**Chapter 9. The Blue Ocean Interface**

As well as the new Declarative Pipeline syntax, one of the key innovations in Jenkins 2 is the new graphical interface, Blue Ocean. At a high level, we can summarize the features of Blue Ocean as follows:

* Provides a graphical representation of pipeline processing
* Provides a graphical interface for creating new Declarative Pipelines
* Provides a more segmented view of pipeline processing at the level of the stages in the pipeline, including being able to drill into logs at that level
* Supports views by branches for Multibranch Pipeline projects
* Supports working with pull requests for Multibranch Pipeline projects
* Provides a guided setup for new pipelines from source management repositories
* Provides a pipeline editor based on adding stages, steps, etc., through a combination of interaction with graphical (point-and-click) elements and typing
* Can better represent parallel stages in comparison to the Stage View output
* Provides links back to the “classic” (legacy) Jenkins view for corresponding items or those that do not have a custom Blue Ocean representation

The interface keys off of the stage definitions in the pipeline and adds graphical elements to represent each stage. Those representations include circular icons and colors to represent the processing progress and the resulting states of success and failure.

You can also view logs segmented by steps and click through to get more details.

That’s the very high-level view of the new interface. As with the traditional Jenkins interface, it’s easiest to understand the available functionality by presenting the various screens and options with example jobs.

The remainder of this chapter is divided into two parts. Part 1 takes you through the various screens, pages, and views associated with managing existing pipelines that are being executed. Part 2 takes you through working with the pipeline editor to create, edit, and debug pipelines. Together, both parts will provide you with a well-rounded understanding of Blue Ocean.

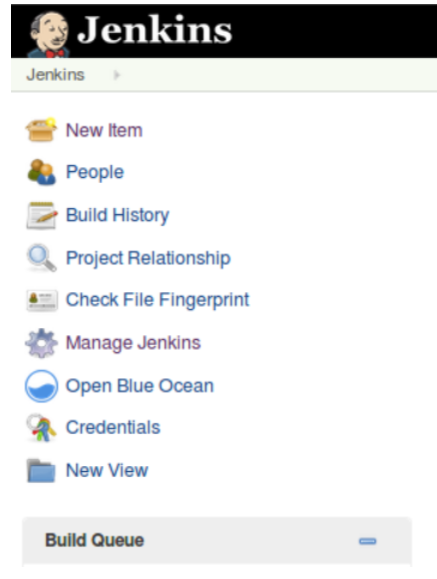
# Part 1: Managing Existing Pipelines

In this first part of the chapter, we’ll cover using the Blue Ocean interface to see how it handles the execution and output of existing pipelines. The easiest way to do this is to walk through the different screens you will be exposed to, and discuss the functionality and features of each.

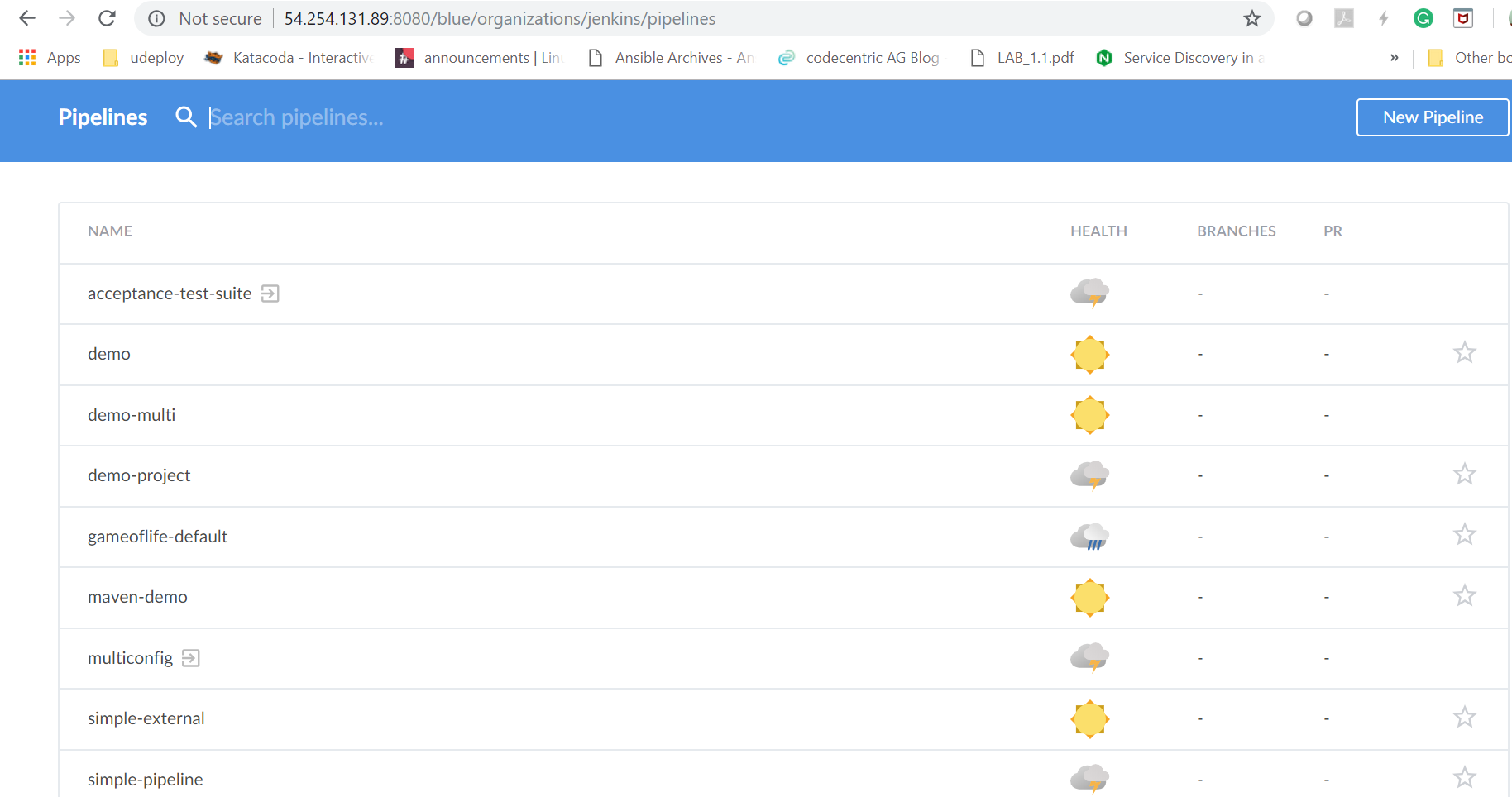
We’ll start where we always start with Jenkins: the dashboard.

## The Dashboard

The main menu on the lefthand side of the Jenkins dashboard contains a menu option to launch the Blue Ocean interface



You can also launch the interface directly by entering the URL in your browser. The shortest version is <your Jenkins URL>/blue. Either the menu item or the URL will open up the Blue Ocean dashboard, as shown in



Like the traditional Jenkins dashboard, this page lists your Jenkins jobs. Although it’s focused on Pipeline projects, all of your jobs will show up here.

To understand this page completely, let’s discuss the various navigational links and elements available on it.

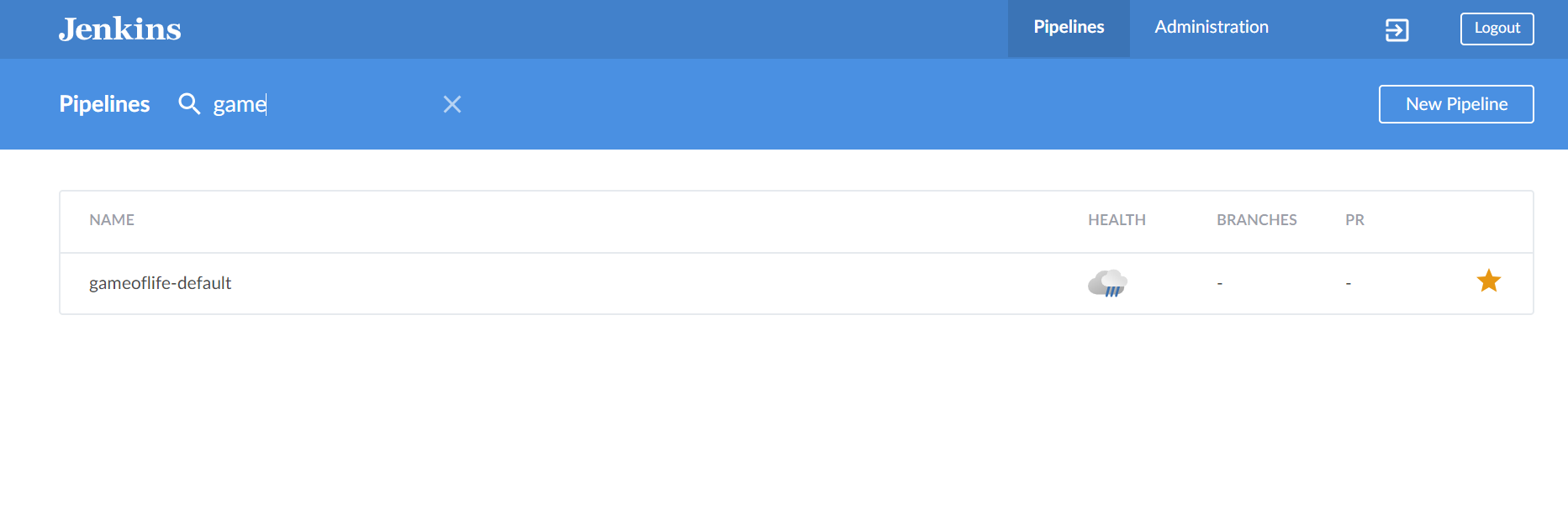
In the top blue bar, the terms “Jenkins” and “Pipelines” are links that take you to this same page. They are useful in certain cases—for example, if you have done an operation such as a search that filters the job list, and you then want to get back to the full list.

The Administration item in the same row links to the traditional Manage Jenkins page for administering settings for the Jenkins instance.

The square icon with the arrow pointing to the right takes you back to the classic Jenkins dashboard, and the Logout button should be self-explanatory.

In the next row, the Pipelines link serves the same purpose as the Pipelines link in the row above it that we just mentioned.

Next to that, the magnifying glass is a search function (as you might expect). Clicking on it allows you to type in an expression to search for among the names of the listed pipelines. For example, as seen in [Figure 9-3](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_use_search_fun_blueOdash), if we click in the search area and enter an “o,” the list will change to showing only those projects that have an “o” in their name.



The New Pipeline button can be used to create a new pipeline—we’ll cover that functionality in another section of the chapter.

