Search

Introduction to OpenShift Applications

Tell us what you think!

Congratulations on completing at least 25% of the course. We would appreciate feedback on your learning experience. Click below to fill out a brief survey.

TAKE ME TO THE SURVEY (HTTPS://SURVEY.OLE.REDHAT.COM/SURVEY?COURSE=DO101&MODALITY=COURSE&OFFERING=101)

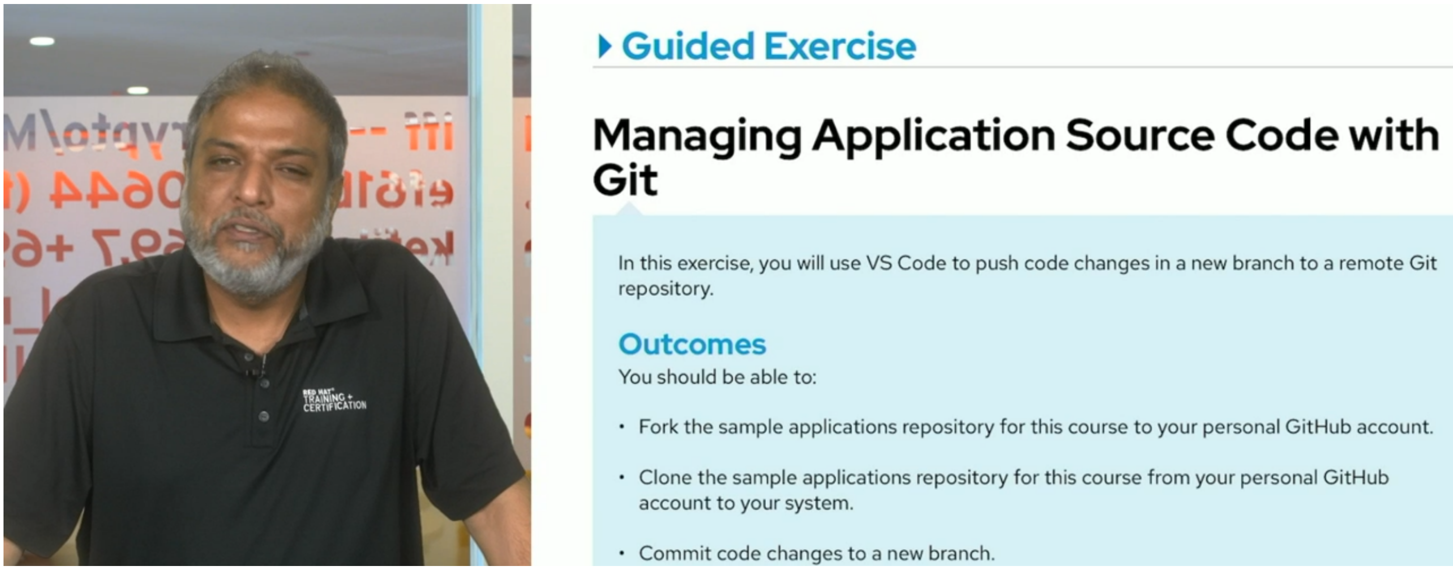
NOT RIGHT NOW

□ Don't ask again

TRANSLATIONS ▾

← PREVIOUS (/ROL/APP/COURSES/DO101-4.2/PAGES/CH01S05) → NEXT (/ROL/APP/COURSES/DO101-4.2/PAGES/CH01S07)

Guided Exercise: Managing Application Source Code with Git ☆



▸ Guided Exercise

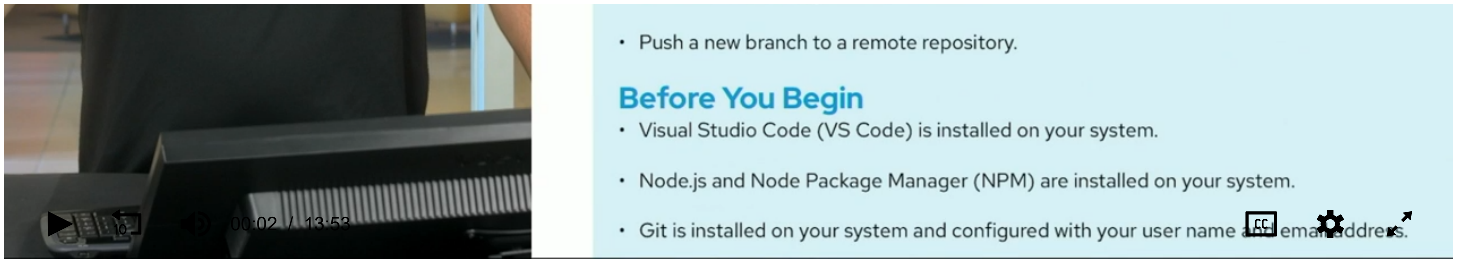
Managing Application Source Code with Git

