



Cloud Developer Certification Preparation

Exercise 5.4:

**Managing source code for projects in IBM Bluemix
DevOps Services**

Exercise 5.4: Prerequisites

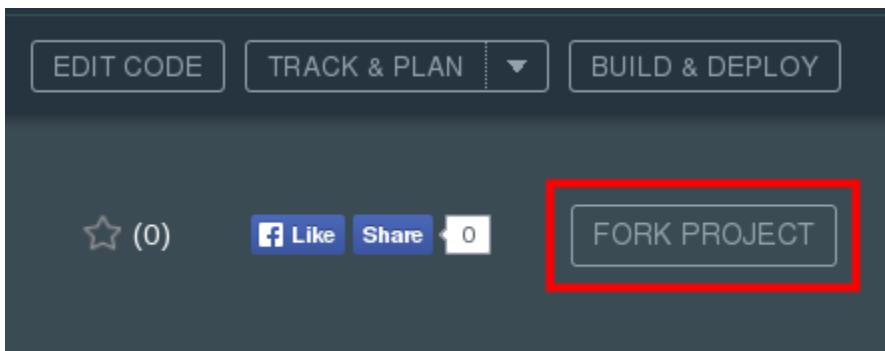
Sign up for a 30-day free trial [IBM Bluemix account](#) if you don't already have one. Also, create a DevOps account: <https://hub.jazz.net/>. You should use the same credentials for both the Bluemix and DevOps accounts.

You also need the following software:

- A web browser supported by Bluemix:
 - Chrome: the latest version for your operating system
 - Firefox: the latest version for your operating system and ESR 31 or ESR 38
 - Internet Explorer: version 10 or 11

Exercise 5.4.1: Forking a project

1. Log in to IBM DevOps Services and fork the project located at <https://hub.jazz.net/project/ecosysdevcnc/cdc-lab-5/overview> into a new project.



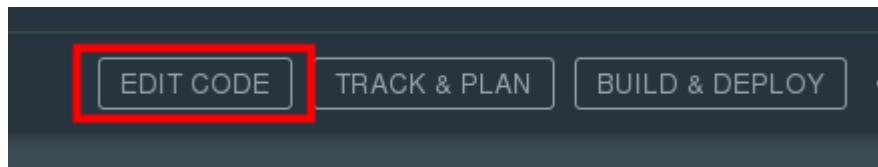
2. Give the new project a unique name, select all the check boxes, and choose an appropriate Bluemix runtime configuration. Click **CREATE**.

A screenshot of a 'Fork Project' dialog box. It starts with a title 'Fork Project'.

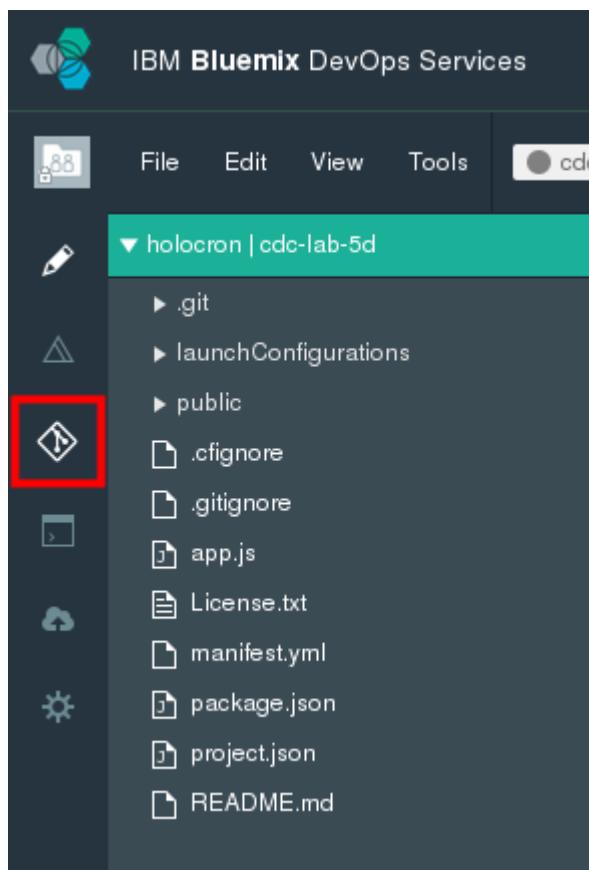
- 'Name your project:' input field contains 'holocron cdc-lab-5e', which is highlighted with a red box.
- 'URL:' field shows the URL <https://hub.jazz.net/project/holocron/cdc-lab-5e>.
- A checked checkbox 'Private project (Invited team members only)'.
- An unchecked checkbox 'Restrict membership (IBM only)'. A note below it says: 'You can restrict this project's membership because your email address ends with ibm.com. If this project is for **IBM confidential** business, you must select this option and agree to certain conditions.' with a link 'Learn more'.
- An unchecked checkbox 'I accept the terms and conditions'.
- A checked checkbox 'Add features for Scrum development' with an information icon.
- A checked checkbox 'Make this a Bluemix Project' with an information icon. This checkbox is also highlighted with a red arrow pointing to it.
- 'Select a Bluemix space to bill your services to:' section with dropdown menus:
 - 'Region': 'IBM Bluemix US South'
 - 'Organization': 'vmorris@us.ibm.com'
 - 'Space': 'dev'
- A note at the bottom: 'These selections can be changed later in the options for your Project Settings.'
- At the bottom right, there are 'CANCEL' and 'CREATE' buttons, with 'CREATE' highlighted with a red box.