Underneath the top blue rows, you have the main part of the page that lists the projects/jobs currently defined in the Jenkins instance. The fields for each project row are described in

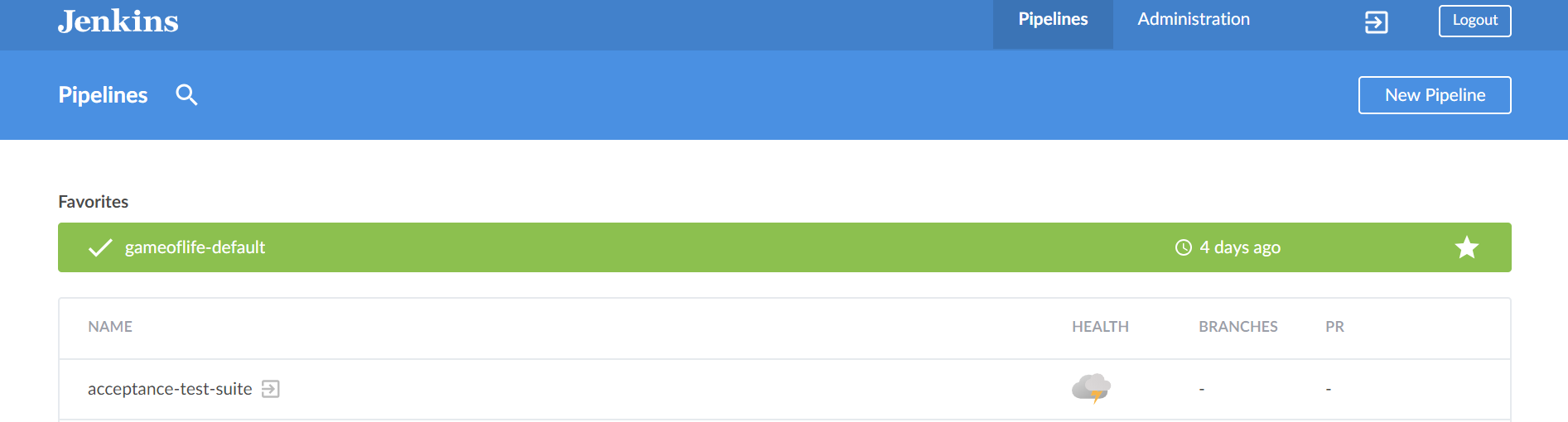


The name and health indicators are the same as in the classic Jenkins view.

The Branches column only applies to the new Multibranch Pipeline project type. This provides a summary of the last run for the set of branches. (Multibranch Pipeline projects are discussed in detail in [Chapter 8](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch08.html#CH_Understanding_Project_Types).)

The PR column only applies if there are active pull requests (PRs), such as for a GitHub-based project. It shows the number of outstanding PRs if they exist. (PRs are covered in more detail later in this chapter.)

The last (unnamed) column allows you to choose to make a project a “favorite.” In this case, that means creating a shortcut to that project under a Favorites section at the top of the page

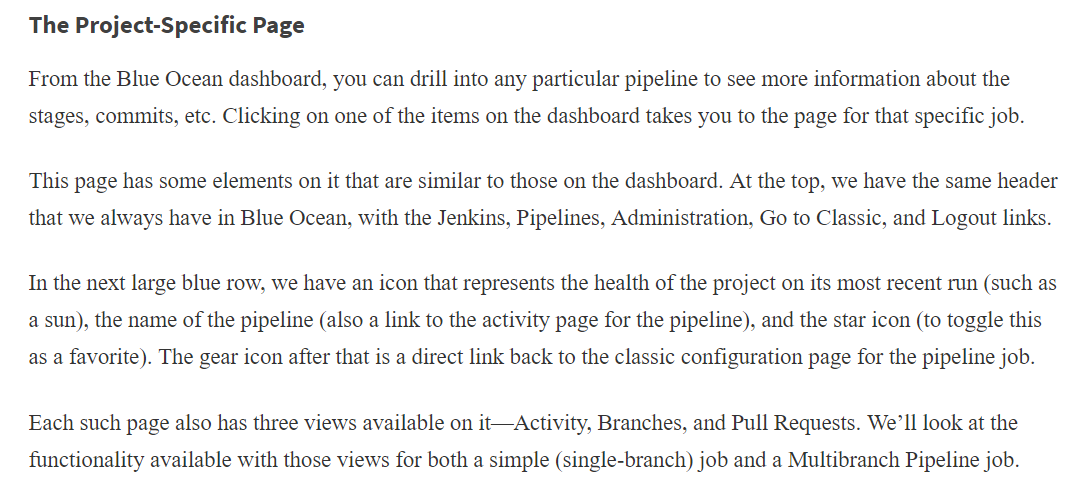


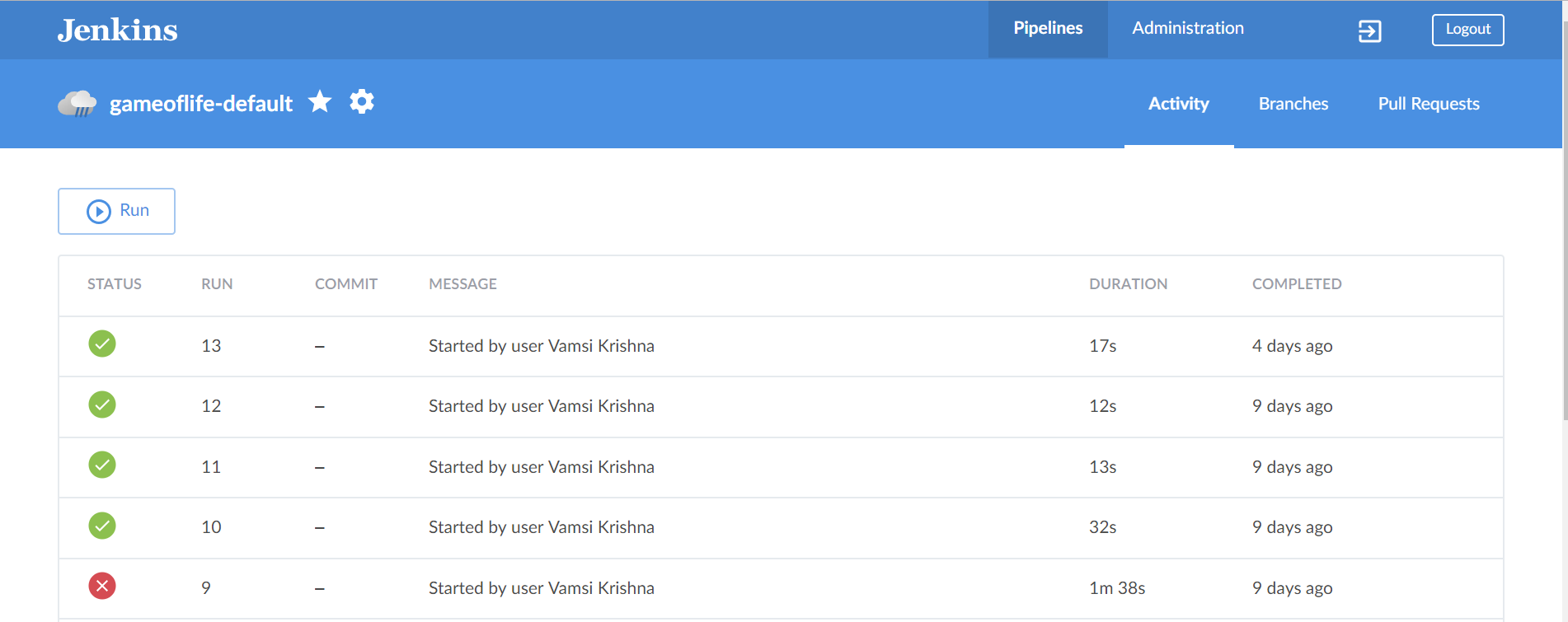
When a project is “favorited” by virtue of being starred, the shortcut at the top of the screen includes several options on the righthand side. We’ll discuss more about what each of those does in a later section of this chapter. The favorite shortcut can be removed by again clicking on the star. In this way, the star icon acts as a toggle for providing shortcut access to specific projects.

Finally, near the bottom of the dashboard screen will be some text that identifies the particular version of Blue Ocean that is running, and the version of Jenkins that it is being run on.

Clicking on any of the projects listed on the dashboard will take you to the specific page for that project. We’ll discuss the contents of that page next.