In this exercise, you will use VS Code to push code changes in a new branch to a remote Git repository.

Outcomes

You should be able to:

- Fork the sample applications repository for this course to your personal GitHub account.
- Clone the sample applications repository for this course from your personal GitHub account to your system.
- Commit code changes to a new branch.



- Push a new branch to a remote repository.

Before You Begin

- Visual Studio Code (VS Code) is installed on your system.
- Node.js and Node Package Manager (NPM) are installed on your system.
- Git is installed on your system and configured with your user name and email address.

In this exercise, you will use VS Code to push code changes in a new branch to a remote Git repository.

Outcomes

You should be able to:

- Fork the sample applications repository for this course to your personal GitHub account.
- Clone the sample applications repository for this course from your personal GitHub account to your system.
- Commit code changes to a new branch.
- Push a new branch to a remote repository.
- Visual Studio Code (VS Code) is installed on your system.
- Node.js and Node Package Manager (NPM) are installed on your system.
- Git is installed on your system and configured with your user name and email address.

Procedure 1.3. Steps

1. Fork the sample applications for this course into your personal GitHub account.

   1.1. Open a web browser, and navigate to https://github.com/RedHatTraining/DO101-apps (https://github.com/RedHatTraining/DO101-apps). If you are not logged in to GitHub, then click **Sign in** in the upper-right corner.

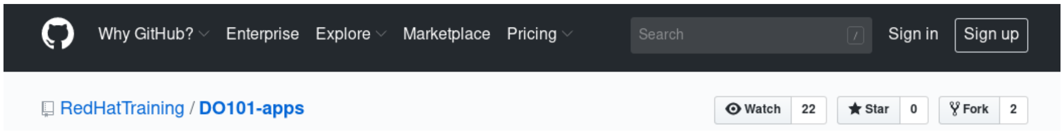
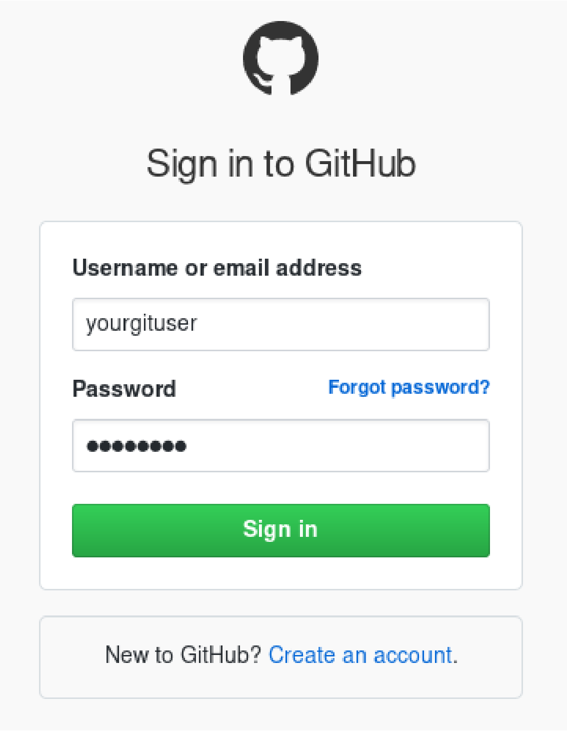


Figure 1.46: Sign-in page for the course Git repository.

   1.2. Log in to GitHub using your personal user name and password.

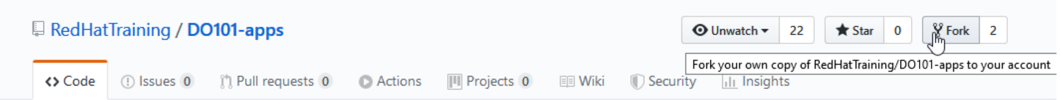


Username or email address: yourgituser

Password: ●●●●●●●●

Figure 1.47: GitHub Sign-in page.