3. After you see the message about successfully creating your project, click **EDIT CODE**.

Exercise 5.4: Managing source code for projects



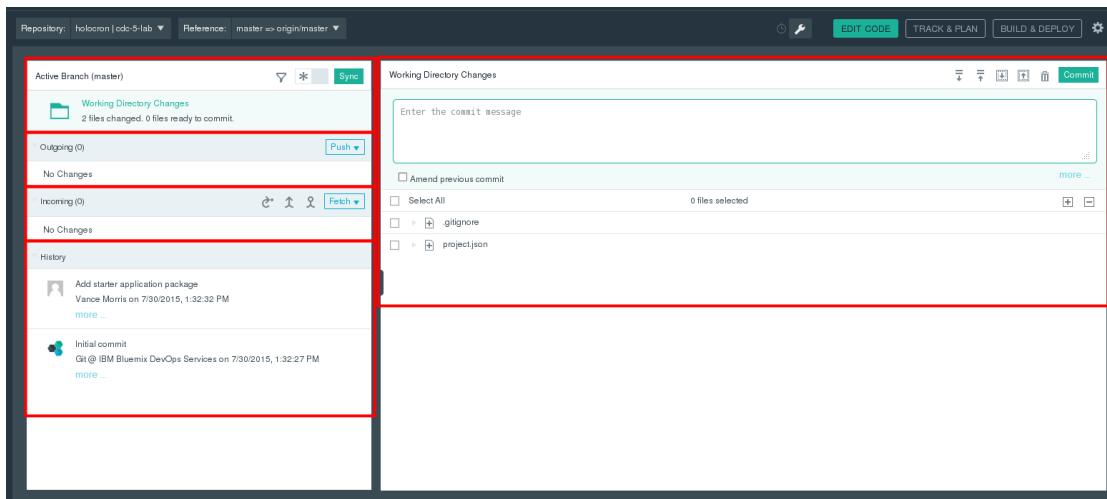
4. Wait a moment for the workspace to be set up. When the project is ready, click the **Git** icon located on the left column.



