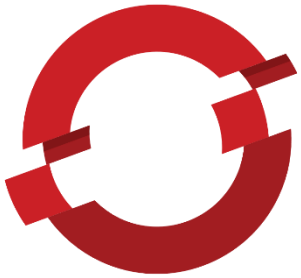


RED HAT®
OPENSHIFT.io

Getting Started with OpenShift.io

Early Access Program instructions

RED HAT®
DEVELOPER
PROGRAM

Document version 1.0-beta1
17-November-2017

Legal Notice

Table of Contents

Welcome

Thank you for your interest in the OpenShift.io Early Access Program and providing us feedback during this pre-beta phase of development! The purpose of this document is to guide you through your first experiences with OpenShift.io (OSIO). Within minutes you'll have an application running in the cloud and shortly thereafter you will have customized and deployed a new version of that application.

As a member of the Early Access Program, approach OpenShift.io with a willingness for adventure. The service is "pre-beta" quality. Updates and changes to the system are delivered continuously -- often dozens of times per day. Consequently, features and elements of the user interface change often. And, yes, sometimes bugs are a part of that delivery -- bugs which can, at times, be severe and take the entire system offline. However, as part of our continuous delivery model, everyone who pushes a change out to the live system assumes responsibility for the impact of that change, so even the severe bugs tend to be addressed quickly.

A more important aspect of the Early Aspect Program is to get **your feedback**. Tell us what you like, what you hate, what you wish you could do, what you wish the service would stop doing, etc. Everything is fair game. See the Giving Feedback section for information on how to contact us.

If your OpenShift.io environment gets into a state whereby you are finding it difficult or impossible to move forward, please refer to the Troubleshooting section of this document for information on how to "factory reset" your OpenShift.io environment.

Welcome to the world of OpenShift.io -- we hope you enjoy your experience and look forward to hearing your feedback!

About this document

The purpose of this getting started guide is to walk you through your initial set up of OpenShift.io and quickly expose you to some key functionality in OpenShift.io. If this document is successful, the majority of first-time OpenShift.io users will have a custom, albeit simple, application running on the internet in under one hour.

Changes to this document are frequent. To download the latest revision of this document, please visit: https://openshift.io/osiogettingstarted.pdf.

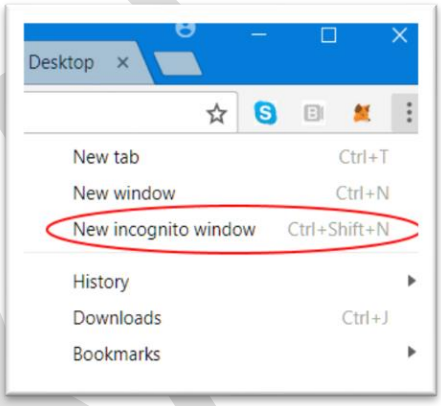
Prerequisites and Setup

Before your OpenShift.io experience can begin, it is necessary to perform some initial setup tasks. The great news is because you have found and are reading this document, you have already completed the first step!

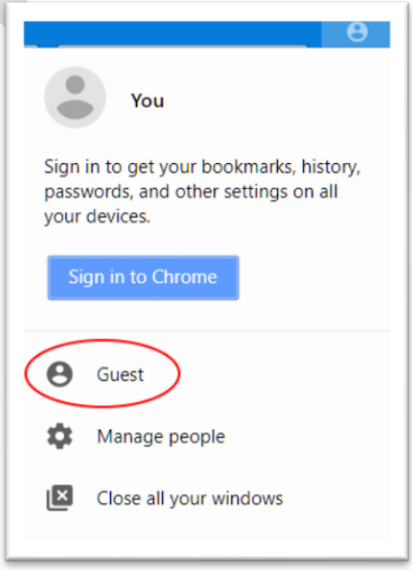
Browser requirements

During this pre-beta phase of development, we require all participants to use the Google Chrome browser, and to interact with OpenShift.io in either Incognito mode or with Chrome Guest credentials. If you ever have difficulties using OpenShift.io, this is the first thing to confirm -- that you are using Chrome in Incognito or Guest mode.

A. To use **Incognito mode**, from the Chrome menu select “New incognito window” option:

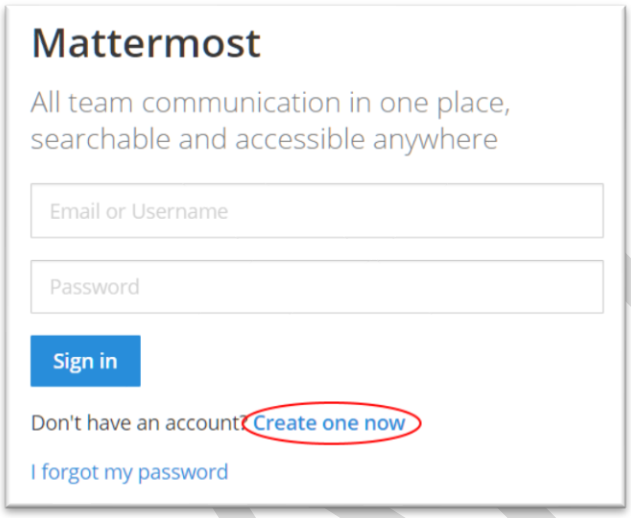


B. If you prefer to use Guest mode, click on the person/accounts icon in the browser’s title bar and select the Guest option:

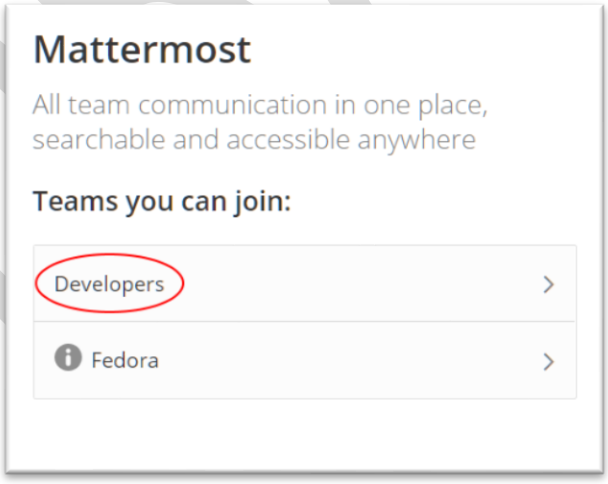


Mattermost Chat account set-up

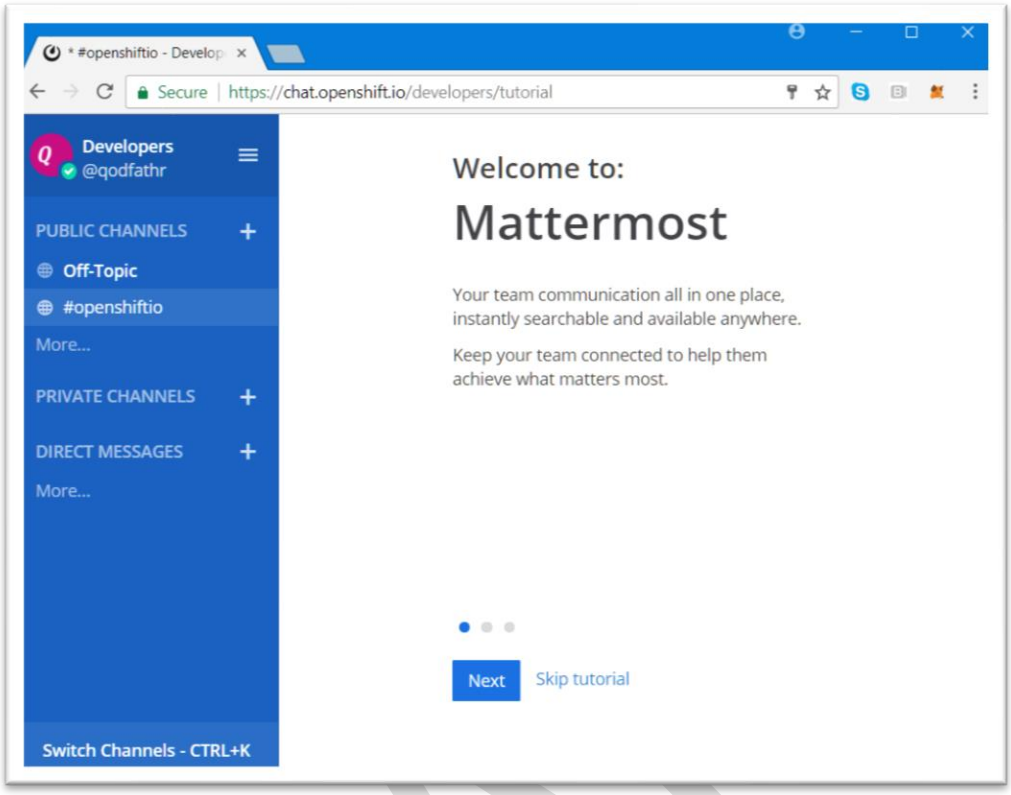
You'll be able to interact with the OpenShift.io team directly via chat. If you prefer to use IRC, you will find the team in the #openshiftio channel on freenode. Otherwise, please visit https://chat.openshift.io. You will see a login screen. Unfortunately, chat is not currently integrated with the OpenShift.io authentication system, so it is necessary for you to create a chat account. Click the **Create one now** link to create a new account:



You will go through a typical registration process, and you'll need to verify your email address. Upon completion of the process, you'll be able to log in. Upon your first login, you'll be likely asked to choose a Team to join. Please choose the Developers team:



Mattermost will offer you a tutorial; feel free to follow it (it is short) or simply click the **Skip tutorial** link:

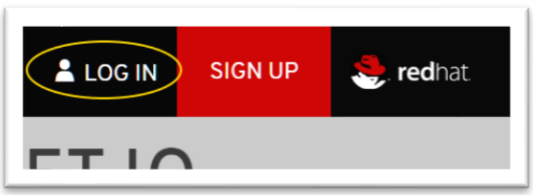


You will now be in the #openshiftio channel and ready to chat with the OpenShift.io team and other OpenShift.io users just like you.