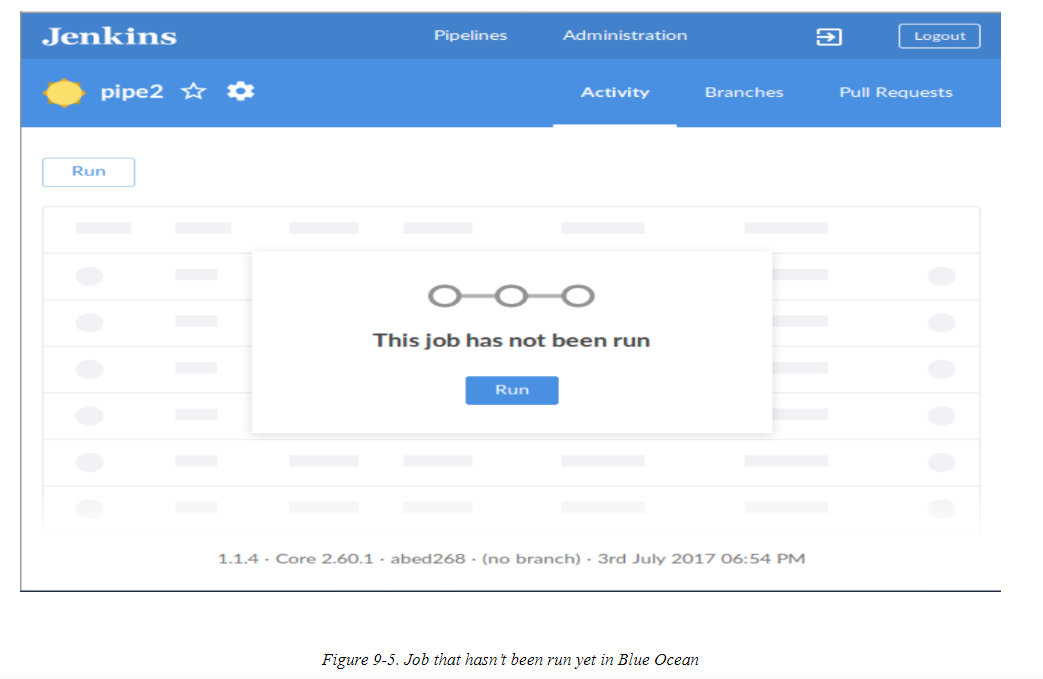
## The Project-Specific Page



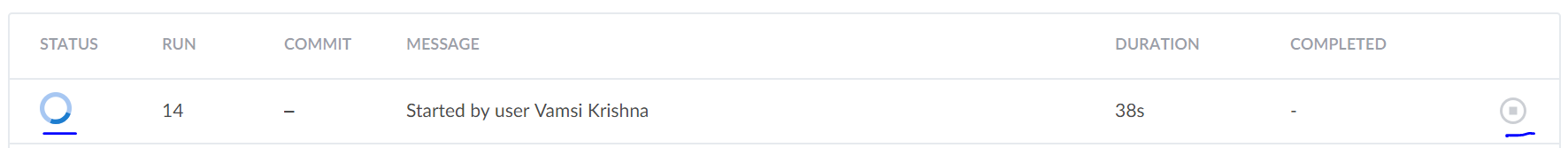


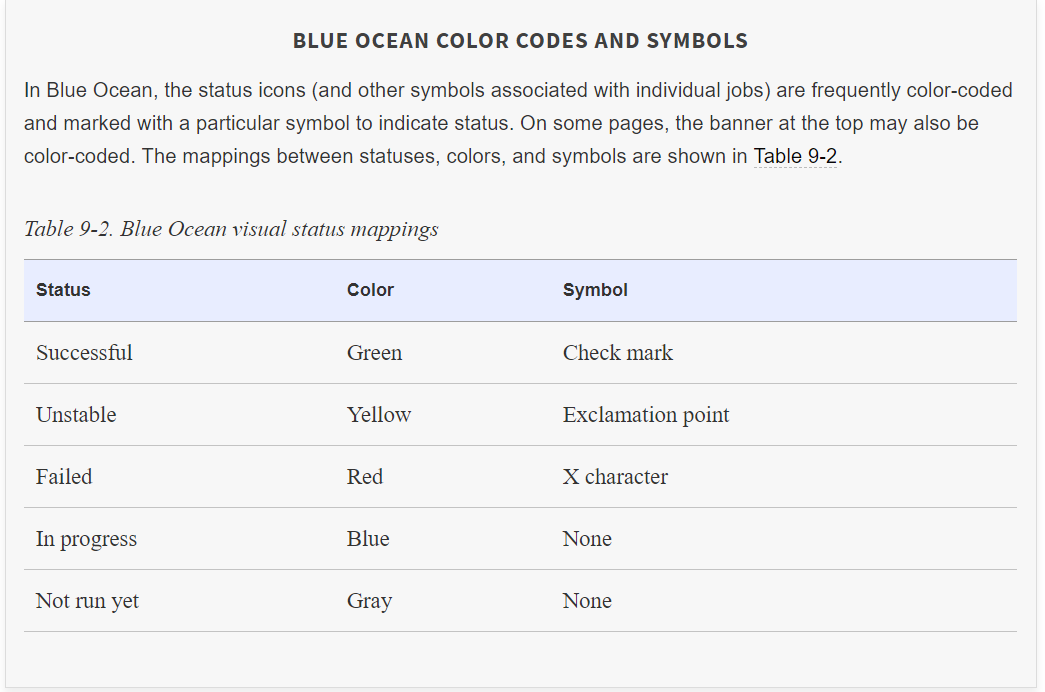
### SIMPLE PIPELINE ACTIVITY VIEW

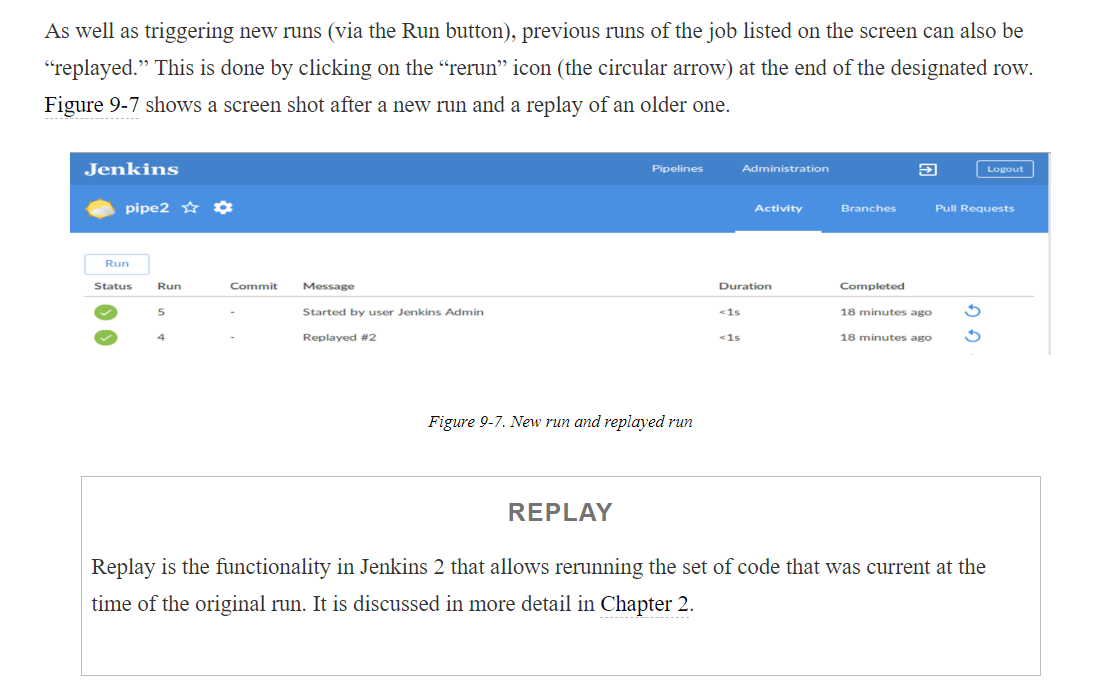
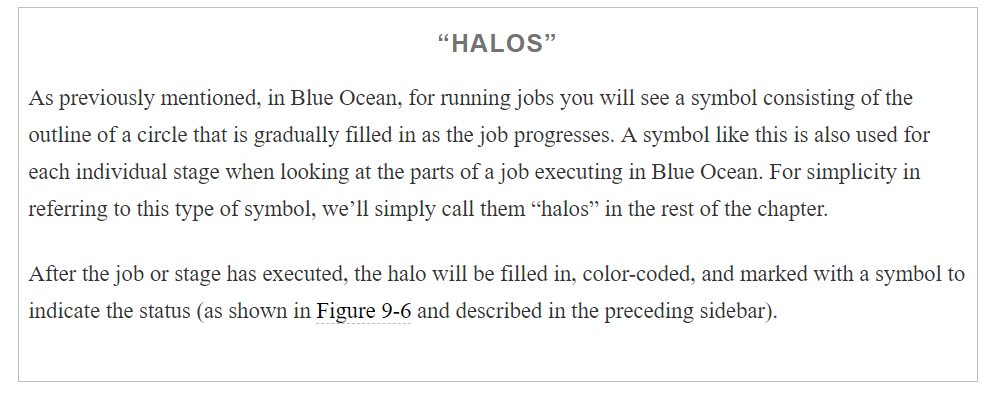
As the name implies, the Activity view is intended to show all activity (runs) of a selected pipeline. This is the default view for this page. It includes the runs for all branches of the selected pipeline that have been executed. If you select a job from the Blue Ocean dashboard that hasn’t been executed yet, you’ll see a screen like [Figure 9-5](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_new_blue_ocean_job).



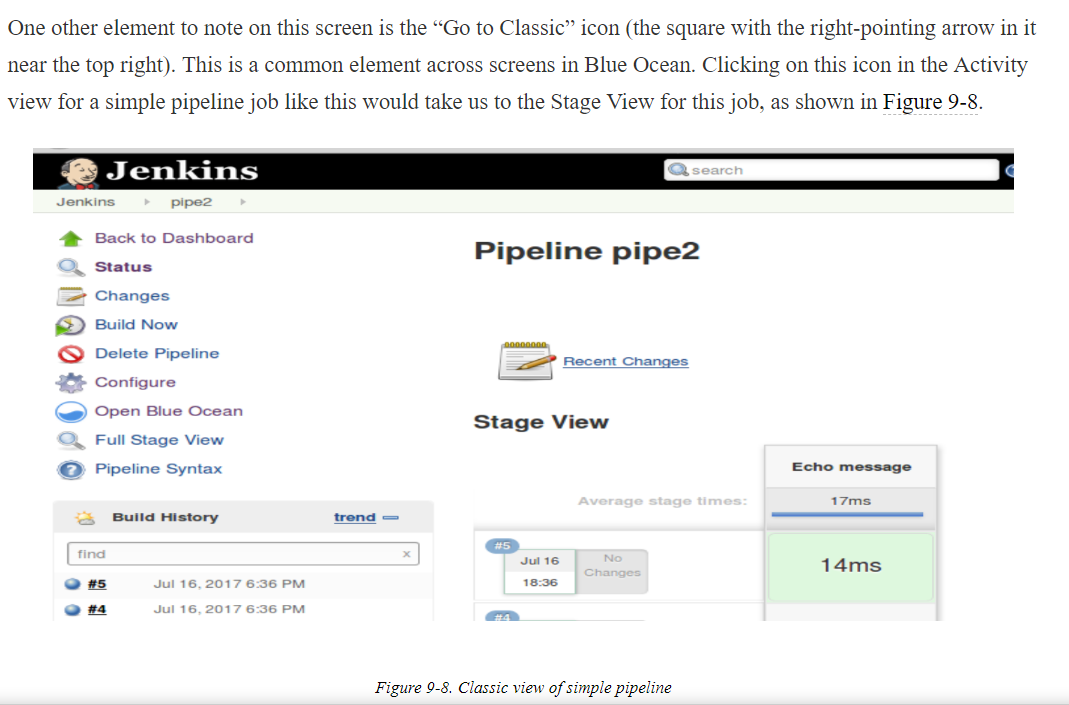
The Run button (either at the top left or in the dialog) then can be used to run the job. When the job is running,the circle icon on the left will gradually fill in as the job progresses. The icon in the last (rightmost) column can also be used to stop the job from running, if needed







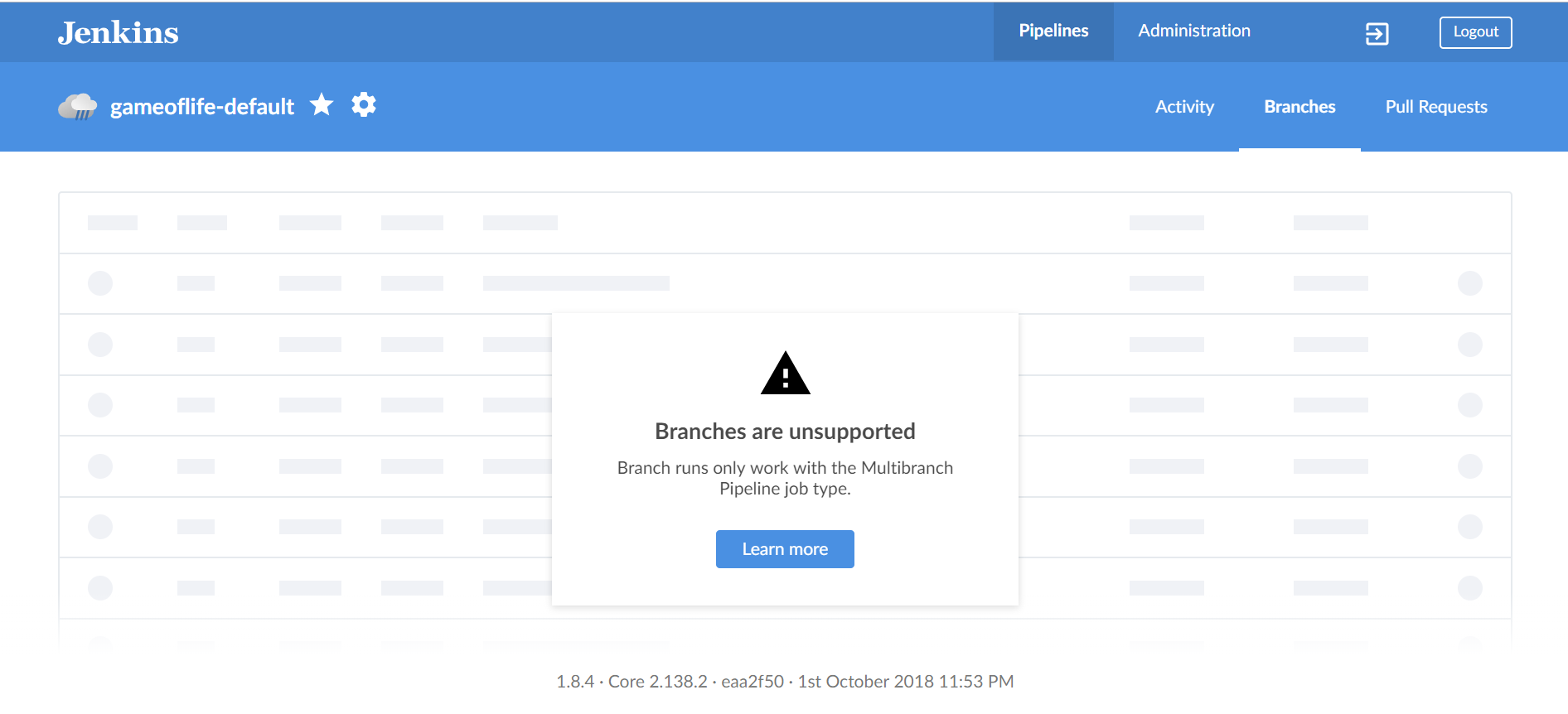
Above only pipeline project



### SIMPLE PIPELINE BRANCHES AND PULL REQUESTS VIEWS

In the same line as the Activity tab are the Branches and Pull Requests tabs. While these tabs are present for all pipelines, they are only applicable to Multibranch Pipelines.