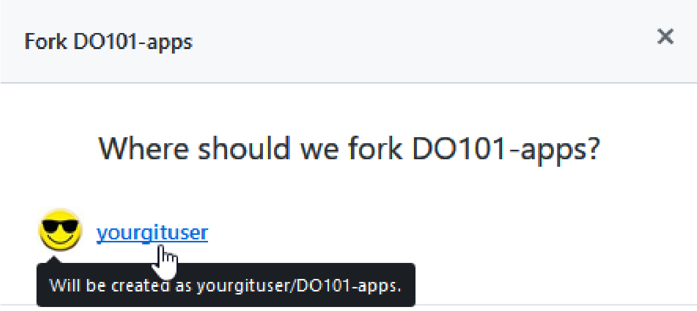
1.3. Return to https://github.com/RedHatTraining/DO101-apps (https://github.com/RedHatTraining/DO101-apps) and click **Fork** in the upper-right corner.



Fork your own copy of RedHatTraining/DO101-apps to your account

Figure 1.48: Fork the Red Hat Training DO101-apps repository.

1.4. In the `Fork DO101-apps` window, click *`yourgituser`* to select your personal GitHub project.



Will be created as yourgituser/DO101-apps.

Figure 1.49: Fork the Red Hat Training DO101-apps repository.

> **IMPORTANT**
>
> While it is possible to rename your personal fork of the https://github.com/RedHatTraining/DO101-apps (https://github.com/RedHatTraining/DO101-apps) repository, the example output in this course assumes that you retain the name `DO101-apps` when you fork the repository.

1.5. After a few minutes, the GitHub web interface displays your new *`yourgituser`*`/DO101-apps` repository.

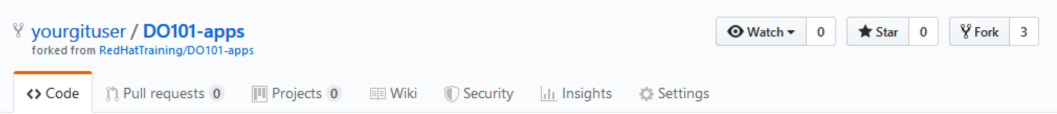


Figure 1.50: A personal fork of the Red Hat Training DO101-apps repository.

2. Clone the sample applications for this course from your personal GitHub account using VS Code.

   2.1. If you do not have VS Code open from a previous exercise, the open it.

   2.2. In the Command Palette (**View** → **Command Palette...**), type `clone`. Select **Git: Clone** from the list of options.

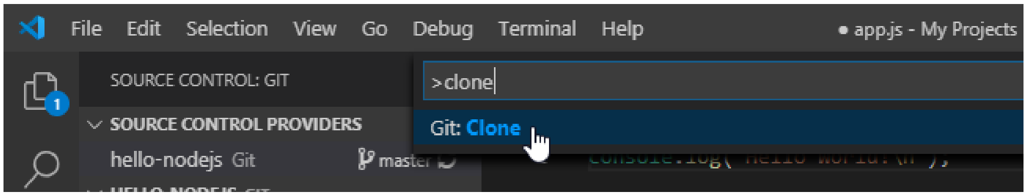


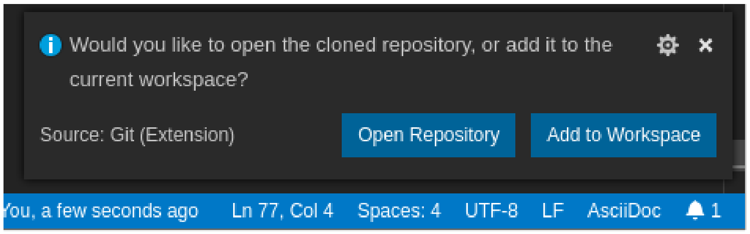
Figure 1.51: Using the command palette to clone a repository.

   2.3. In the prompt that displays, enter the HTTPS URL for your repository, `https://github.com/`*`yourgituser`*`/DO101-apps`.