First time OpenShift.io login

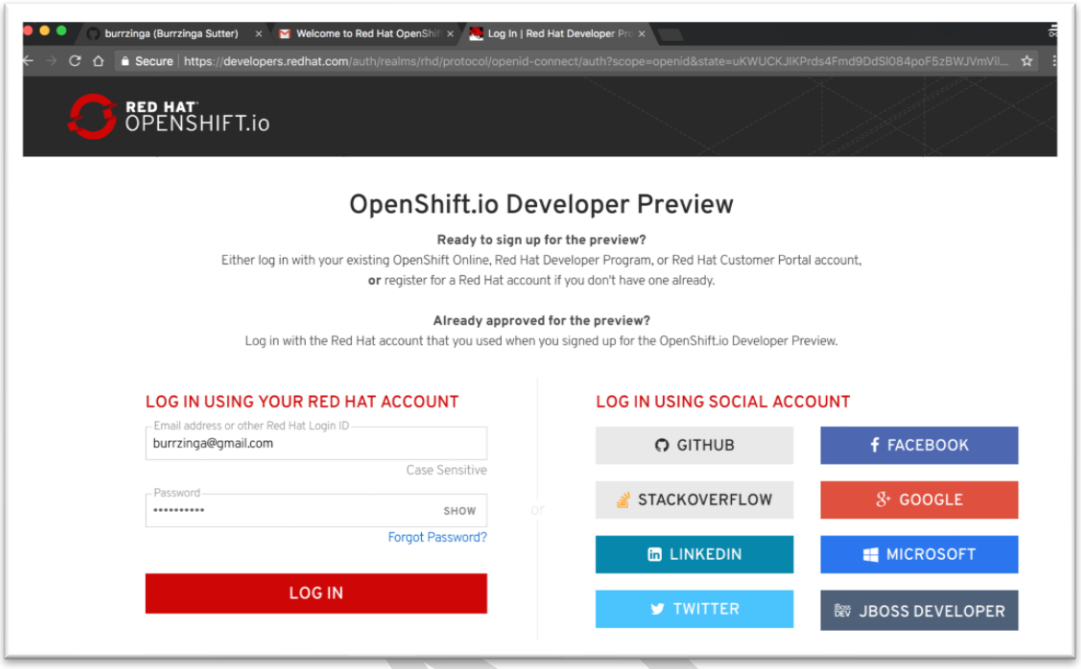
When you first login to OpenShift.io, you will have to grant OpenShift.io permission to interact with two remote services on your behalf -- (1) GitHub.com and (2) OpenShift Online. OpenShift.io will use GitHub to manage your source code; it will use OpenShift Online for many capabilities, most notably to run the applications you create.

In your browser (remember to either be in Incognito or Guest mode), navigate to openshift.io and click the "LOG IN" button in the upper right side of the page. (Don't see the button? Either make your window wider (recommended) or click the MENU button):

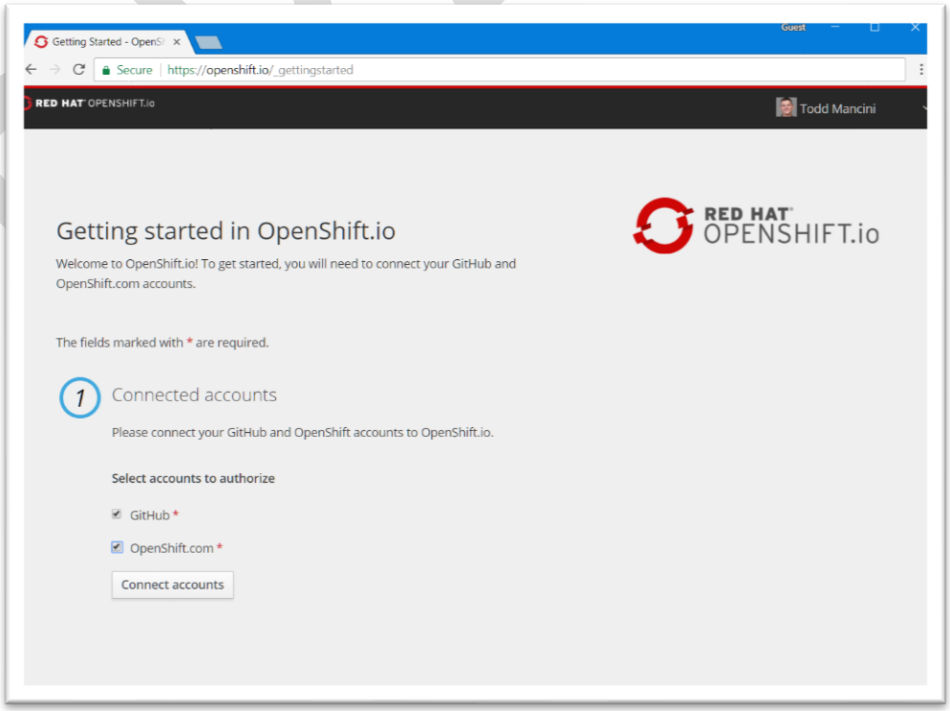


or

The login form will be displayed. Type in your login credentials and click the LOG IN button.

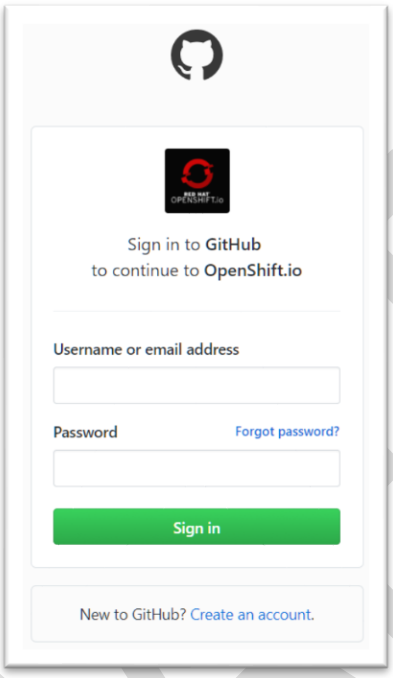


You will be asked to link your OpenShift.io account to your OpenShift.com and GitHub accounts. **Please select both checkboxes** under the ‘Select accounts to authorize’ section and click the ‘Connect accounts’ button.

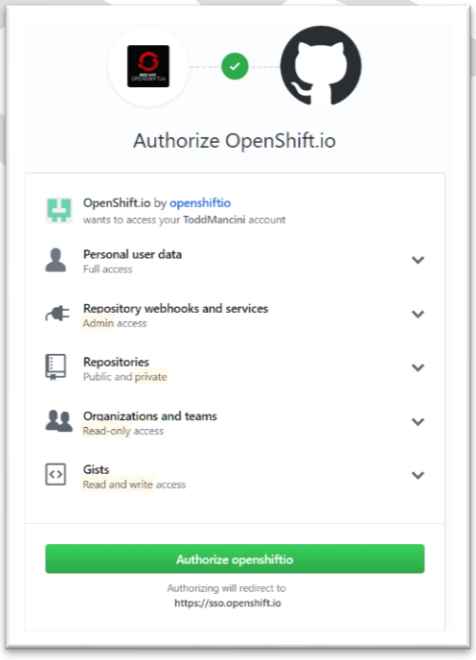


GitHub.com Authorization

GitHub will ask for your credentials. If you do not yet have a GitHub account, go ahead and create one now (it's free).



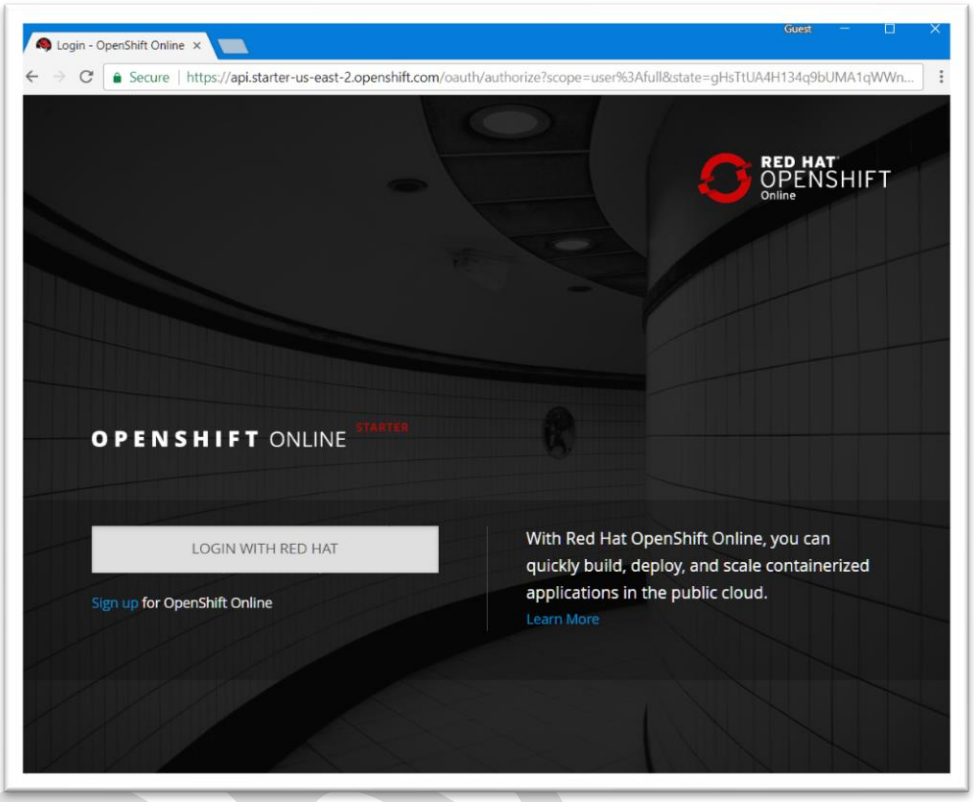
You will then be asked to authorize certain rights to OpenShift.io. Don't not change any of the requested settings and click the Authorize openshift.io button.