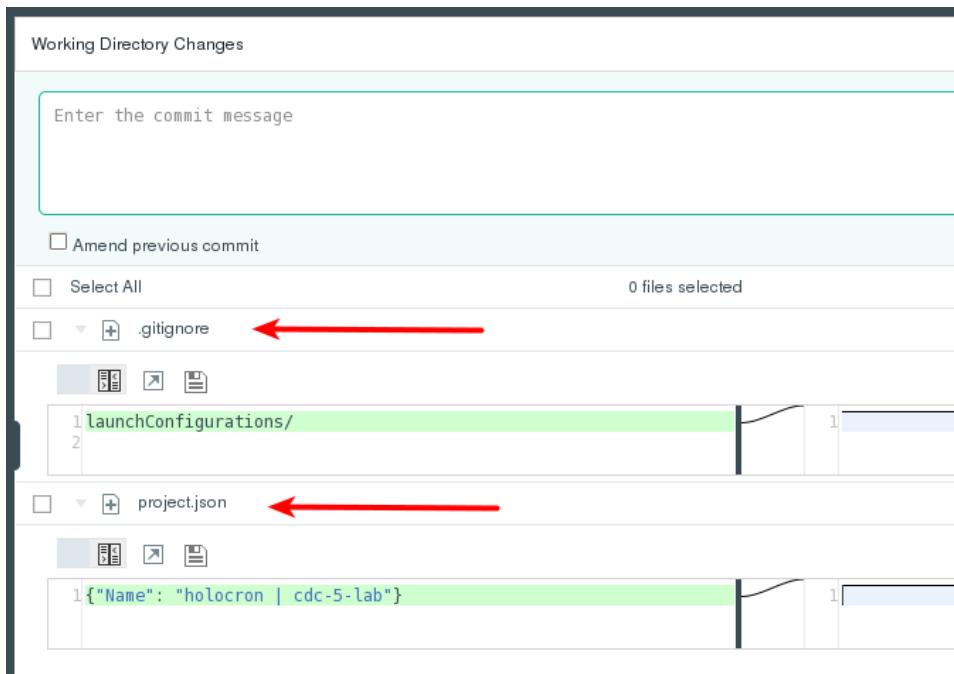
Exercise 5.4: Managing source code for projects

Exercise 5.4.2: Adding local changes to the Git repository

The Git Repository view, or "Git view" is organized into two panes. The left pane shows any staged outgoing changes, any incoming changes, and the active branch's history. The right pane shows any working directory changes and provides the functionality to select changed files, enter a commit message, and stage the files for a push.



The process of forking a project causes changes to occur in the files. Notice that there are two files in the right pane. Click each one to get a view of what has changed.

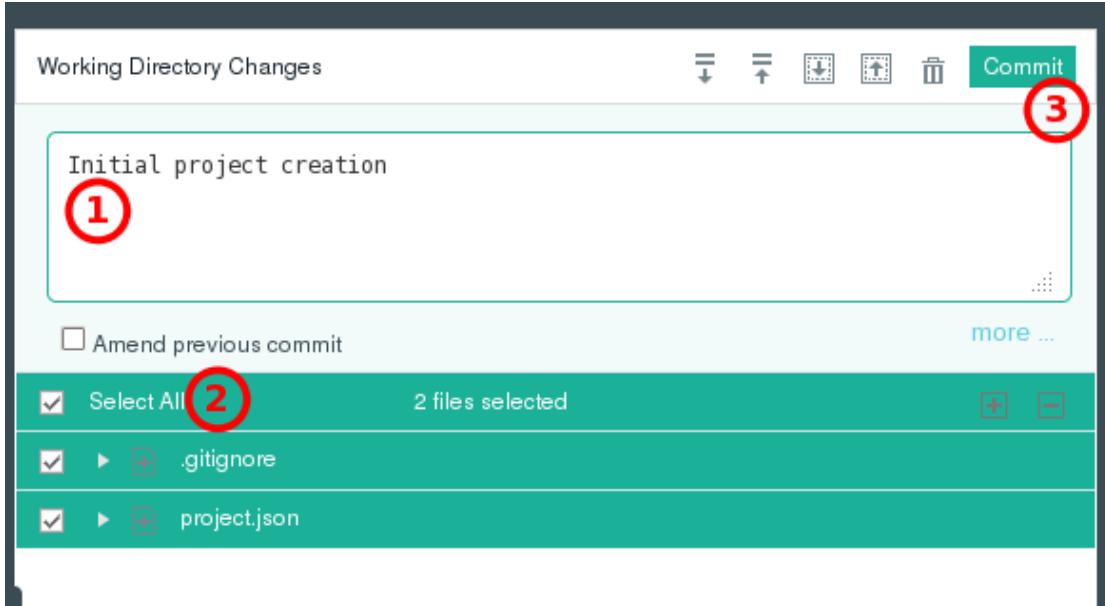


Exercise 5.4: Managing source code for projects

The `.gitignore` is a file used by Git to list all the files and directories in the project that should be ignored by Git. Items such as Bluemix launch configurations and build artifacts generally do not need to be tracked by Git and should not be committed to the master branch.

The `project.json` is a file used by DevOps Services to identify your project. It should remain unchanged and can be committed to the master branch.

1. Enter a commit message, such as `Initial project creation`, select **Select All**, and then click **Commit** in the top right.