   2.4. In the file window that opens, choose a local folder to store the repository clone. VS Code creates a `DO101-apps` subfolder in the folder you select. The default location is your home folder.

   Click **Select Repository Location** to select your location.

   2.5. After VS Code clones the repository, a prompt displays in the lower-right corner of the VS Code window. Click **Add to Workspace** to add the cloned repository to your VS Code workspace.


Would you like to open the cloned repository, or add it to the current workspace?

Source: Git (Extension)

Figure 1.52: Add a cloned repository to the current workspace.

NOTE

This prompt remains active for only a few seconds. If the prompt closes, add the cloned repository folder to your workspace (**File** → **Add Folder to Workspace...**). In the file window that displays, navigate to the location of the cloned `DO101-apps` folder. Select the `DO101-apps` folder, and then click **Add**.

3. Add a feature to the `express-helloworld` application to display `Hello Mars!` when a user accesses the `/mars` application endpoint. Implement the new feature in the `devenv-versioning` branch.

   3.1. From the Source Control view (**View** → **SCM**), click **master** in the `DO101-apps` entry under `SOURCE CONTROL PROVIDERS`.

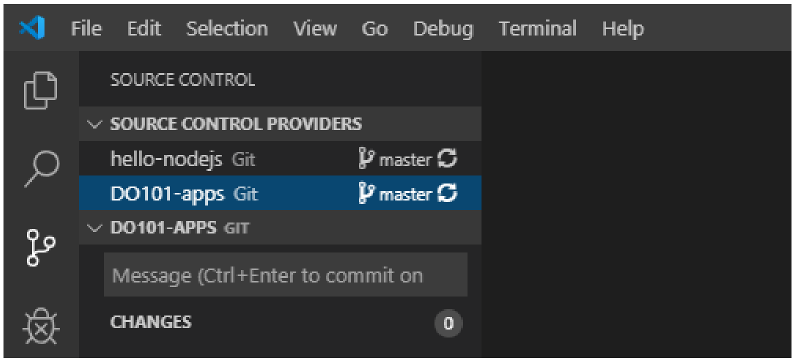


Figure 1.53: Checkout a new branch for a repository.

> **NOTE**
>
> If the Source Control view does not display the `SOURCE CONTROL PROVIDERS` heading, right-click `SOURCE CONTROL` at the top of the Source Control view and select `Source Control Providers`.

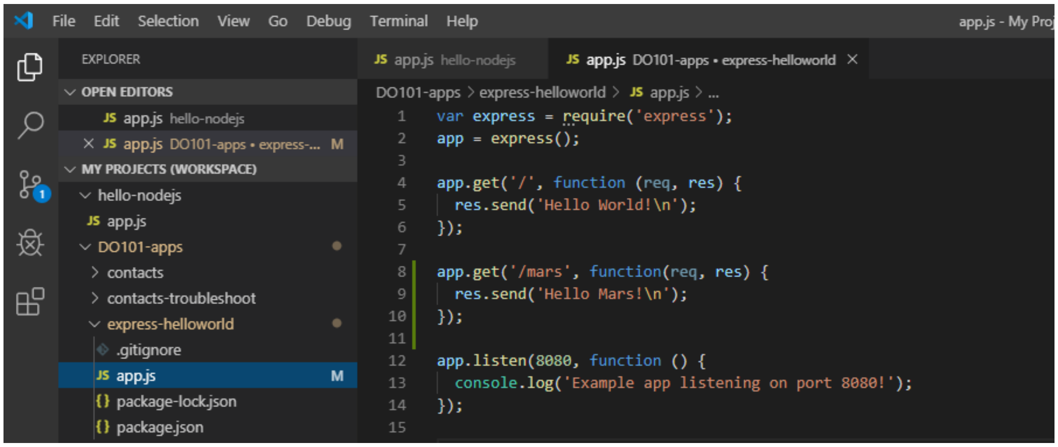
3.2. Select `Create new branch…` from the list of options.

3.3. When prompted, enter `devenv-versioning` for the branch name. The `DO101-apps` entry in the Source Control view updates to the `devenv-versioning` branch.

3.4. In the Explorer view (**View** → **Explorer**), click the `DO101-apps/express-helloworld/app.js` file. VS Code opens an editor tab for the file.