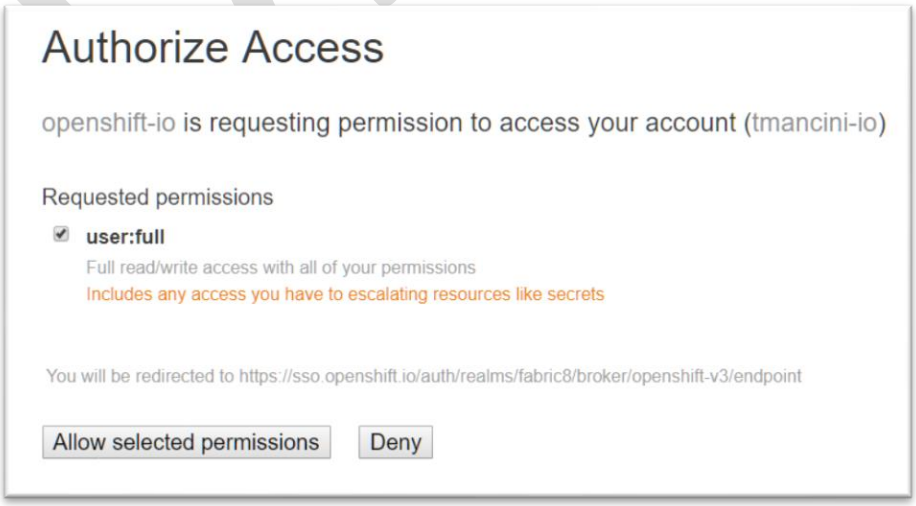
DURING THE EARLY ACCESS PROGRAM, OPENSHIFT.IO REQUESTS A LOT OF GITHUB PERMISSIONS. THE NUMBER OF REQUIRED PERMISSIONS WILL BE REDUCED BEFORE SERVICE LAUNCH.

OpenShift Online Authorization

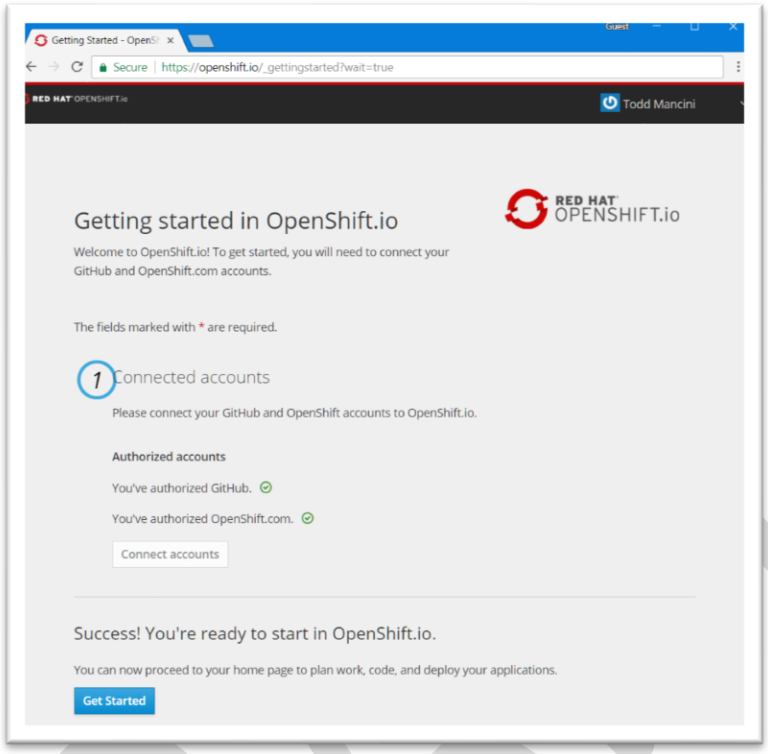
OpenShift Online will require you to log in. However, as your OpenShift.io account and your OpenShift Online account are the same federated identity, you should only need to click the LOGIN WITH RED HAT button and not need to supply any additional credentials.



OpenShift Online will confirm that you are granting the **user:full** permission to OpenShift.io. Leave the option checked and click the Allow selected permissions button.



If everything worked properly, you will see a success message and may click the Get Started button. If anything went wrong, you'll be required to attempt the previous steps again.



Giving Feedback

There are three primary ways to give us your feedback. We welcome any kind of feedback through any channel, although we have some guidelines on which avenues work best for different types of feedback.

Immediate Help -- Chat

The most direct way to engage with the OSIO team is to join us in our chat room. Once you have set up a chat account per the instruction in Mattermost Chat account set-up, you will have immediate access to the entire OSIO development team. Your feedback, questions, comments and concerns are always welcomed! Don't be shy – we'd rather you spend 3 minutes in chat trying to solve a problem instead of two hours debugging by yourself.

Bugs and Feature Requests -- GitHub Issues

If you'd like to file a bug or make a feature request, you may do so as a GitHub issue. This is also a great place to search to see if your issue has already been reported. If you create a new issue, do not worry about applying all of the correct labels; the OpenShift.io team will take care of that for you.

Other Feedback – Email

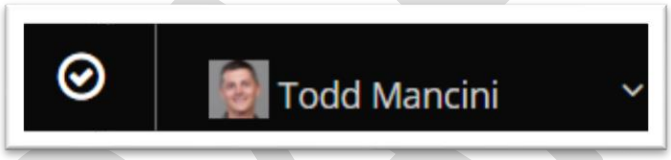
If you prefer to go the more traditional route of communicating via email, you may reach the OpenShift.io team via openshiftio@redhat.com.

OpenShift.io “Hello World”

Logging in

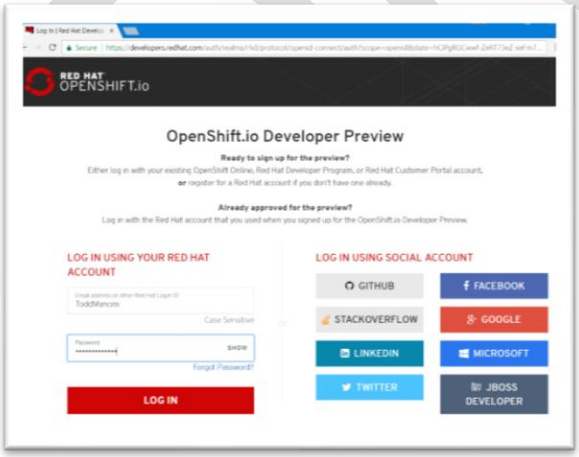
Adhering to the Browser requirements, start Google Chrome and navigate to https://openshift.io. Click the LOG IN button in the upper right-hand corner. (Refer to First time OpenShift.io login if you do not see a LOG IN button.)

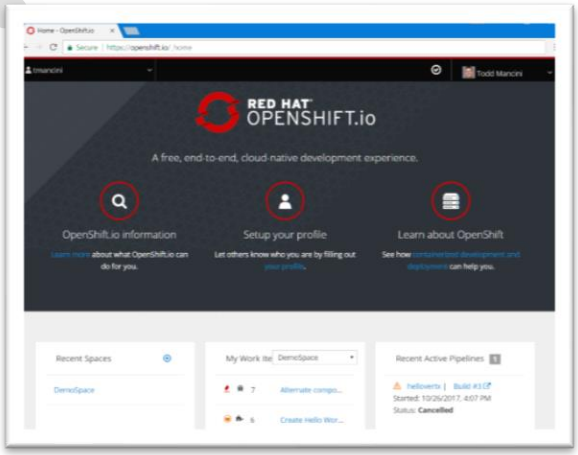
Note if instead of a log in button you are seeing something like this:



then you have been automatically logged in and may skip to the next section.

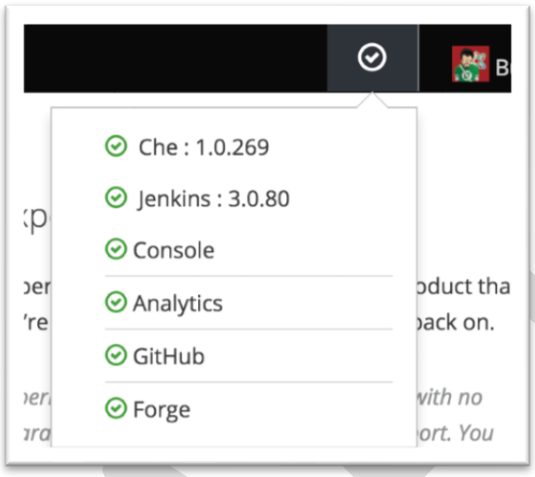
After you click the LOG IN button, you will be prompted to supply your credentials. Do so and you will be redirected to your OpenShift.io Account Home page.





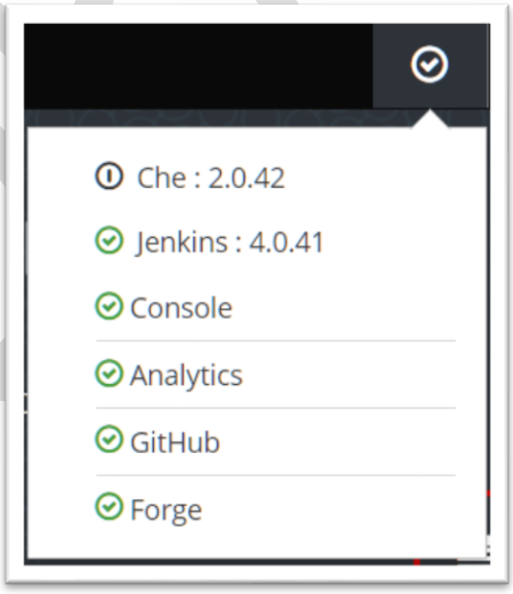
Checking Status of OpenShift.io Services

OpenShift.io is powered by several services. The operational status of these services may be checked by clicking the status widget in the upper-right corner of OpenShift.io. In general, you want to see all green checkmarks, such as this:



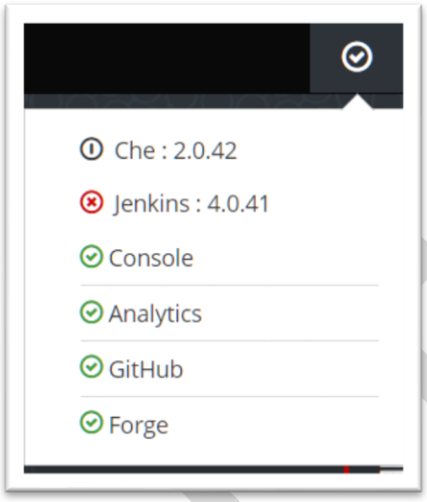
Some services, when marked with the green checkmark, provide additional information such as the current version of the service.

Some services – notably Che– are fine if they are in an idle state, as represented by a gray icon:



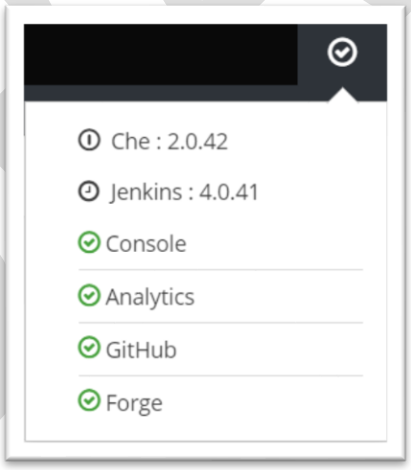
These services will awaken on-demand when needed.