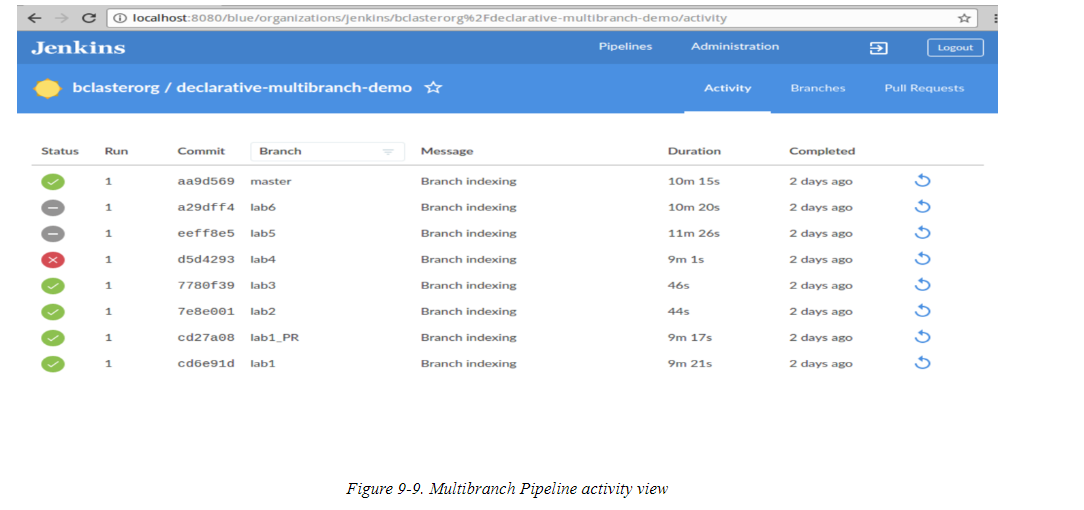
Clicking on either of these tabs for a non–Multibranch Pipeline will simply bring up an error dialog telling you that these do not apply, with a link to go to for more information.



### MULTIBRANCH PIPELINE ACTIVITY VIEW

Now that we’ve looked at the Blue Ocean interface for a simple pipeline, let’s look at a Multibranch Pipeline. A similar interface exists for these.

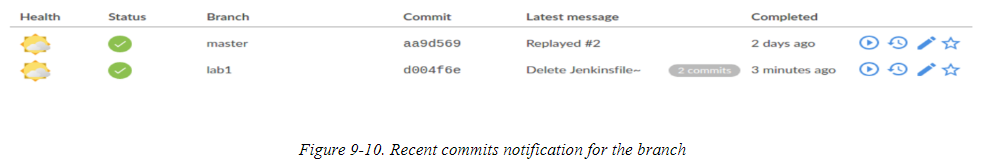
[Figure 9-9](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_MBP_ac_view) shows the Activity screen for a Multibranch Pipeline job. Note that we have several of the same links, icons, and headings as we saw on the Blue Ocean dashboard and on the simple pipeline pages.



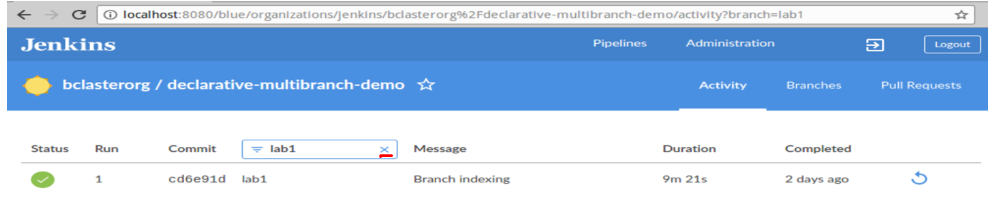
Here again, each row in the main part of the screen represents a run of a job for an individual branch. Clicking on any part of the row except for the icon in the far right column opens up a detail screen for that particular run. We’ll discuss the run detail screens later in this chapter.

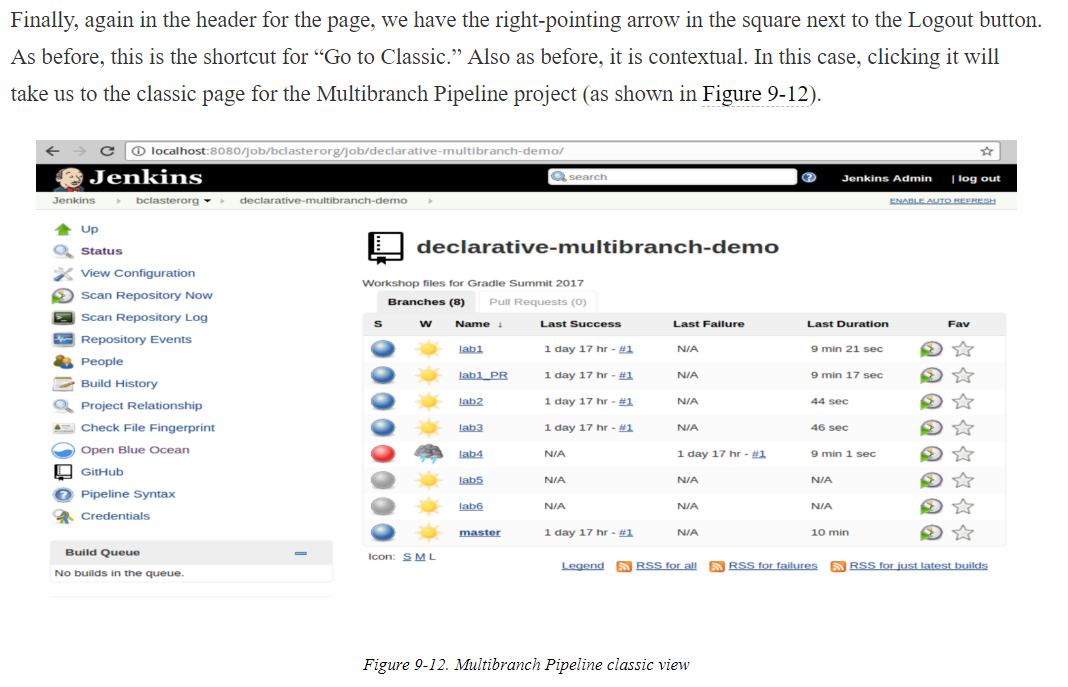
The names and values in each column are fairly self-explanatory, except for the last column. Clicking on that circular arrow executes an operation that reruns that particular run. (For previous runs, this amounts to a replay. The Replay feature is discussed in more detail in [Chapter 2](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch02.html#CH_The_Foundations).) As with the simple pipeline, once you start a run by clicking on that icon, the icon under the left Status column will change to a halo; the icon at the end of the row will change to one that can be used to stop the build (by clicking on it).

Additionally, in cases of multiple commits, you will see an “*n* commits” notification next to the “*Latest message*” field ([Figure 9-10](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_recent_commit_note_brnch)). Clicking on this will take you to the detail screen for that run.



For a Multibranch Pipeline project, the Branch column header on this screen also serves as a filtering mechanism. Clicking on the column header makes it become an editable field. You can select the desired branch from the drop-down or type in the desired branch name (see [Figure 9-11](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_filt_act_view)). To close the filtered view, click the “X” to the right of the branch name.

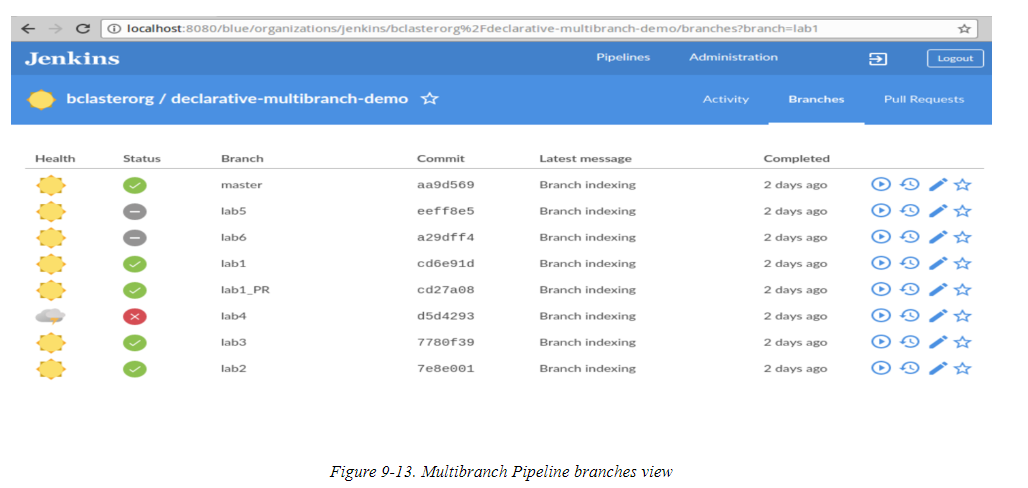




With a Multibranch Pipeline project, the Branches and Pull Requests tabs are valid and provide additional functionality. Let’s take a moment to talk about those.

### MULTIBRANCH PIPELINE BRANCHES VIEW

Whereas the Activity view for a Multibranch Pipeline project shows all the runs for all branches, the Branches tab is for working with the separate branches at the higher level as separate jobs. [Figure 9-13](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_MBP_branches_view) shows an example of this view, for the pipeline we’ve been looking at.