Exercise 5.4: Managing source code for projects

Notice that the working directory changes pane is now empty of any files, and a new entry is made in the Outgoing section of the left pane. This is called "staging" a commit.

The screenshot shows the IBM Bluemix DevOps Services interface. At the top, there are repository and reference dropdowns. On the left, a sidebar contains icons for file operations like edit, delete, copy, move, and sync. The main area has tabs for Active Branch (master), Working Directory Changes (which says 'Nothing to commit.'), and Outgoing (1). The Outgoing tab is highlighted with a red box. It lists an 'Initial project creation' commit by Vance Morris on 7/31/2015 at 10:24:17 AM. Below that is an Incoming (0) section with 'No Changes'. At the bottom is a History section with a commit to add a starter application package by Vance Morris on 7/30/2015 at 1:32:32 PM. A 'Push' button is visible in the Outgoing section.

The master branch is still unchanged and will remain so until you push the outgoing commit. If there were any incoming changes from other branches, you might need to merge them together with your outgoing changes, and you will have the opportunity to do so before the push completed.

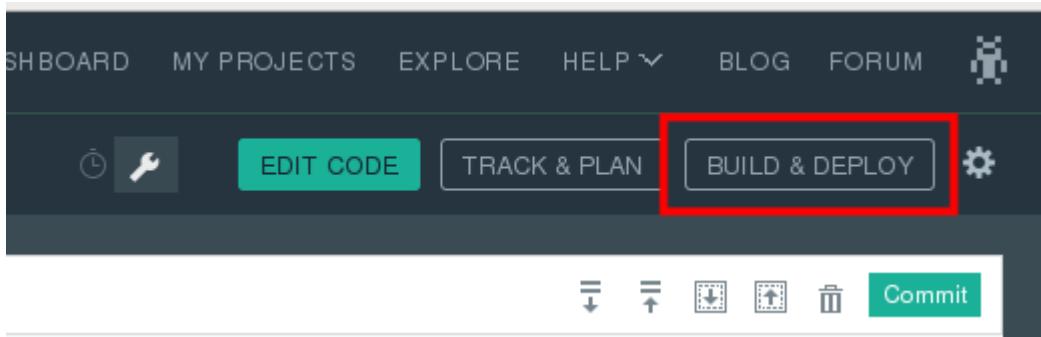
2. Click **Push** and note that the History is updated to include your commit. Any other branches will be able to pull this commit and its changes into their local working copy.

This screenshot shows the same interface after the commit has been pushed. The History section now includes three entries: 'Initial project creation' by Vance Morris, 'Add starter application package' by Vance Morris, and 'Initial commit' by 'Git@ IBM Bluemix DevOps Services' on 7/30/2015 at 1:32:27 PM. The 'Push' button is no longer visible.

Exercise 5.4.3: Verifying the integrity of code delivered to the repository with a build

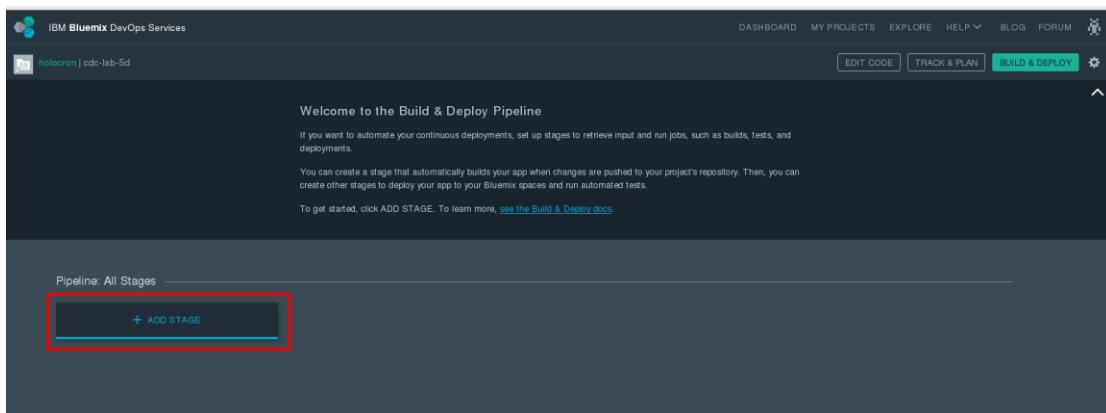
Next we will configure DevOps Services to perform a build of the code and have the build process trigger a successful push to the master branch. The builder will watch the master branch for any new pushes, and when detected, it will automatically build the project into a deployable package.