You typically only need to be concerned if a service is showing as failed with a red X, as Jenkins is in this example:



If you see any failed services, the first thing you should do is simply wait. It is possible that the service is restarting. If after two minutes the service state has not changed, either jump on the OpenShift.io chat channel and request some assistance, or attempt the procedures as described in the Troubleshooting section.

A service which is starting will display with a gray clock icon, such as Jenkins in this example:

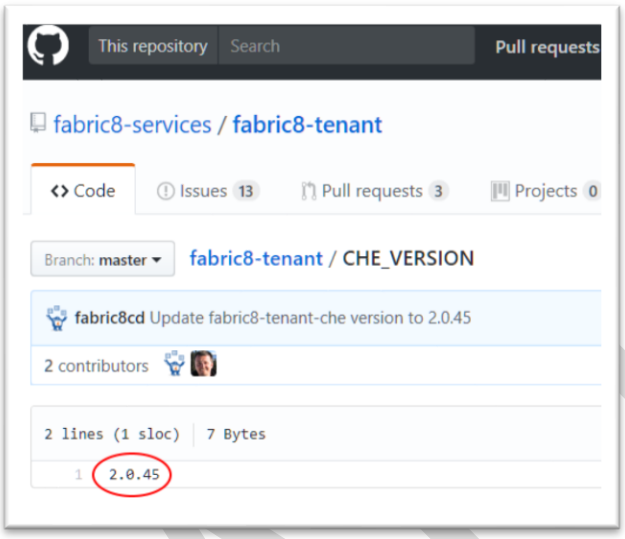


Are you curious about the version numbers for Che and Jenkins? These two services are updated frequently, and it's possible that your OpenShift.io account is not running the latest versions. If you'd like to see what is the most recent version numbers for these services, visit the following URLs:

- Che Version URL: https://github.com/fabric8-services/fabric8-tenant/blob/master/CHE_VERSION

- Jenkins Version URL: https://github.com/fabric8-services/fabric8-tenant/blob/master/JENKINS_VERSION

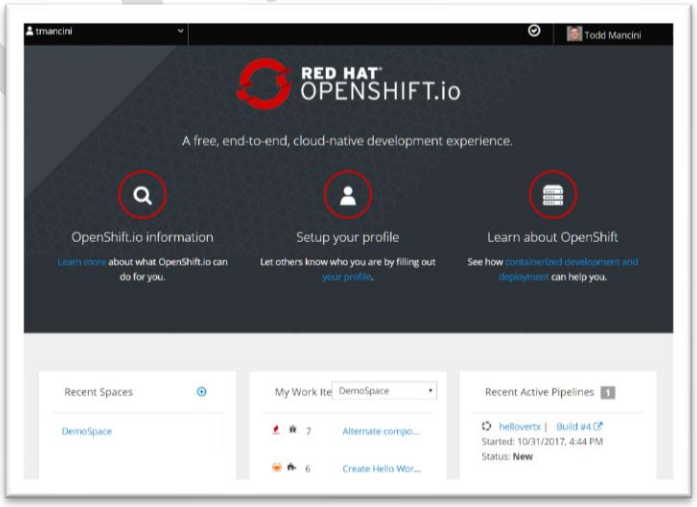
When visiting these URLs, you will see a screen like this, which indicates that the latest version of Che is 2.0.45:



As the status widget in the prior example is showing a Che version of 2.0.42, but the latest Che version is 2.0.45, an update is in order. Follow the instructions in Update Tenant Services to update all OpenShift.io services in your account.

Creating your First Space

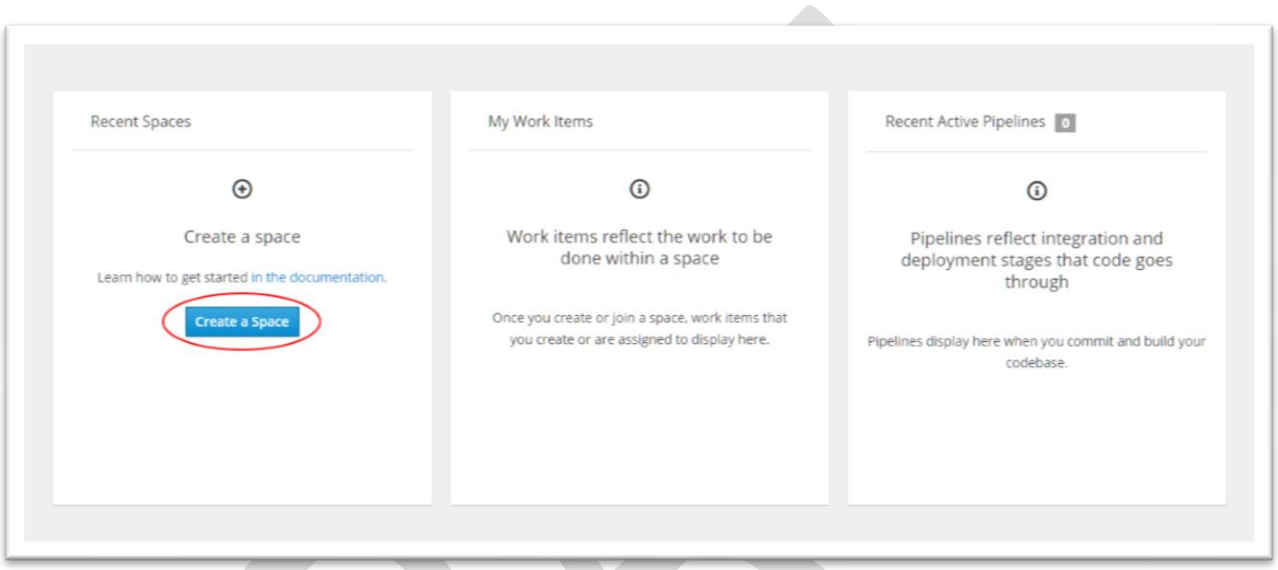
Once you log in, you see your account home dashboard. This is a quick overview of relevant and recent items created within your account. If this is your first time using OpenShift.io, there will be no data to see. Once you've been using OSIO, your dashboard will look more like this:



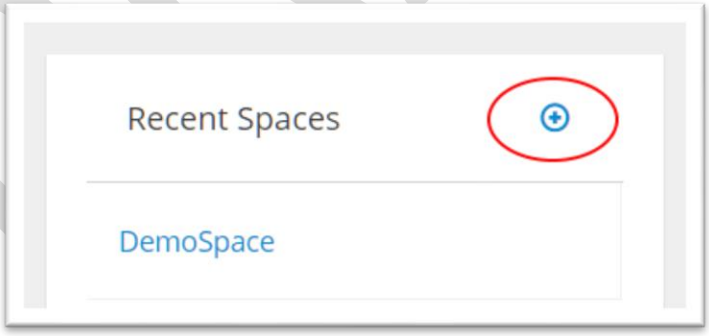
Your first action is to create an *OpenShift.io Space*. All the work that you will do in OSIO takes place in the context of a Space; a Space can be very small, such as containing a “hello world” example as we will create today, or a Space can be very large, containing all the development artifacts and team collaboration on a large development project.

OSIO requires that each Space that you create be given a unique name, so let’s create a new Space and give it a name.

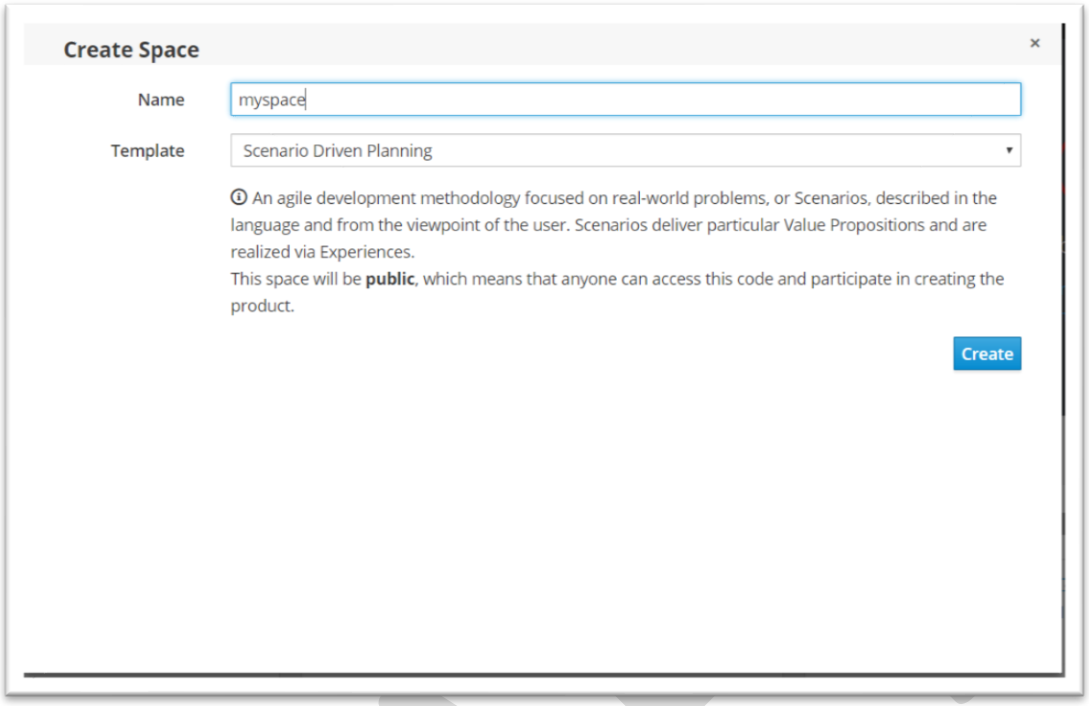
Scroll down the page and click the "Create a Space" button in the “Recent Space” tile.