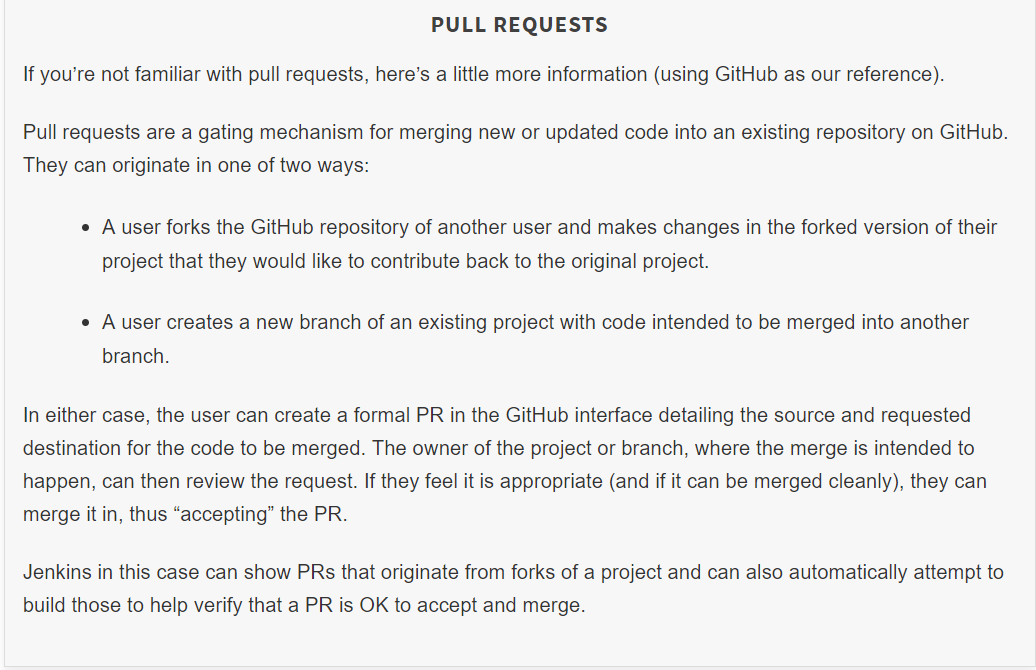
Each row represents one of the branches in the Multibranch Pipeline. The overall health indicator and last run status for each branch are on the left, followed by the branch name and the SHA1 commit that last updated it. In most cases, the value in the “Latest message” field will be “Branch indexing,” since that’s the process that runs when Jenkins scans for changes. Clicking on one of these rows (aside from the four icons at the end) takes you to a detailed run screen for the most recent run of that branch.

The four icons at the end of each row are clickable and invoke different functionality, summarized in [Table 9-3](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#TAB_branch_view_icons).

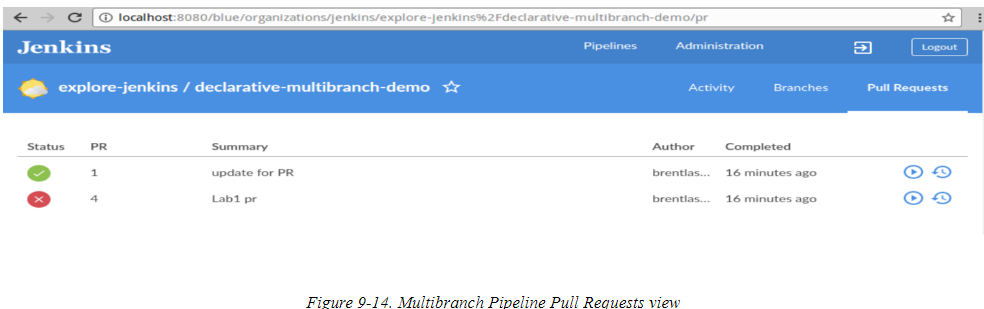
### 

### MULTIBRANCH PIPELINE PULL REQUESTS VIEW

For pipelines based on repositories that support pull requests, such as GitHub, this view is used to show any open pull requests. If you don’t have any open pull requests, switching to this view simply pops up a message telling you that you don’t have any.

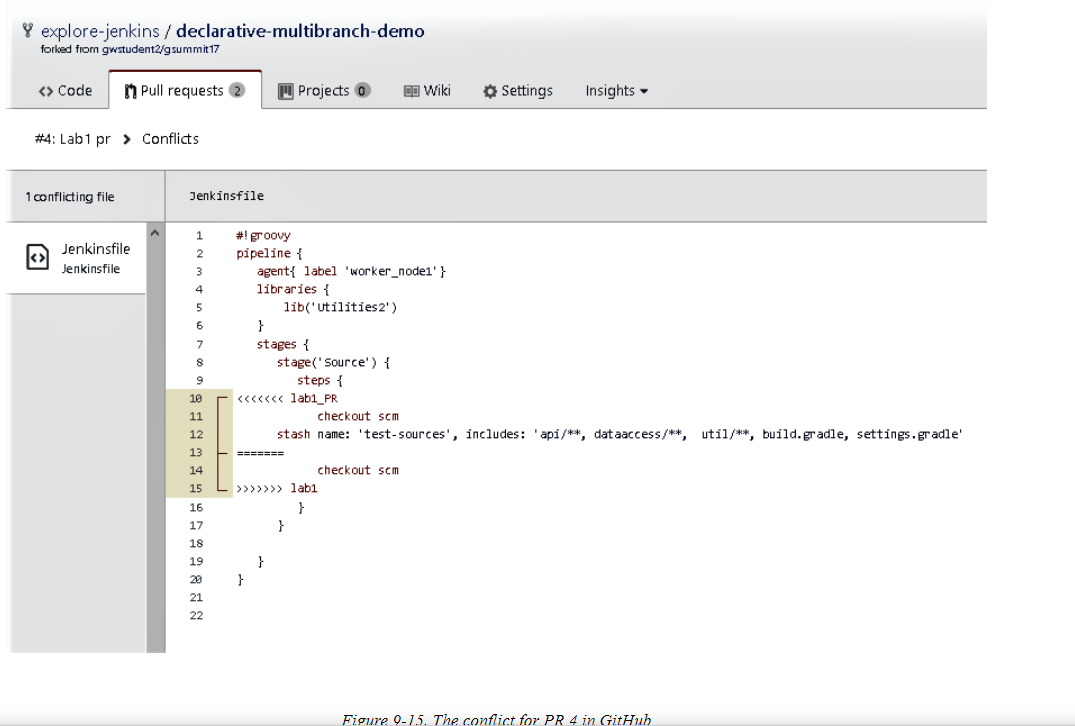


When you do have one or more open pull requests (originating from a fork) in the related GitHub project, they will show up in the Pull Requests view, as shown

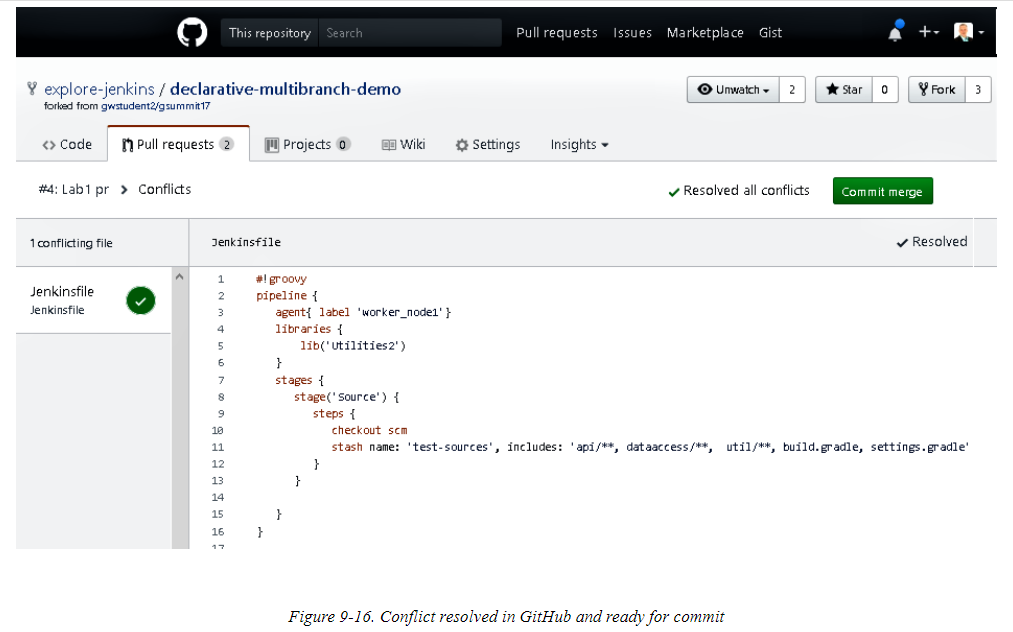


Note that Jenkins has tried to build these. Thus, we have the typical columns for things like the status and completed time, and the option to rerun them.

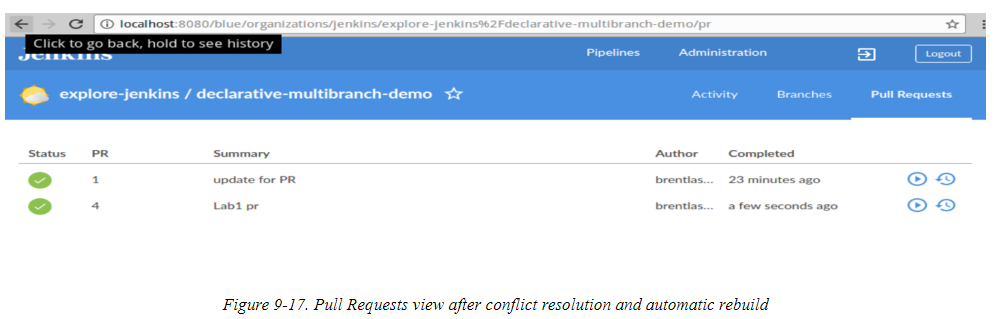
In this case, PR 4 had a conflict when Jenkins tried to build it. Via the connection already established from Jenkins to GitHub, the fact that Jenkins could not build it correctly will also be surfaced in the GitHub interface



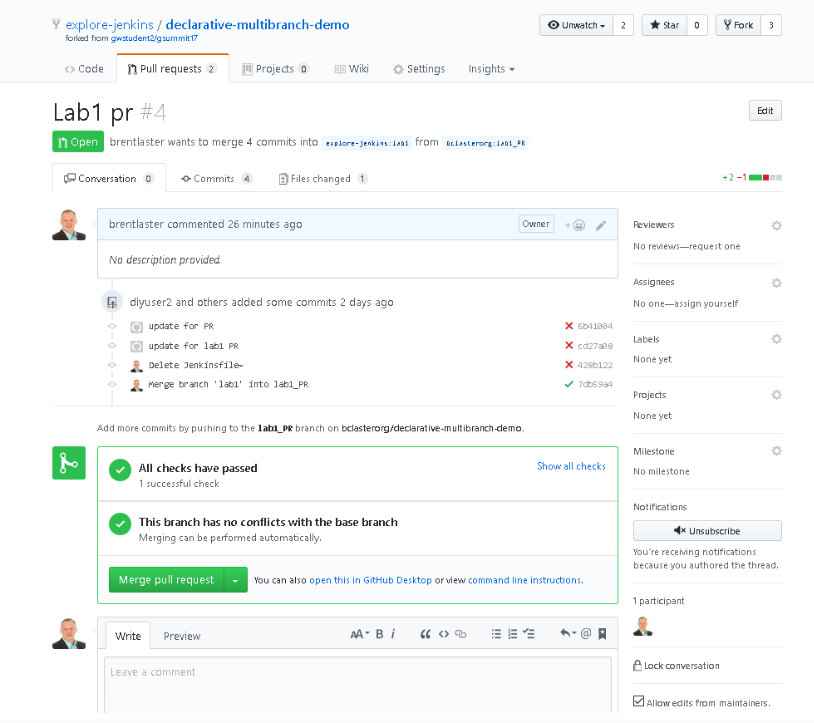
We can then choose to either not accept the PR (and close it), or resolve the conflict (by updating the code locally and pushing it back, or by updating it directly in GitHub). In this case, we just resolve the conflict in GitHub and commit the merge