1. Click **BUILD & DEPLOY** in the top right.



The build and deploy pipeline view gives a high level view of the various stages that are configured in the pipeline.

2. Click **ADD STAGE**.



Exercise 5.4: Managing source code for projects

A stage consists of a series of jobs that run in sequence and the input for those jobs. Stages can be configured to run automatically based on triggers.

The input can be either the project SCM repository or build artifacts from a preceding stage. Input applies to all jobs in the stage. When a stage is run, the input is fetched. The files are placed in the working directory before each job starts.

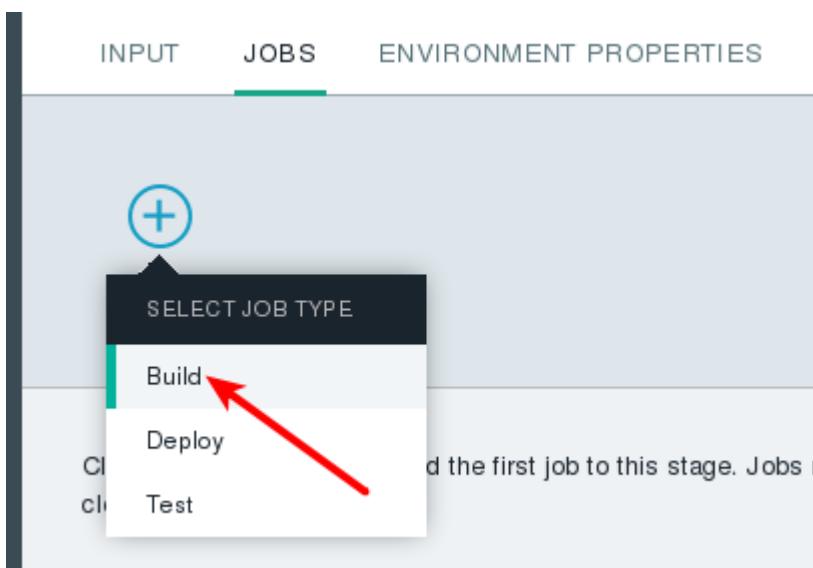
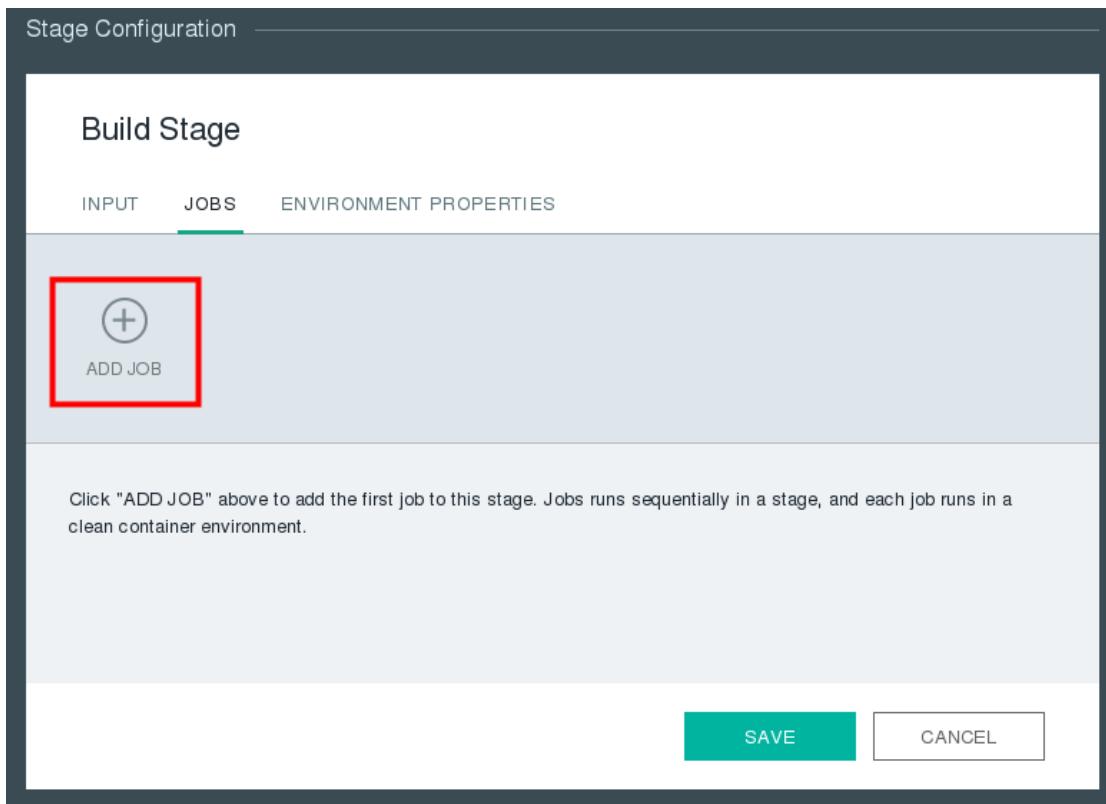
Jobs perform the work for a stage, such as building, deploying, and testing. The jobs in a stage run sequentially, and each job runs in a clean container environment. Files do not persist across job executions, so any dependencies required by the jobs must be provided in the stage input or installed as part of the job.