If you already have some Spaces listed, click the  icon:



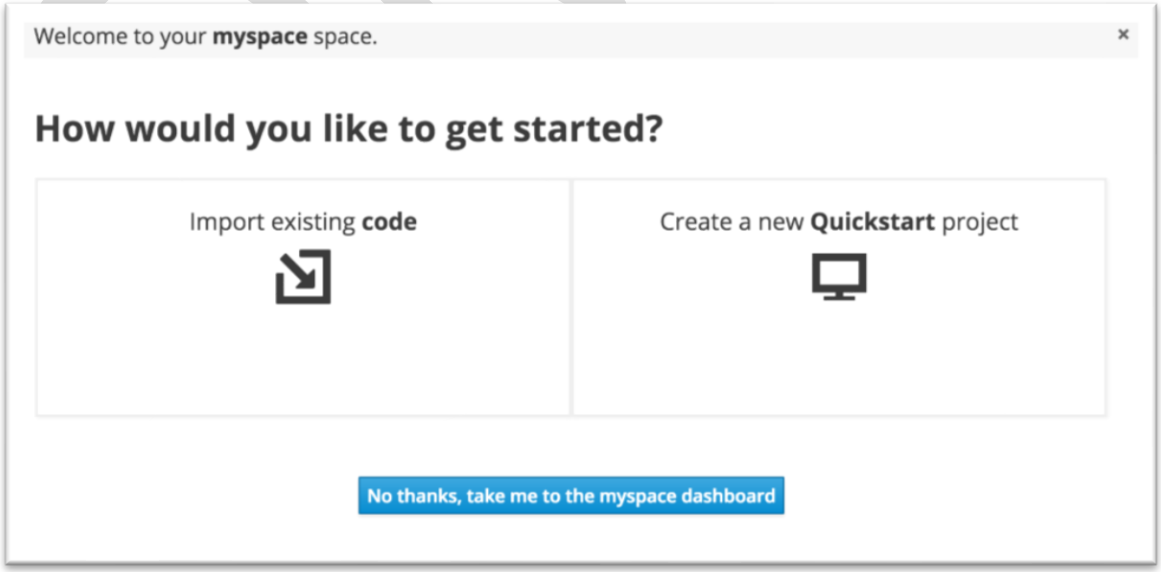
Either way, the Create Space dialog will appear:



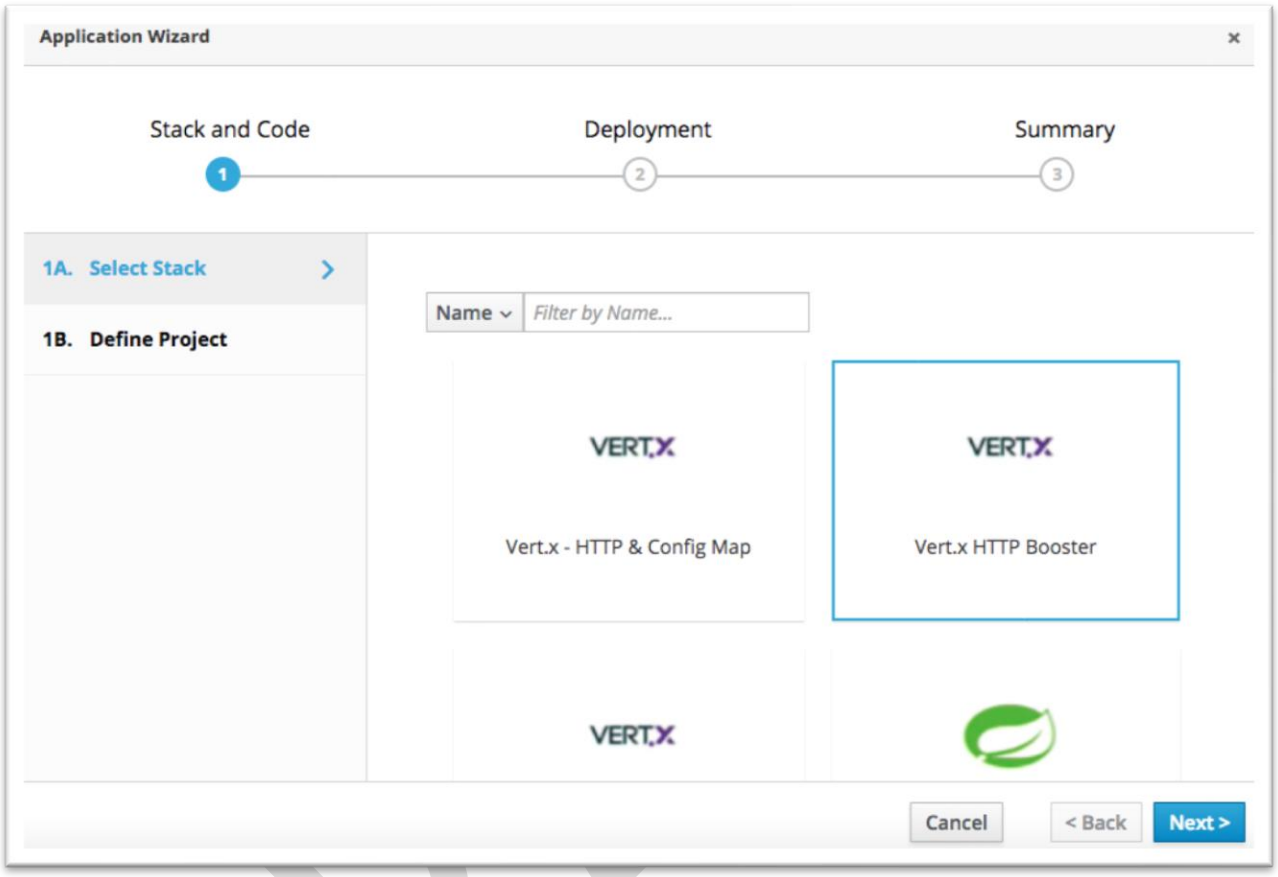
Give the Space a name of ‘myspace’ and click the Create button

Create a new Quickstart project

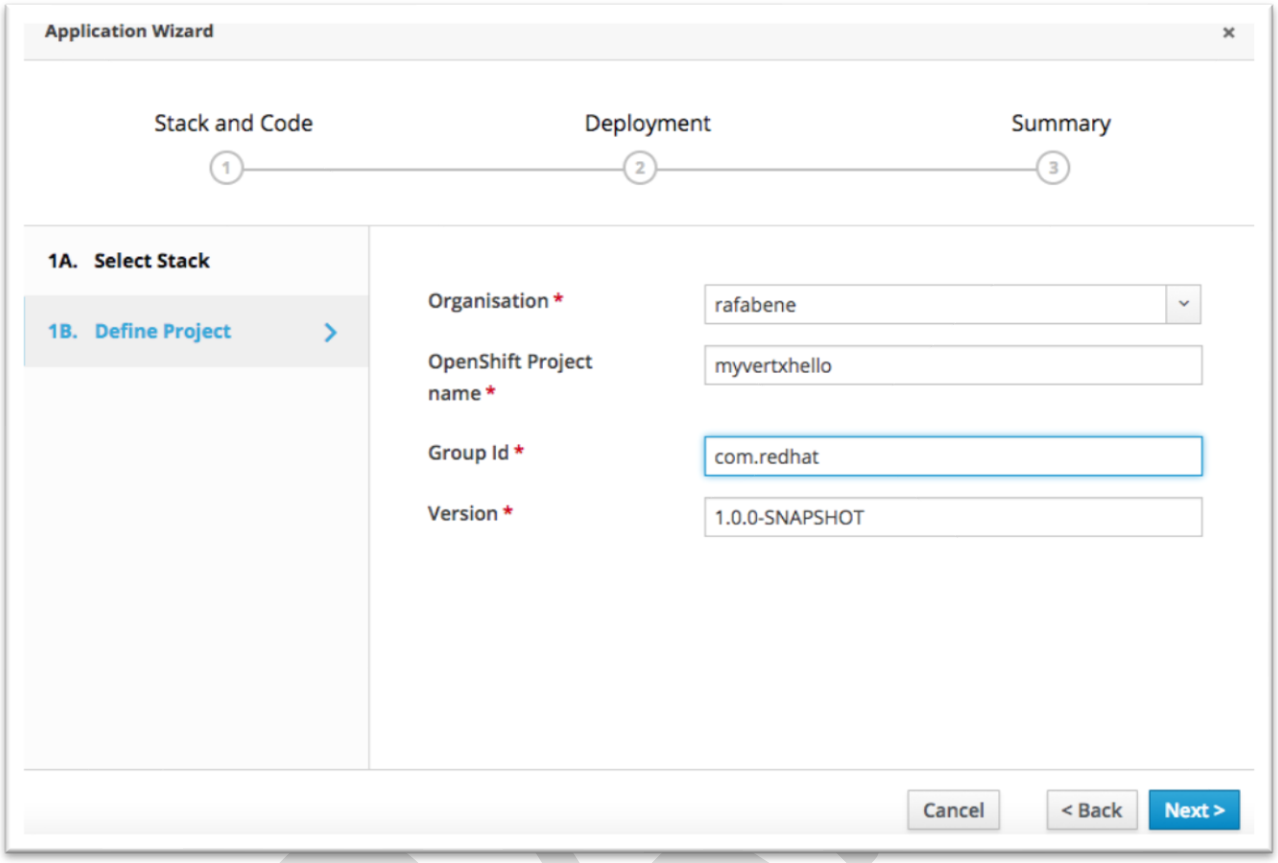
Your new Space is instantly created. To help you get started populating your Space, OSIO offers you two paths. The first lets you import code that you already have written and the second brings you through a guided experience to create a new software project via a “Quickstart.”



Click on the "Create a new **Quickstart** project". The Application Wizard appears.



Select "Vertx. HTTP Booster". Click "Next>".