3.5. To implement the new feature, add the following lines to the `app.js` file:

Insert these lines before the line that begins with `app.listen`:



```
var express = require('express');
app = express();

app.get('/', function (req, res) {
  res.send('Hello World!\n');
});

app.get('/mars', function(req, res) {
  res.send('Hello Mars!\n');
});

app.listen(8080, function () {
  console.log('Example app listening on port 8080!');
});
```

Figure 1.54: An added feature to the Express application.

Save the `app.js` file (**File** → **Save**).

SHOW SOLUTION

4. Install the application dependencies and execute the application. Verify that the application returns a `Hello Mars!` message when you access the `/mars` end point.

> **NOTE**

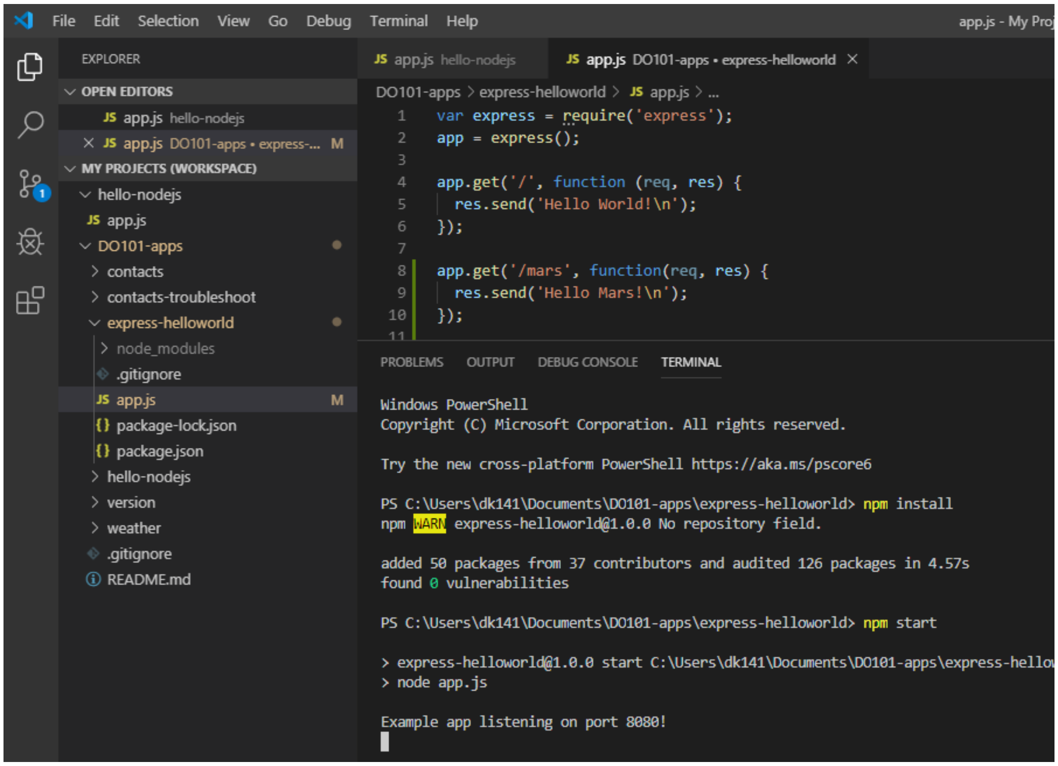
This step requires the Node Package Manager (NPM). When you install the Node.js package on Ubuntu systems, NPM is not installed as a dependency. Thus, you must also install the NPM package on Ubuntu systems.

The Ubuntu NPM package installs several different software development packages. You can skip this step if you need to minimize the number of installed packages on your Ubuntu system. Skipping this step does not prevent you from completing this exercise.

4.1. Right-click `express-helloworld` in the Explorer view (**View** → **Explorer**) and then select `Open in Terminal`.

4.2. Execute the `npm install` command in the integrated terminal to install the application dependencies.

4.3. Execute the `npm start` command in the integrated terminal to start the application.



```
var express = require('express');
app = express();

app.get('/', function (req, res) {
  res.send('Hello World!\n');
});

app.get('/mars', function(req, res) {
  res.send('Hello Mars!\n');
});
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\dk141\Documents\DO101-apps\express-helloworld> npm install
npm WARN express-helloworld@1.0.0 No repository field.

added 50 packages from 37 contributors and audited 126 packages in 4.57s
found 0 vulnerabilities

PS C:\Users\dk141\Documents\DO101-apps\express-helloworld> npm start

> express-helloworld@1.0.0 start C:\Users\dk141\Documents\DO101-apps\express-hellow
> node app.js

Example app listening on port 8080!
```

Figure 1.55: Executing a Node.js application from the integrated terminal.