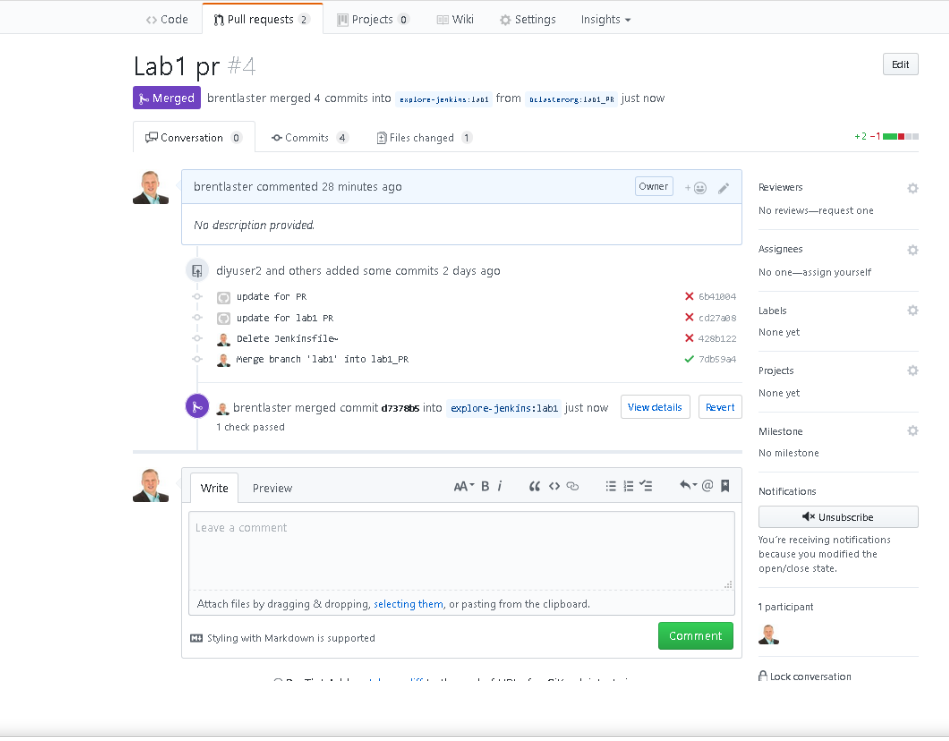
Once we have resolved the conflict and merged the change back on GitHub, Jenkins automatically detects that and rebuilds the PR. As shown in [Figure 9-17](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_pull_reqs_post_conflict_reb), with the conflict resolved, the PR builds successfully this time.



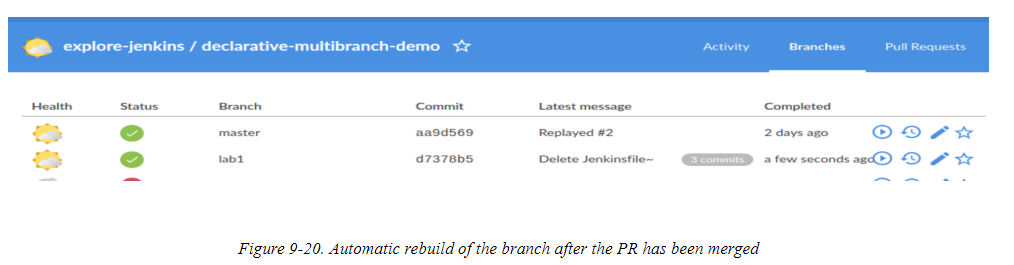
This is also reflected on the GitHub side. Note the “All checks have passed” section in



The next step is to go ahead and merge the cleanly building PR via the “Merge pull request” option on GitHub. After doing this, the GitHub interface will show that we have successfully merged this PR



After this occurs, at the next refresh interval Jenkins will do the branch indexing, detect the new change in the branch due to the merged PR, and rebuild the branch ([Figure 9-20](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_auto_rebuild_branch_PRmerge)). It will also add the tag showing there were multiple commits involved.



Since the PR has been merged, it will be removed from the Pull Requests view as well.

From many of the screens that we’ve already looked at in the interface, you can get to a more detailed view of a pipeline run (either one that has already occurred or one that has been initiated and is in progress). Next, we’ll look at the screen in Blue Ocean that shows the details of a particular run of a pipeline

## **The Run Page**

Below shows a pipeline job in progress in the Blue Ocean interface. Like the other pages we’ve looked at, this page has a number of common graphical elements on it and tabs that can be selected to see different elements of this particular run. But regardless of the selected tab, the large banner across the top, which we refer to as the “status banner,” remains on the screen. Let’s briefly discuss what information it conveys, and then see what’s contained on each tab.



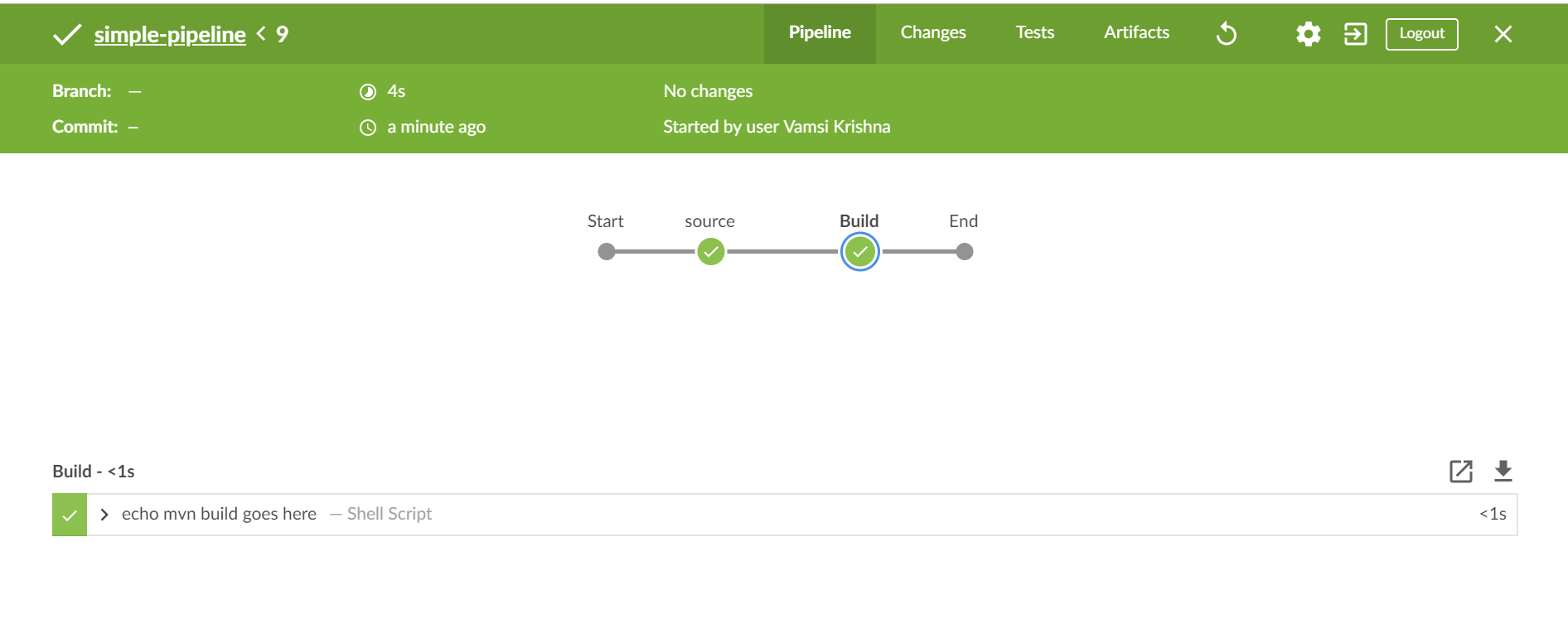
### THE STATUS BANNER

When the run is in progress, the status banner will have a blue background (indicating it’s running). In the upper-left corner is a halo indicating how much of the overall job has been completed. Next to that is the job name, the run number, a set of four tabs (which we’ll talk about shortly), and then some of the same icons we’ve already discussed that also appear on other screens, including a button to stop the build that’s in progress (or start it if it’s not running). In the next row, on the left are the branch name and the last commit SHA1. Still in that row, to the right, we have a “time” column. The top time value shows how much time this run is taking (or did take, if previously run). The bottom time value notes when this job was last run. Finally, toward the middle of that row, we may have additional information about the changes that were incorporated (and presumably are the motivation) for this run.

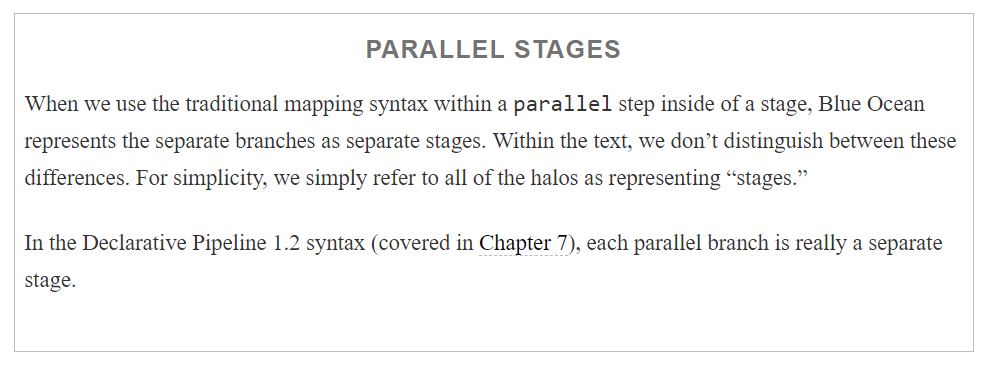
Within the main run page, the tabs for Pipeline, Changes, Tests, and Artifacts can change the views. We’ll look at each of those next.

### PIPELINE

Below shows an example run page with the Pipeline tab selected

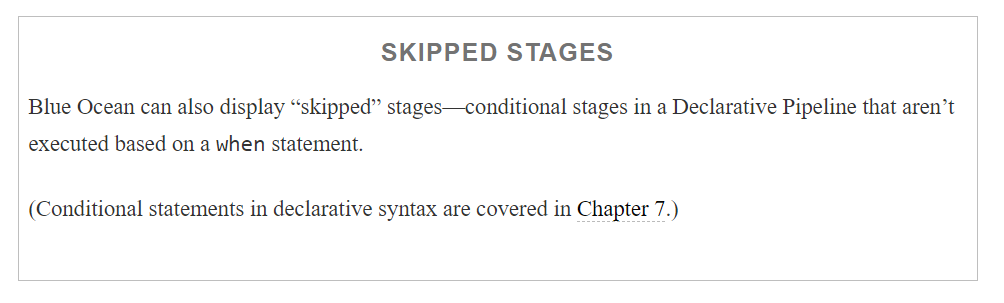


With the Pipeline tab selected, underneath the banner area is a graphical representation of this pipeline. The parts of the pipeline are represented at the granularity of stages, and each stage in the pipeline is represented by a halo. Where stages are coded/executed in parallel, the halos are lined up in the same column.



As each stage is executed, the halo for the stage is updated accordingly. When that stage is completed, the halo is color-coded and updated with a symbol to indicate success, failure, or an unstable state (as described in [“Blue Ocean Color Codes and Symbols”](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#blue_ocean_color_codes_and_symbols)). Partially filled in halos represent work in progress in those stages, and gray/empty halos represent stages in the pipeline that have not been processed yet.