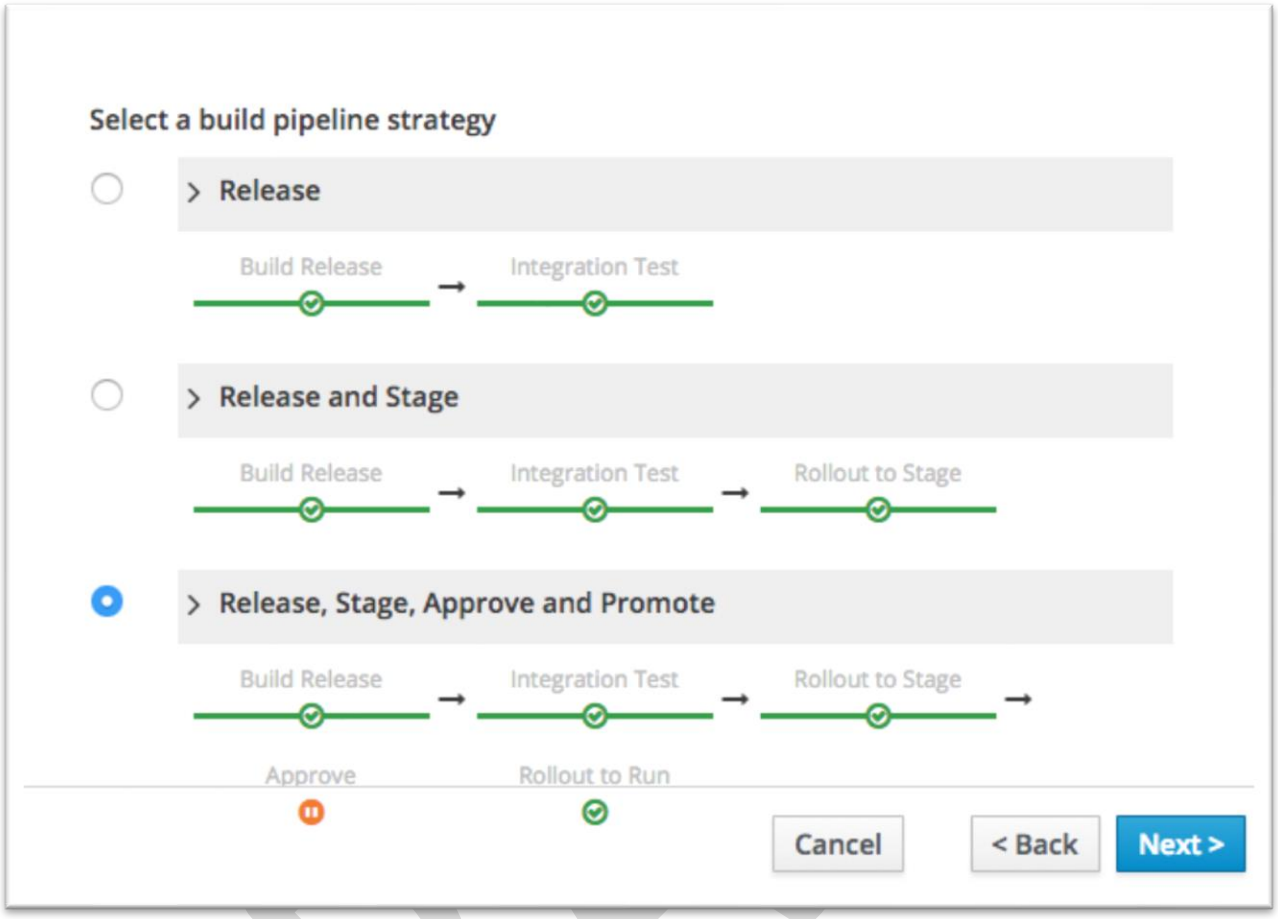


`

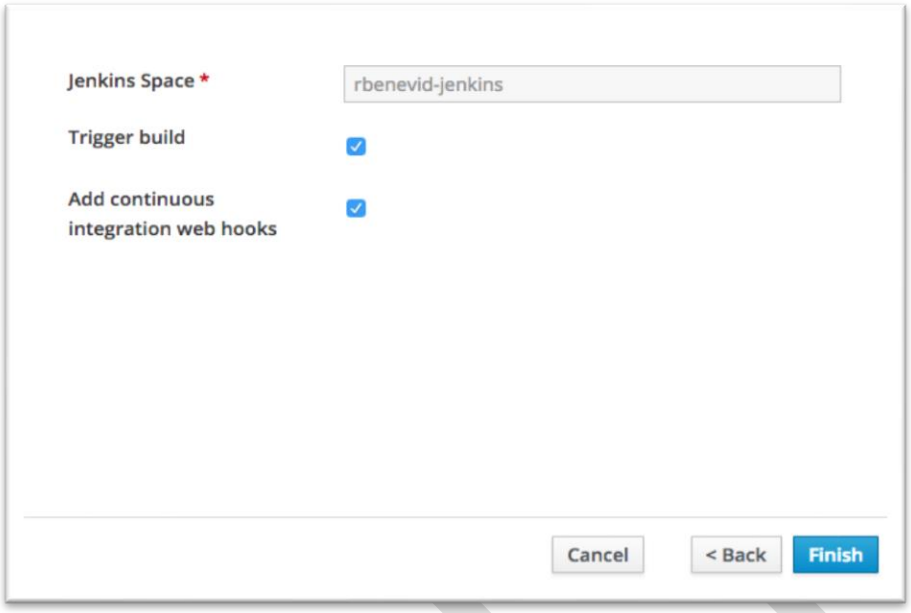
Step 1B of the Wizard appears. Fill in the fields as follows:

- **Organization:** This only appears if you are a member of member of multiple GitHub organizations. A new repository for the Quickstart project will be created in the organization you choose.
- **Name**: myvertxhello
- **Group Id:** com.redhat
- **Version**: 1.0.0-SNAPSHOT

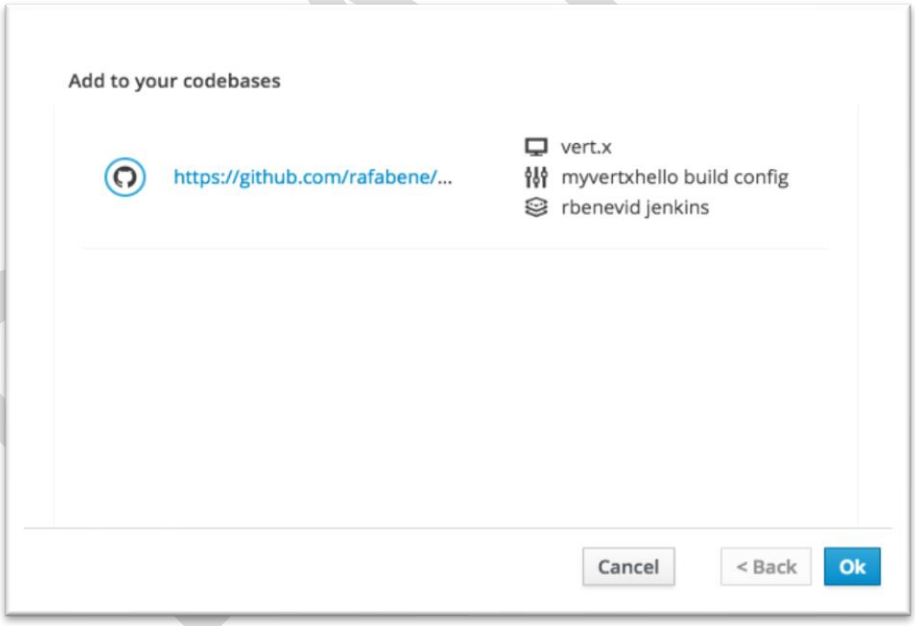
Click the "Next>" button.



You will now be asked to select a Pipeline strategy. Pipelines are automated continuous integration and continuous deployment processes (CI/CD), powered by Jenkins. The OpenShift.io built-in pipeline strategies enable anyone to create a repeatable, reliable, and incrementally improving process for taking this software from code to the proper execution environment. Select the “Release, Stage, Approve and Promote” pipeline selected and click the “Next>” button.



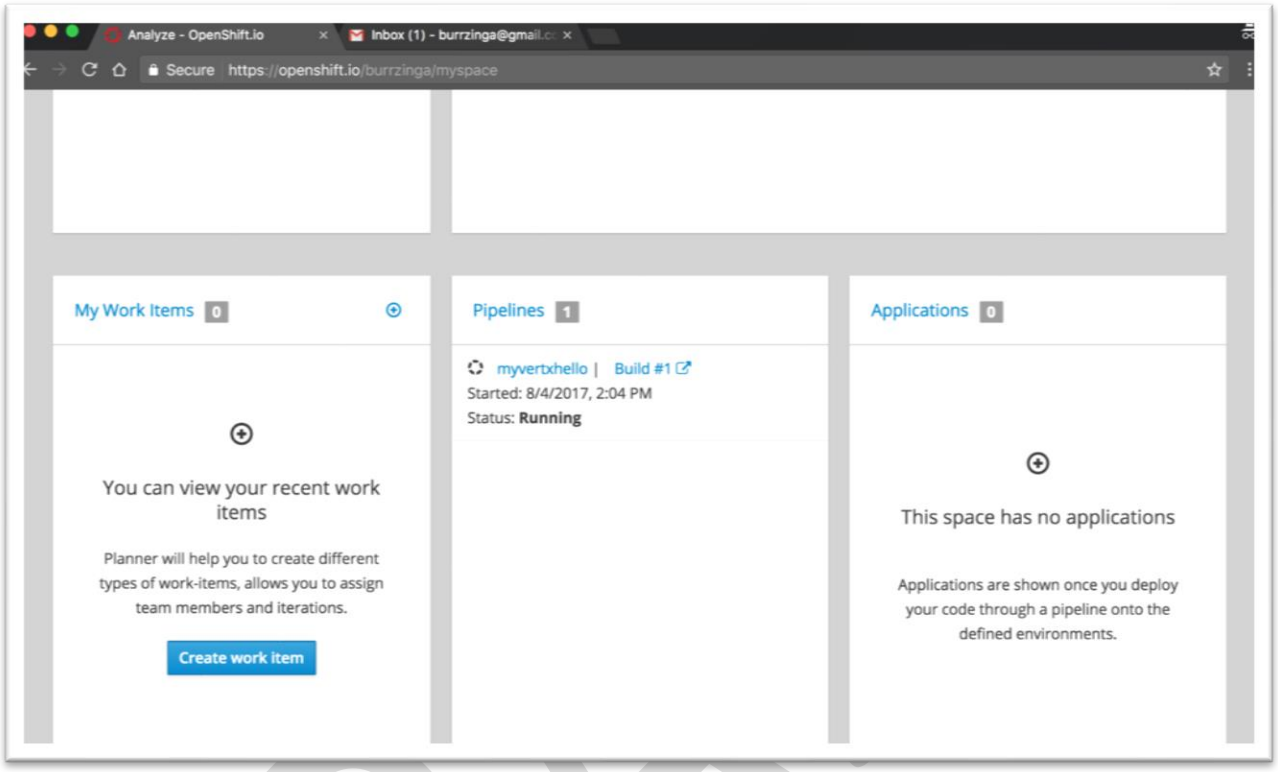
Leave the build configurations in the step 2B unchanged and click the Finish button.



Congratulations – the Quickstart project is created! The Quickstart project source code is hosted in the specified Github account and repository and the project's pipeline is hosted in OpenShift Online Starter.