3. Give the stage a name to identify it. Then, click the **JOB**S tab. The input for the stage is the SCM repository that contains the master Git branch of your project. Also, note that the default has the stage run whenever any change is pushed to Git.

The screenshot shows the 'Stage Configuration' dialog box. At the top, the stage name 'Build Stage' is displayed. Below it, there are three tabs: 'INPUT' (underlined), 'JOBS' (with a red arrow pointing to it), and 'ENVIRONMENT PROPERTIES'. The 'INPUT' tab is active, showing 'Input Settings' with 'Input Type' set to 'SCM Repository' and 'GIT URL' set to 'https://hub.jazz.net/git/holocron/cdc-lab-5d'. Under 'Branch', 'master' is selected. In the 'Stage Trigger' section, the radio button 'Run jobs whenever a change is pushed to Git' is selected. At the bottom right are 'SAVE' and 'CANCEL' buttons.

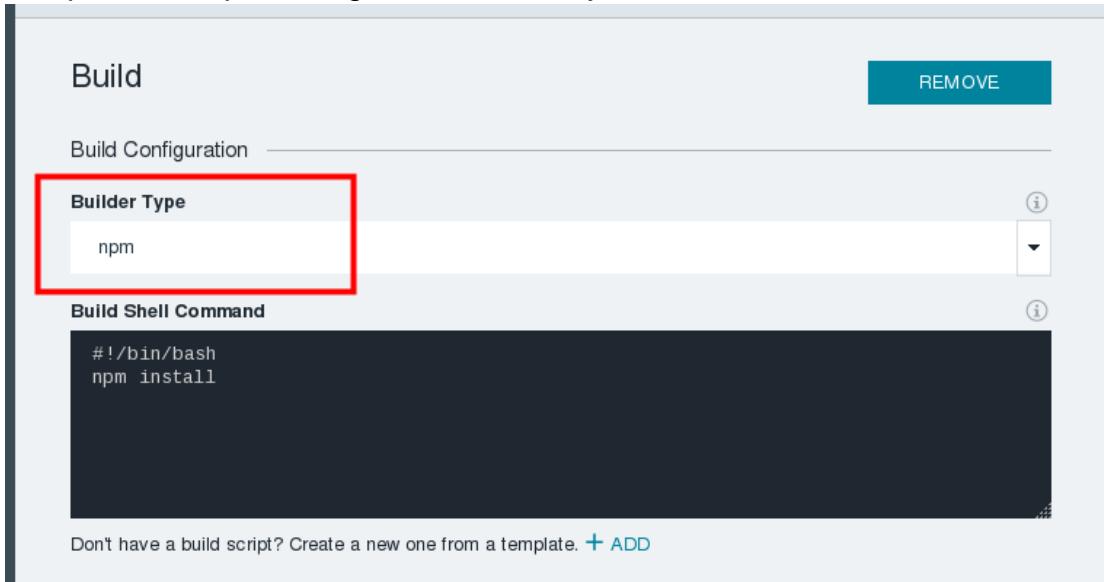
Exercise 5.4: Managing source code for projects

