The current build number, such as "153"

BUILD_ID
: The current build ID, identical to BUILD_NUMBER for builds created in 1.597+, but a YYYY-MM-DD_hh-mm-ss timestamp for older builds



WORKSPACE
The absolute path of the directory assigned to the build as a workspace.
WORKSPACE_TMP
A temporary directory near the workspace that will not be browsable and will not interfere with SCM checkouts. May not initially exist, so be sure to create the directory as needed (e.g., `mkdir -p` on Linux).
JENKINS_HOME
The absolute path of the directory assigned on the master node for Jenkins to store data.
JENKINS_URL
Full URL of Jenkins, like `http://server:port/jenkins/` (note: only available if *Jenkins URL* set in system configuration)
BUILD_URL
Full URL of this build, like `http://server:port/jenkins/job/foo/15/` (*Jenkins URL* must be set)
JOB_URL
Full URL of this job, like `http://server:port/jenkins/job/foo/` (*Jenkins URL* must be set)

SCM-specific variables such as `GIT_COMMIT` are not automatically defined as environment variables; rather you can use the return value of the `checkout` step.

As an example of loading variable values from Groovy:

```
mail to: 'devops@acme.com',
    subject: "Job '${JOB_NAME}' (${BUILD_NUMBER}) is waiting for input",
    body: "Please go to ${BUILD_URL} and verify the build"
```

params

Exposes all parameters defined in the build as a read-only map with variously typed values. Example:

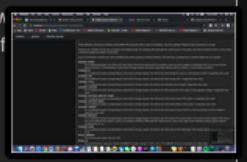
```
if (params.BOOLEAN_PARAM_NAME) {doSomething()}
```

or to supply a nontrivial default value:

```
if (params.getOrDefault('BOOLEAN_PARAM_NAME', true)) {doSomething()}
```

Note for multibranch (`Jenkinsfile`) usage: the `properties` step allows you to define job properties, but these take effect when the step is run, w
parameter definitions are generally consulted before the build begins. As a convenience, any parameters *currently* defined in the job which have def
listed in this map. That allows you to write, for example:

```
properties([parameters([string(name: 'BRANCH', defaultValue: 'master')])])
git url: '…', branch: params.BRANCH
```



or to supply a nontrivial default value:

```
if (params.getOrDefault('BOOLEAN_PARAM_NAME', true)) {doSomething()}
```

Note for multibranch (`Jenkinsfile`) usage: the `properties` step allows you to define job properties, but these take effect when the step is run, whereas build parameter definitions are generally consulted before the build begins. As a convenience, any parameters *currently* defined in the job which have default values will also be listed in this map. That allows you to write, for example:

```
properties([parameters([string(name: 'BRANCH', defaultValue: 'master')])])
git url: '…', branch: params.BRANCH
```

and be assured that the `master` branch will be checked out even in the initial build of a branch project, or if the previous build did not specify parameters or used a different parameter name.

currentBuild

The `currentBuild` variable, which is of type RunWrapper, may be used to refer to the currently running build. It has the following readable properties:

`getBuildCauses`
: Returns a JSON array of build causes for the current build

EXPERIMENTAL - MAY CHANGE `getBuildCauses(String causeClass)`
: Takes a string representing the fully qualified `Cause` class and returns a `JSON array` of build causes filtered by that type for the current build, or an empty `JSON array` if no causes of the specified type apply to the current build

`number`
: build number (integer)

`result`
: typically `SUCCESS`, `UNSTABLE`, or `FAILURE` (*may* be null for an ongoing build)

`currentResult`
: typically `SUCCESS`, `UNSTABLE`, or `FAILURE`. Will never be null.

`resultIsBetterOrEqualTo(String)`
: Compares the current build result to the provided result string (`SUCCESS`, `UNSTABLE`, or `FAILURE`) and returns true if the current build result is better than or equal to the provided result.

`resultIsWorseOrEqualTo(String)`
: Compares the current build result to the provided result string (`SUCCESS`, `UNSTABLE`, or `FAILURE`) and returns true if the current build result is worse than or equal to the provided result.

`displayName`
: normally `#123` but sometimes set to, e.g., an SCM commit identifier.

`fullDisplayName`

next build of the project, or null

absoluteUrl
URL of build index page

buildVariables
for a non-Pipeline downstream build, offers access to a map of defined build variables; for a Pipeline downstream build, any variables set globally on `env` at the time the build ends. Child Pipeline jobs can use this to report additional information to the parent job by setting variables in `env`. Note that build parameters are *not* shown in `buildVariables`.

changeSets
a list of changesets coming from distinct SCM checkouts; each has a `kind` and is a list of commits; each commit has a `commitId`, `timestamp`, `msg`, `author`, and `affectedFiles` each of which has an `editType` and `path`; the value will not generally be `Serializable` so you may only access it inside a method marked `@NonCPS`

upstreamBuilds
a list of upstream builds. These are the builds of the upstream projects whose artifacts feed into this build.

rawBuild
a `hudson.model.Run` with further APIs, only for trusted libraries or administrator-approved scripts outside the sandbox; the value will not be `Serializable` so you may only access it inside a method marked `@NonCPS`

keepLog
`true` if the log file for this build should be kept and not deleted.

Additionally, for this build only (but not for other builds), the following properties are writable:

- **result**
- **displayName**
- **description**
- **keepLog**

scm

Represents the SCM configuration in a multibranch project build. Use `checkout scm` to check out sources matching `Jenkinsfile`.
You may also use this in a standalone project configured with *Pipeline from SCM*, though in that case the checkout will just be of the latest revision in the branch, possibly newer than the revision from which the Pipeline was loaded.