Click "Ok" and you will see your *Space dashboard* with information about your Space. You will have one codebase and one pipeline. (Due to a known bug, you may have to refresh the page to see these). Since the pipeline template that was chosen implements CI/CD, combined with

the fact that new code was added to the code repository, the build pipeline starts executing immediately. Note that the pipeline animation indicates that it is currently executing.



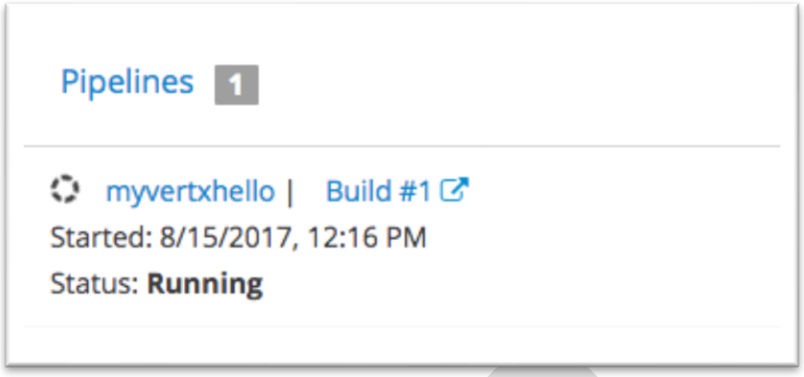
Environments

Before exploring your new Space, it is worth having a cursory understanding of *OpenShift.io Environments*. The free OpenShift Online Starter subscription included with your OpenShift.io subscription provides to you, among other things, two environments into which you may deploy your applications – Stage and Run. In general, your applications are first deployed to Stage so that you and other stakeholders may review them; then, if approved, the applications are promoted to Run. You may be curious why these two environments were not named Stage and *Production*. This is because the freely available OpenShift Online Starter does not support the running of production applications -- that requires you to purchase OpenShift Online Pro. The purpose of Stage and Run is to permit every developer to experience the modern workflow of pushing to a staging environment prior to production. In this sense, the Run environment is more akin to a pseudo-production environment, for educational purposes.

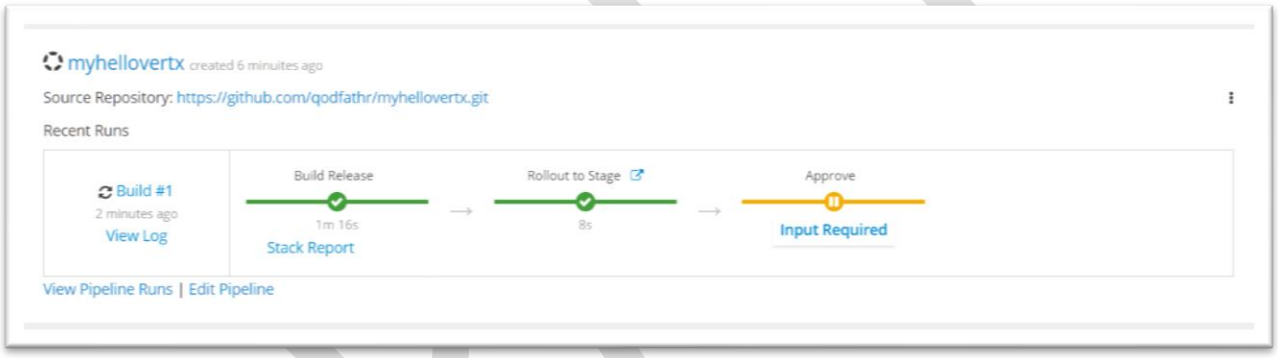
Pipelines

The selected Release, Stage, Approve and Promote pipeline is building your application and will deploy version 1.0.1 of your project into the Stage environment. The pipeline will then pause, awaiting your approval to deploy into the Run environment.

On the Space overview dashboard, notice the section named Pipelines:



Click on the word **Pipelines** to switch to the Pipelines screen see the execution details. Wait until the pipeline reaches the "Approvel" stage:



Accessing the Jenkins log -- optional

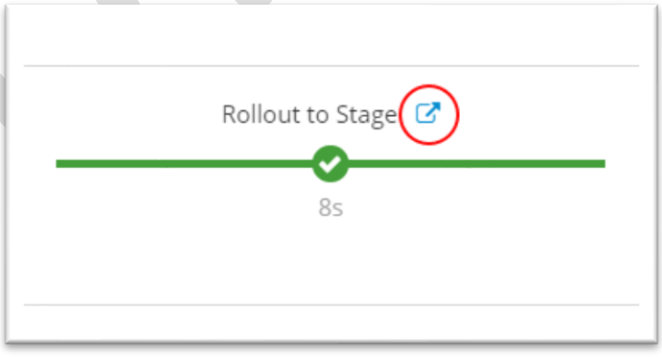
While you wait the pipeline execution, you can click on "View log". That will open a new tab with the Jenkins log for that build. You do not need to do this and if you do not understand Jenkins output, you may want to avoid looking at this information now. If you do view the log, you will likely need to click a log in button or two, and then you will see something like:

```
+refs/heads/*:refs/remotes/origin/*
 > git config remote.origin.url https://github.com/fabric8io/fabric8-pipeline-library.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/fabric8io/fabric8-pipeline-library.git # timeout=10
Fetching without tags
Fetching upstream changes from https://github.com/fabric8io/fabric8-pipeline-library.git
 > git fetch --no-tags --progress https://github.com/fabric8io/fabric8-pipeline-library.git
+refs/heads/*:refs/remotes/origin/*
Checking out Revision 727a9b0b87a27bb484bad95256c32f7f54f69341 (master)
Commit message: "Merge pull request #310 from rawlingsj/master"
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 727a9b0b87a27bb484bad95256c32f7f54f69341
First time build. Skipping changelog.
[Pipeline] echo
Kubernetes Plugin Version 013
[Pipeline] echo
Loaded PipelineConfiguration PipelineConfiguration{jobNameToKindMap={}, ciBranchPatterns=[PR-.*],
cdBranchPatterns=[master], cdGitHostAndOrganisationToBranchPatterns={}, disableITestsCD=false,
disableITestsCI=false, useDockerSocketFlag=false}
[Pipeline] echo
Loaded the useDockerSocket flag false
[Pipeline] podTemplate
[Pipeline] {
[Pipeline] node
Still waiting to schedule task
jenkins-slave-xclh1-h58s0 is offline
```

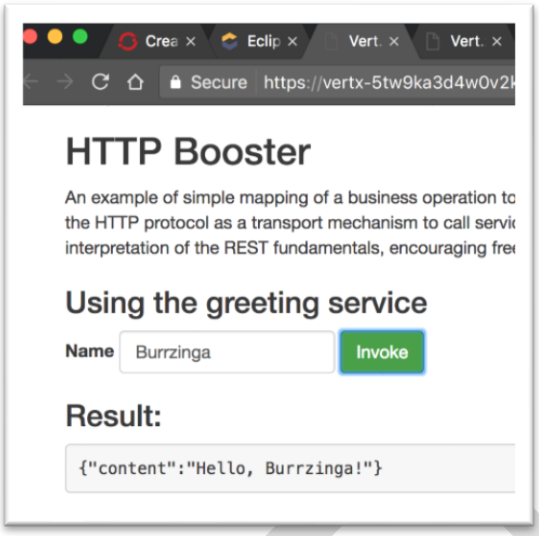
Eventually the Jenkins console output will pause, awaiting your approval. Do not select either the Proceed or Abort link; instead close the Jenkins tab and return to the OpenShift.io Pipelines screen.

Running the application

Your application has been deployed to the Stage environment and is awaiting your approval. Naturally, before you can decide to approve the application you'll want to look at it. In the pipeline where is says "Rollout to Stage", notice that there is an external link icon:

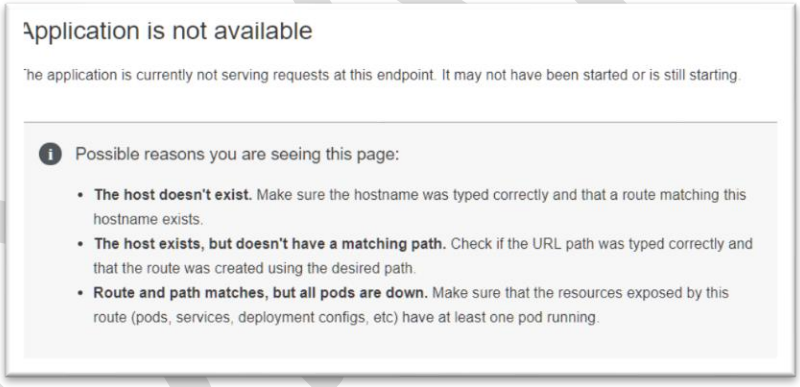


Click this link, and your application will open in a new browser tab. The Vert.x HTTP Booster Quickstart produces a simple API behind a REST endpoint over HTTP with a minimal user interface:



Type your name into the edit box and click Invoke. You will then see the JSON result returning from the API.

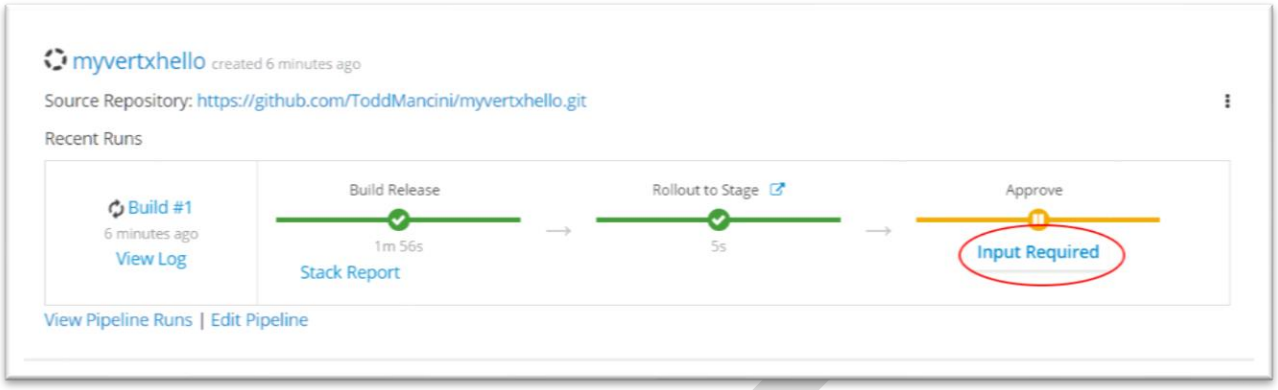
If you do not see the HTTP Booster but rather see an indication that the “application is not available,”