Underneath the graphical representation of the pipeline is a section with logs for each step. Another feature of the graphical representation is that other parts of the screen can be filtered based on which stage is currently active. To make a stage active after a run, you can click on the halo for that particular stage. At any point in time, one(and only one) stage will be active (except in the case of parallel stages).

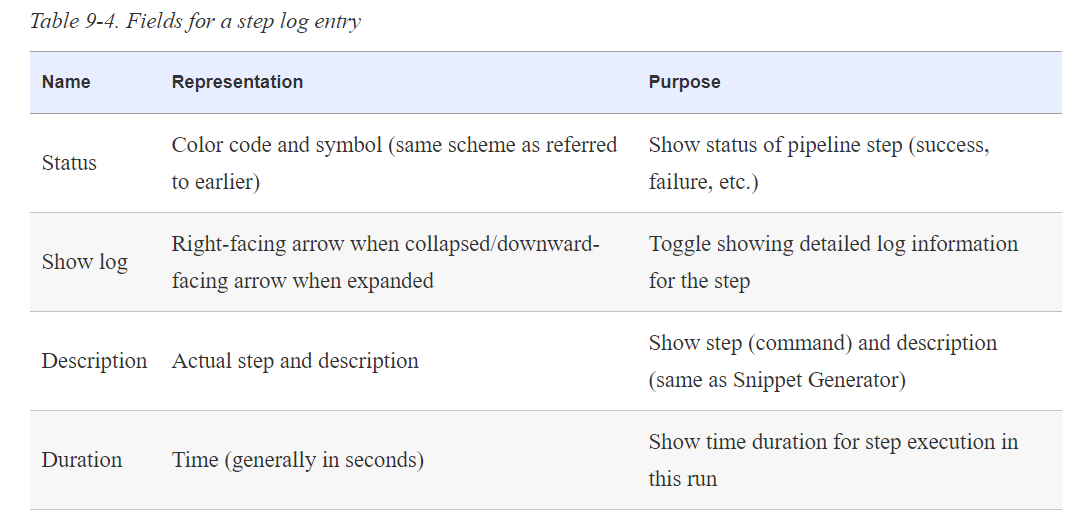


The log of steps below the graphical pipeline is filtered based on the currently selected stage. We’ll look at that next.

#### **Step logs**

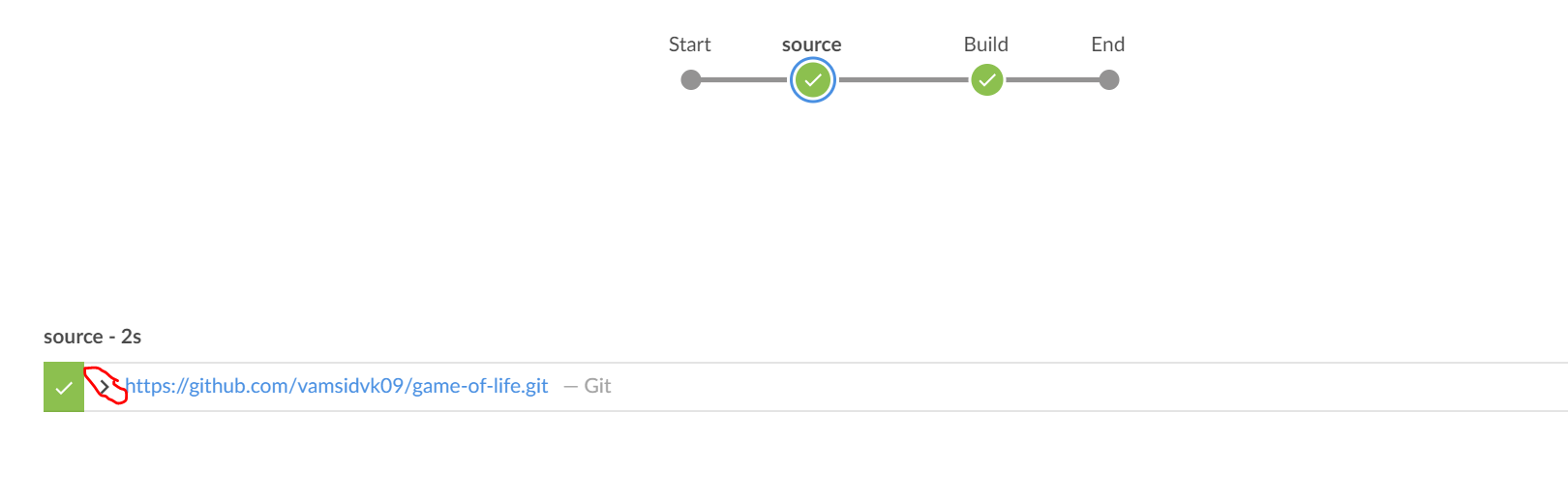
The section at the bottom of the Pipeline view allows you to look at logs for any stage of the pipeline, segmented by steps. The set of steps shown are the ones for the currently selected stage in the graphical representation above.

A separate row is shown for each step in the stage. [Table 9-4](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#TAB_fields_step_log_entry) lists the possible fields for a step log entry.





The main benefit here is derived from being able to select a stage, and then a step in that stage, and then dive in to get the logs just for that step. Clicking on the “>” sign in the second column of the row causes the log to be shown. The sign will then change to a “V”. Clicking on the field again causes the log to be collapsed.





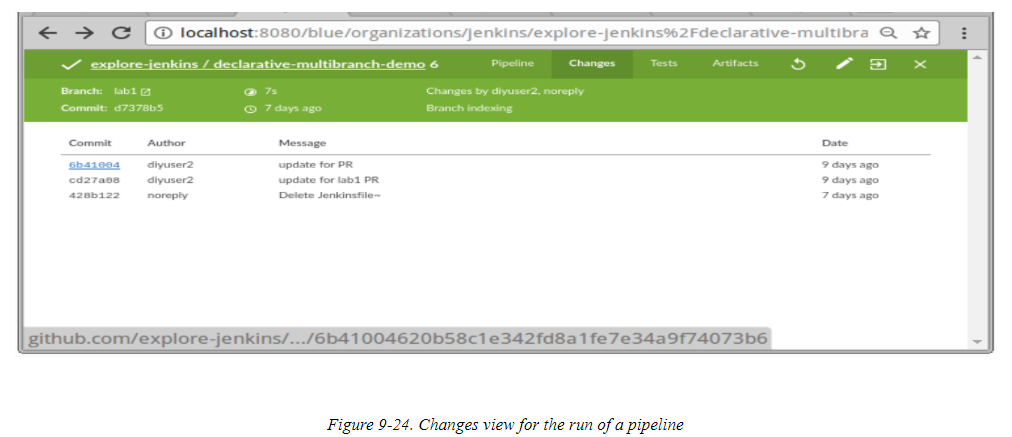
One other note about this view: on the far right side, immediately above the set of rows for the steps, are two icons that allow you to look more closely at logs. The one in the form of a square with a diagonally pointing arrow allows you to display the log for the selected step in a new full-size screen. The icon next to it, with the downward pointing arrow, allows you to download a copy of a log.



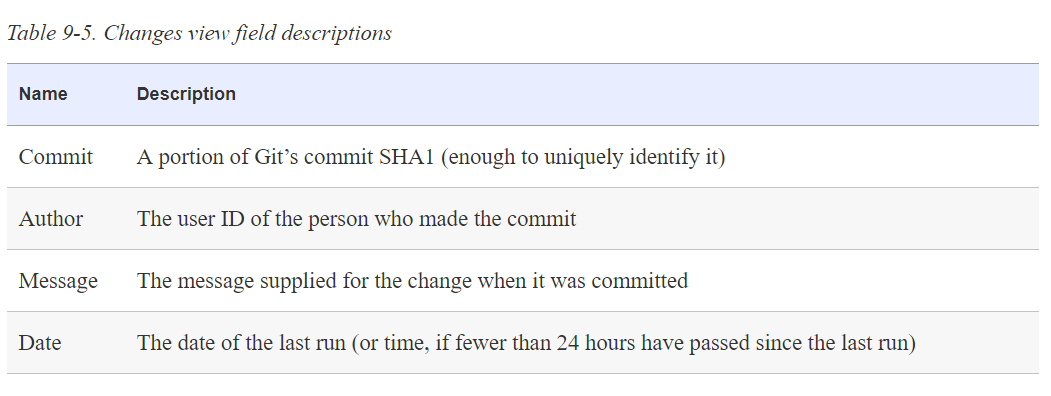
### CHANGES

The next view to look at is shown by selecting the Changes tab. As the name implies, this view shows the set of changes that were made in source management for this run.

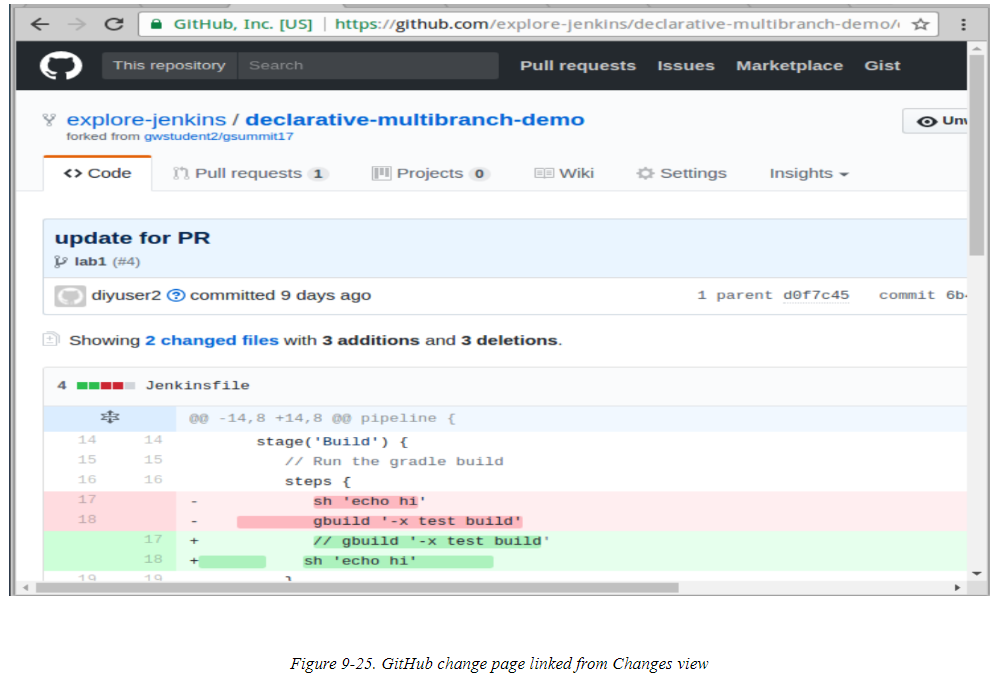
Below shows the Changes view for one run of a particular pipeline.