4.4. In a browser, navigate to http://localhost:8080/ (http://localhost:8080/). Verify that the application responds with a `Hello World!` message.

4.5. In a browser, navigate to http://localhost:8080/mars (http://localhost:8080/mars). Verify that the application responds with a `Hello Mars!` message.

4.6. To stop the application, click in the integrated terminal and type **Ctrl**+**C**. If a prompt displays, then provide an appropriate response to terminate the process.

4.7. To clean up, click the **Kill Terminal** icon to close the integrated terminal window.

5. Commit your changes locally, and then push the new commit to your GitHub repository.

5.1. Access the Source Control view (**View** → **SCM**) and then click `DO101-apps` in the `SOURCE CONTROL PROVIDERS` list.

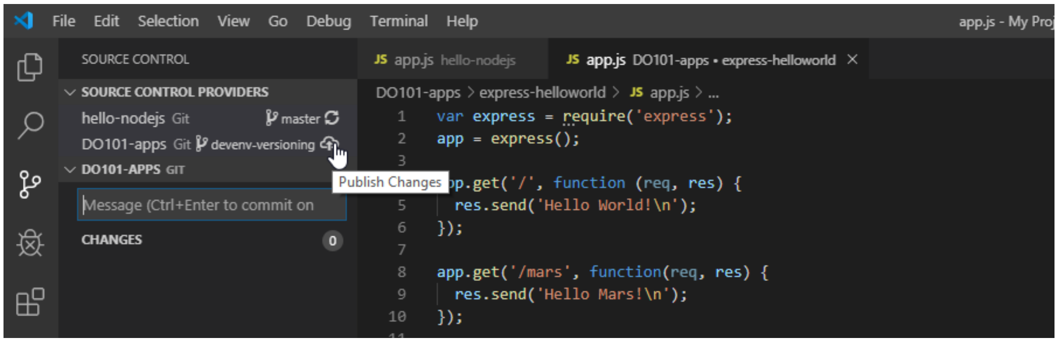
5.2. Hover over the entry for the `app.js` file under the `CHANGES` heading for the `DO101-apps` repository. Click the `Stage Changes` icon to add the changes to the `app.js` file to your next commit.

5.3. Add a commit message of `add /mars endpoint` in the message prompt, and then click the check mark button to commit the staged changes:

Figure 1.56: A commit message for a new feature.

6. Publish the `devenv-versioning` branch to your GitHub repository. Verify that your changes are present on GitHub.

   6.1. In the Source Control view (**View → SCM**), locate the `DO101-apps` entry under the `SOURCE CONTROL PROVIDERS` heading. Click the `Publish Changes` icon to publish the `devenv-versioning` branch to the remote repository. If a prompt displays, then provide your GitHub user name and password.



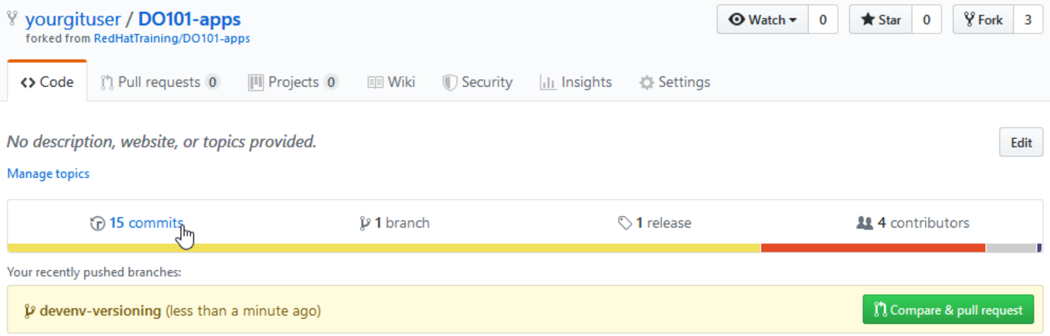
```
var express = require('express');
app = express();

app.get('/', function (req, res) {
  res.send('Hello World!\n');
});

app.get('/mars', function(req, res) {
  res.send('Hello Mars!\n');
});
```

Figure 1.57: Publish a local branch to a remote repository.