4. In the **J OBS** tab, click the + sign to add a job. Then, select **Build**.

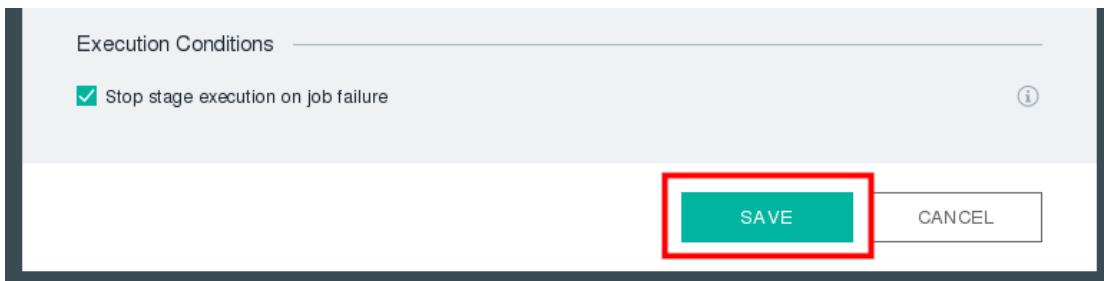


Exercise 5.4: Managing source code for projects

5. Under **Builder Type**, select **npm**. Many builder types are available to facilitate your project's needs, and other configuration options include a working directory, build archive directory, and an option to stop the stage execution if a job fails.



6. Click **SAVE**.



Exercise 5.4: Managing source code for projects

You are automatically returned to the build and deploy pipeline overview and the new stage is displayed.

7. Start a build automatically by clicking the **Play** button within the stage.

The screenshot shows the 'Pipeline: All Stages' interface. A single stage, 'Build Stage', is listed. The stage status is 'STAGE NOT RUN'. Below this, 'LAST INPUT' is shown as 'Not yet run'. To the right is a 'Git URL' link. Under 'JOBS', there is one job named 'Build' which is also 'Not yet run'. A link to 'View logs and history' is provided. At the bottom, 'LAST EXECUTION RESULT' shows 'No results'. A red box highlights the play button (a circular arrow icon) in the top right corner of the stage card.

Exercise 5.4: Managing source code for projects

The stage starts running, and the build job status changes from Pending, to Queued, to Running, to Succeeded.

8. To view the logs from the build, click the job status message.

The screenshot shows a pipeline interface titled "Pipeline: All Stages". Under the "Build Stage" section, a green bar indicates "STAGE PASSED". Below it, under "LAST INPUT", there is a user profile picture of Vance Morris and a message: "Last commit by Vance Morris 16 min ago" with a link "Initial project creation". Under the "JOBS" section, a box contains a checkmark icon and the text "Build Succeeded just now". This entire "JOBS" box is highlighted with a red border. Below this, under "LAST EXECUTION RESULT", there is a build icon and the text "Build 1".

9. From the Stage History view, you can see the history of all previous executions of this stage, and download any artifacts generated as a result of a particular stage execution.

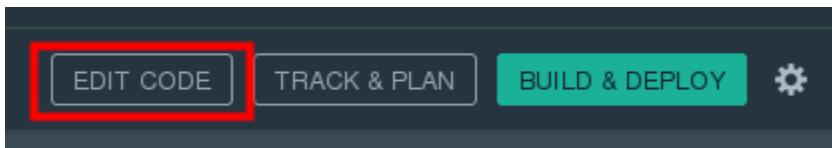
Exercise 5.4: Managing source code for projects

The screenshot shows the Jenkins Stage History interface. At the top, there's a dark header bar with the text "Stage History". Below it, a "Build Stage" card is displayed. The card has a green header bar with the number "1" and the status "Succeeded just now". Underneath, there are two items: "Input Initial project creation" and "Build Succeeded". To the right of the card is a detailed view of the first build. It shows a green checkmark icon followed by "Build 1 Succeeded just now". Below this, it says "STARTED Monday, August 3, 2015 10:52 AM DURATION 11 seconds DEPLOYED TO No spaces". There are three tabs at the top of this section: "LOGS" (which is selected), "CHANGES", and "ARTIFACTS". The "LOGS" tab displays the following log output:

```
Started by user holocron
Building remotely on jenkins-build-slave-cf2c5325a691 (*.Build) in workspace /home/jer
9dc8-481a-8b1d-84636f979585
Cloning the remote Git repository
Cloning repository https://hub.jazz.net/git/holocron/cdc-lab-5d
Fetching upstream changes from https://hub.jazz.net/git/holocron/cdc-lab-5d
using .gitcredentials to set credentials
Checking out Revision 8d1d13ae59b7473984b97438cf452037ad5c853 (detached)
First time build. Skipping changelog.
[e12acd0e-9dc8-481a-8b1d-84636f979585] $ /bin/bash /tmp/hudson1947265415289923303.sh
express@4.12.4 node_modules/express
  |-- merge-descriptors@1.0.0
  |-- utils-merge@1.0.0
  `-- cookie-signature@1.0.6
```