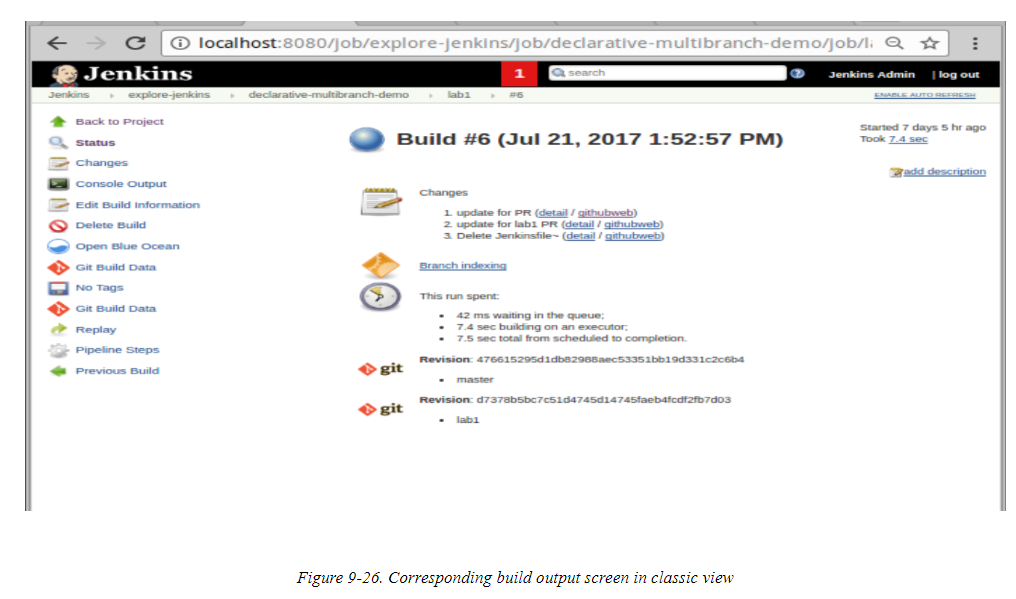
The fields here are fairly self-explanatory if you are familiar with Git; however, we’ll briefly describe them here for completeness (see [Table 9-5](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#changes_view_field_descriptions)).



One useful feature of this Blue Ocean screen is being able to select and click on any of the commits. This action will then jump to the change in the source management system. For example, clicking on the highlighted commit in the previous figure takes us to the GitHub page with the details for it



It’s worth noting that the information presented on the Changes screen is a subset of the information found on the build output screen in the classic view



### TESTS

Plugins such as JUnit allow us to archive test results and have Jenkins report on them. Assuming there are steps that actually run the tests in the pipeline, code like the following can be used to archive the test results:

junit '\*\*/build/reports/\*\*/\*.xml'

Such code would most commonly be included in post-processing designed to always be run for a pipeline. For a Declarative Pipeline, this could take the form of:

post {

       always {

           junit '\*\*/build/reports/\*\*/\*.xml'

       }

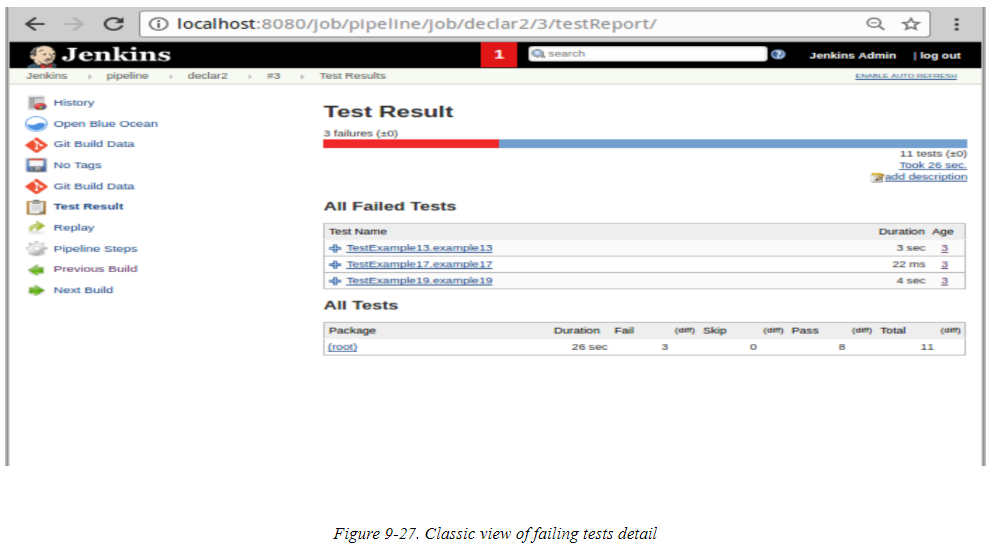
In a Scripted Pipeline, you could include it in the finally block of a try-catch-finally structure:

**finally** {

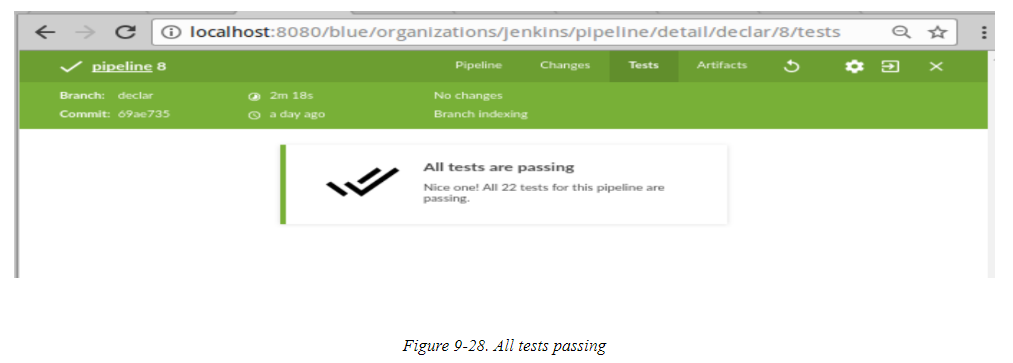
        junit 'build/reports/\*\*/\*.xml'

    }

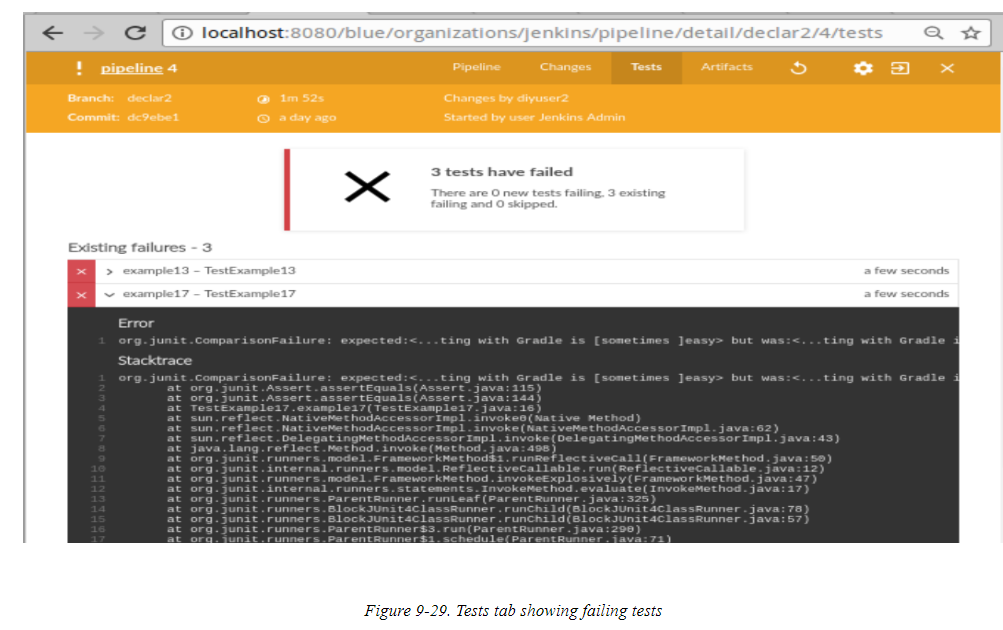
Based on this kind of step, in the classic view, Jenkins can create trend reports for success/failure of tests as wellas produce detail screens on failing tests as shown in [Figure 9-27](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_classic_view_failing_tests).



The Tests view in the run screen in Blue Ocean provides a similar detail screen for failed tests. If the step to archive the tests was not included, the screen will display a message stating “There are no tests archived for this run.” And if all the tests have passed, the screen will display a message like



Pipelines are usually configured/coded in Jenkins such that failing tests set the build result to “unstable.” This is represented in the Blue Ocean interface with a yellow color and an exclamation point, so the Tests view showing an unstable state with the failing test’s detail would look something like



Notice that we can expand the row for each of the failing tests to get the related log, just as we could for the step logs on the Pipeline tab.

### **ARTIFACTS**

If your pipeline is configured to produce and archive artifacts, the Artifacts page will let you view and optionally open or download those artifacts. This tab corresponds to the Build Artifacts portion of the classic view’s screen shown in [Figure 9-30](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_BO_runscreen_artifacts_classic).



The archive pipeline step allows for archiving artifacts in your code. In a simple form, it looks like this:

archive 'build/libs/\*\*/\*.jar'

Like the code to archive test results, such code would most commonly be included in post-processing designed to always be run for a pipeline. For a Declarative Pipeline, this could take the form of:

post {

       always {

         archive 'build/libs/\*\*/\*.jar'

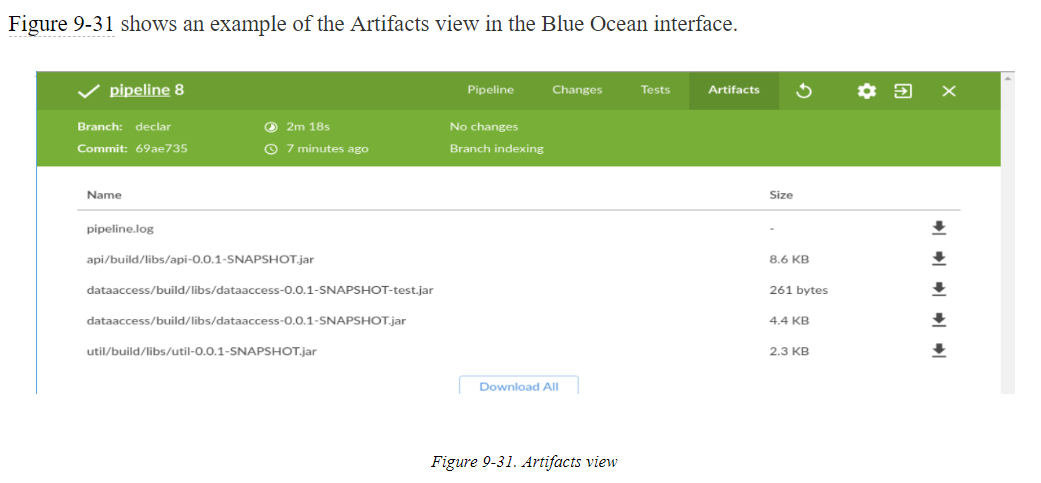
       }

In a Scripted Pipeline, you could include it in the finally block of a try-catch-finally structure:

**finally** {

        archive 'build/libs/\*\*/\*.jar'

    }



Note that the first item listed is *pipeline.log*. This is the log from this run of the pipeline and is always available here, even if no other artifacts are archived.

Note also that the screen has icons on the far right to download the individual artifacts, and you can click on an artifact’s name to “open” the artifact. (With the exception of the pipeline log, opening may translate to just downloading for most artifacts.)

Finally, there is a Download All button at the bottom. As the name implies, this button can be used to download all of the listed artifacts at once as a ZIP file. This is shown in [Figure 9-32](https://www.safaribooksonline.com/library/view/jenkins-2-up/9781491979587/ch09.html#fig_download_all_artifacts_zip).