   6.2. In a browser, navigate to your personal `DO101-apps` repository at `https://github.com/`*`yourgituser`*`/DO101-apps`. The GitHub interface indicates that you recently pushed code changes to the `devenv-versioning` branch.



No description, website, or topics provided.

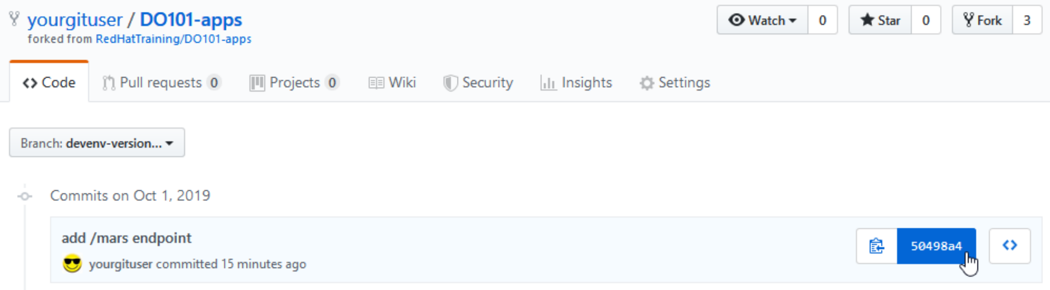
Your recently pushed branches:

devenv-versioning (less than a minute ago)

Figure 1.58: A GitHub repository indicates the presence of a new branch.

   6.3. Click `commits` below the `Code` tab to display a list of recent commits.

   6.4. From the branch list, select `devenv-versioning`. An entry for the `add /mars endpoint` commit displays at the top of the commit history.



Commits on Oct 1, 2019

add /mars endpoint

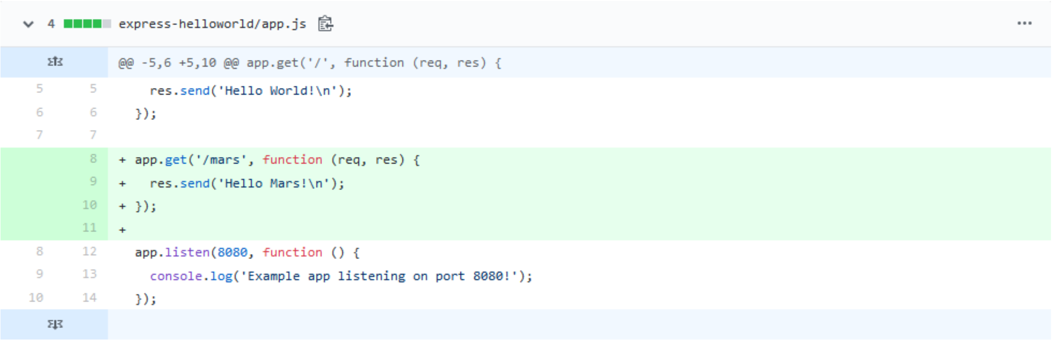
yourgituser committed 15 minutes ago

50498a4

Figure 1.59: The commit history for a GitHub repository branch.

   6.5. To the right of the `add /mars endpoint` entry, click the seven character commit hash. In the example shown, this value is *`50498a4`*.

   GitHub displays the four lines of code that you added to the `express-helloworld/app.js` file.



4 express-helloworld/app.js

```
@@ -5,6 +5,10 @@ app.get('/', function (req, res) {
    res.send('Hello World!\n');
  });

+ app.get('/mars', function (req, res) {
+   res.send('Hello Mars!\n');
+ });
+
  app.listen(8080, function () {
    console.log('Example app listening on port 8080!');
  });
```

Figure 1.60: GitHub displays commit code changes.

You successfully pushed your code changes in a new branch to your GitHub repository.

This concludes the guided exercise.

🡐 PREVIOUS (/ROL/APP/COURSES/DO101-4.2/PAGES/CH01S05) 🡒 NEXT (/ROL/APP/COURSES/DO101-4.2/PAGES/CH01S07)