Exercise 5.4.4: Triggering the build stage by pushing to the Git repository

1. Return to the Web GUI editor by clicking **EDIT CODE**.



2. Alter the project by opening the public/index.html file and changing the <h1> tag to something new.

```
index.html
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <title>NodeJS Starter Application</title>
6     <meta charset="utf-8">
7     <meta http-equiv="X-UA-Compatible" content="IE=edge">
8     <meta name="viewport" content="width=device-width, initial
9       <link rel="stylesheet" href="stylesheets/style.css">
10
11
12   <body>
13     <table>
14       <tr>
15         <td style="width:30%;">
16           NodeJS Starter Application</s
21           Get started by reading our
22           <a href = "https://www.ng.bluemix.net/docs/#start
23           or use the Start Coding guide under your app in yo
24           </table>
25
26
27
28 </html>
```

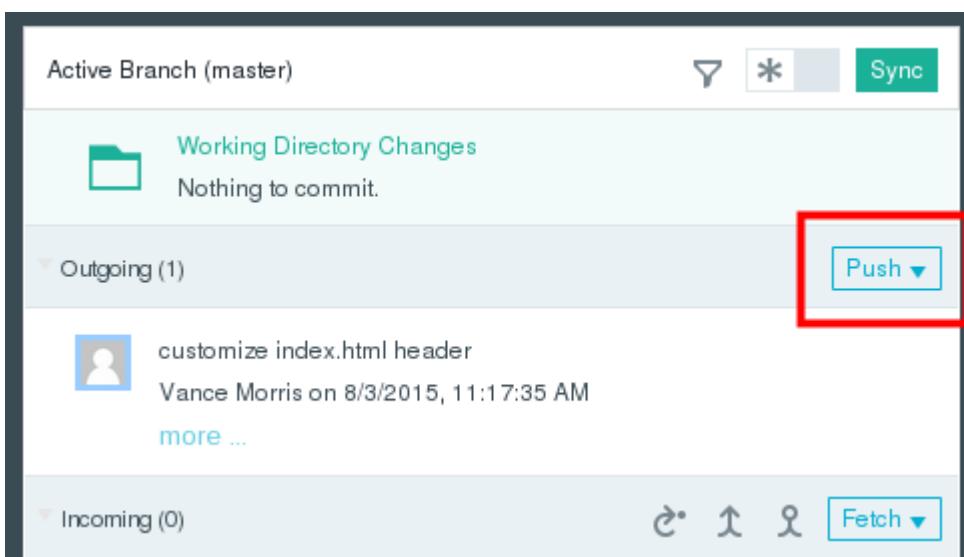
Exercise 5.4: Managing source code for projects

3. Switch to the Git repository view by clicking the **Git** icon in the left column.

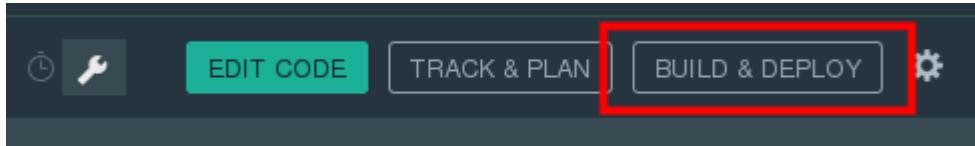


Note that the SCM detected a change in the working directory.

4. Enter an informative commit message, select the check box next to **index.html**, and then click **COMMIT**. The outgoing commit is now staged, and ready to push.
5. Immediately after pushing the commit, switch over to the **BUILD AND DEPLOY** view. You will need to be quick to see the builder stage automatically start, but if you miss it, you can always view the details of the build in the history view.
6. Click **Push** and then immediately click **BUILD AND DEPLOY**.



Exercise 5.4: Managing source code for projects



Observe the automatic execution of the build stage.