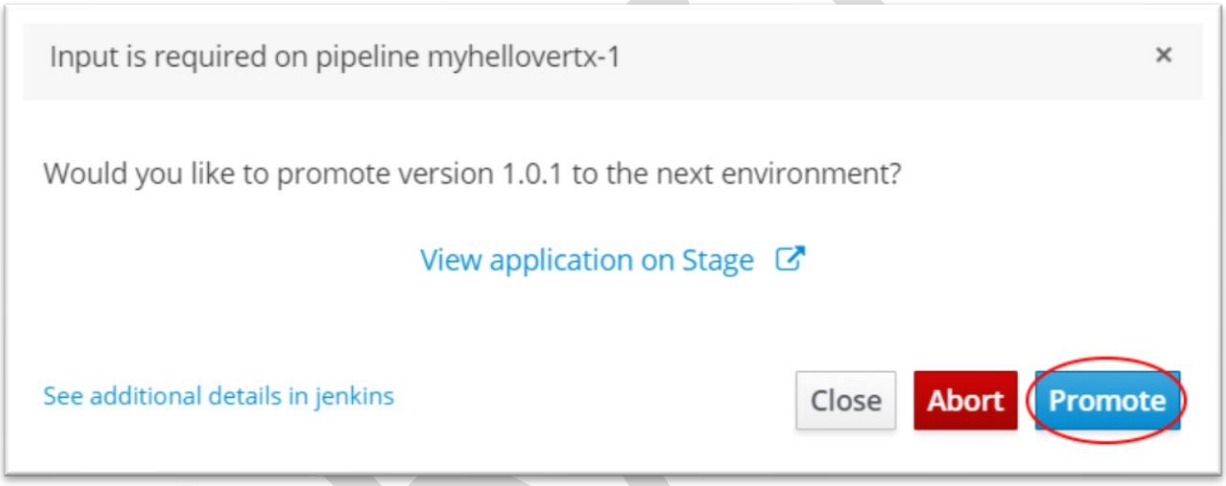
do not panic. When your application is deployed to OpenShift Online, it takes a little time to completely get up and running. Simply keep refreshing the page periodically, and within a short period of time you will see the HTTP Booster.

Approving the application

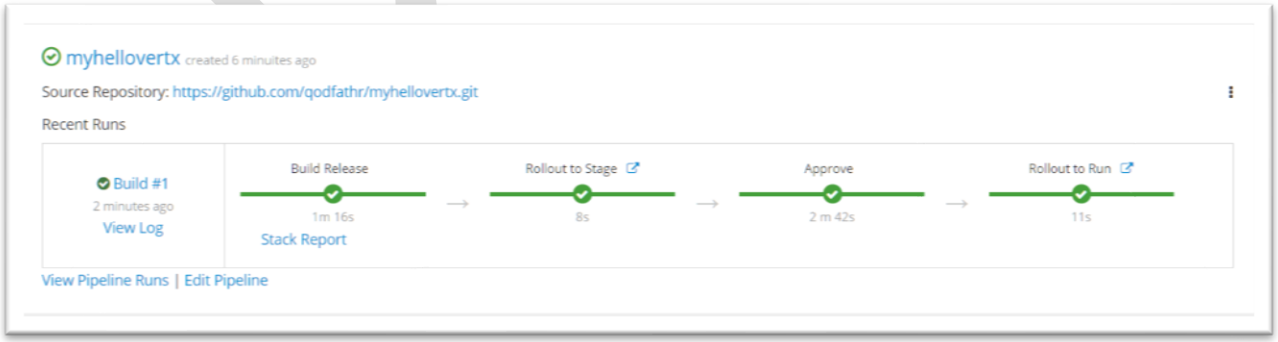
The initial Quickstart-produced application looks good and may now be promoted it to the run environment. Return to the Pipelines screen in OpenShift.io.



Click the “Input Required” button. A promotion dialog will appear.



Click the “Promote” button to promote the application from the Stage environment to the Run environment. Note that the promotion dialog also had a convenient link to see the running application in Stage. Once promoted, the pipeline will continue to execute, concluding with a “Rollout to Run.”



If you like, you may click the external link next to “Rollout to Run” to see the application running in the Run environment. It will behave the same as in Stage as both environments are currently running the exact same version of the application.

Che Workspaces

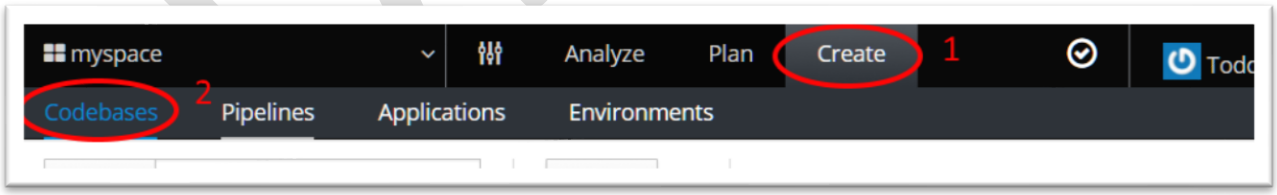
Now that you’ve managed to get the Quickstart-produced “hello world” application running on the internet, it’s time to customize the application. OpenShift.io enables code editing, testing and debugging through hosted instances of Eclipse Che. Eclipse Che is a lightweight, next generation IDE which runs directly in the browser. A key feature of Che are *Che Workspaces* which provide a fully configured runtime environment in which your code can run. In other words, OpenShift.io not only gives you an IDE, it also gives you a containerized development and test environment. Thanks to the power of the Quickstart Application Wizard, that dev and test environment is automatically configured for you with the necessary runtime components. It feels like getting a new and configured personal development machine every time you start working on a new project – because, in a sense, you are.

OpenShift.io Codebases

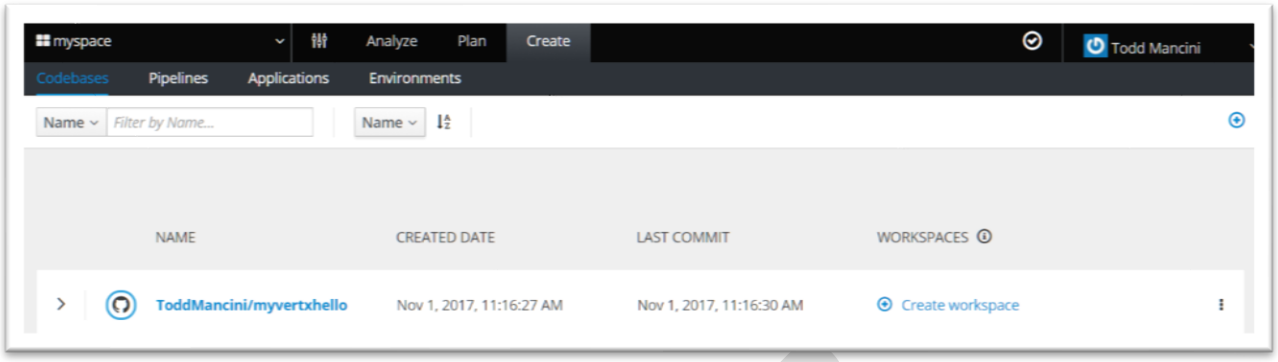
The source code for your application is maintained in a version control system. Today the only version control system usable from OpenShift.io is a git repository hosted on GitHub.com, although this will change in the future to include not only other git hosts but also other version control technologies not based upon git. OpenShift.io also maintains some additional metadata about the version control for a set of code. *OpenShift.io Codebases* are a generic representation of one of these sets of code; today, Codebases are synonymous with a GitHub repo.

Launching Che

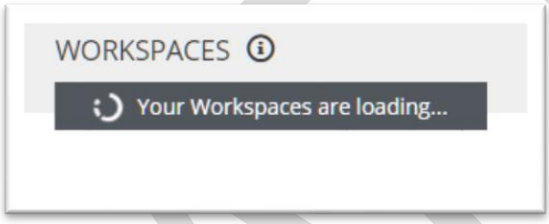
In OpenShift.io, ensure that you are on the top-most Create tab and navigate to the Codebases screen.



Here you will see a list of your Codebases and Workspaces in the current Space. You should have one Codebase and no Workspaces.



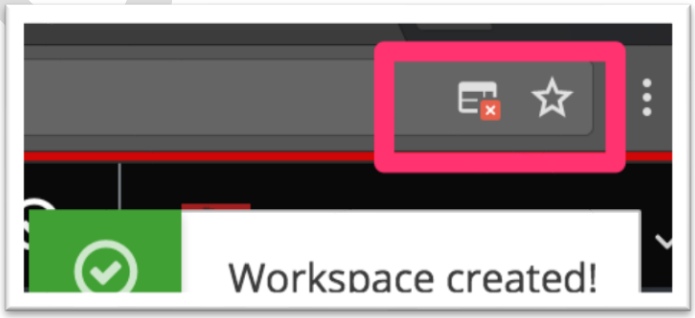
Under the Workspaces column, you may see “Create workspace,” as in the image above. However, if instead you see “your workspaces are loading…,” that indicates that your Che instance has gone idle.



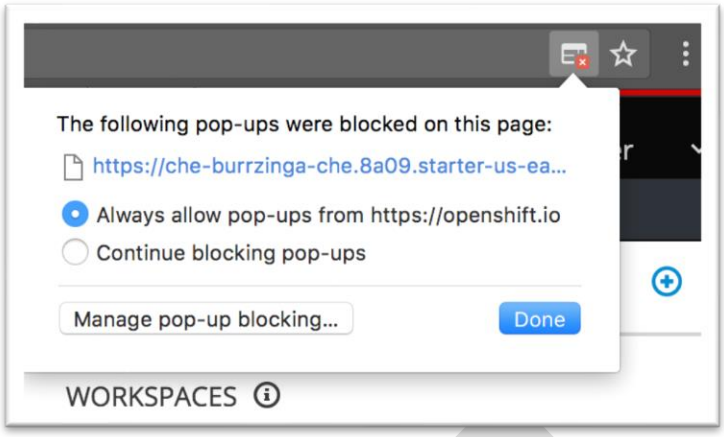
If you Che instance is idle, simply wait for it to start – it will do so automatically, and you’ll then have the “Create workspace” option.

A Codebase may be associated with one or more Workspaces. Think of it this way – the Codebase is the actual code and version history, and the Workspace is a development environment where you can modify, test, debug and run that code. Click “Create workspace” to create a new Workspace associated with the Codebase.

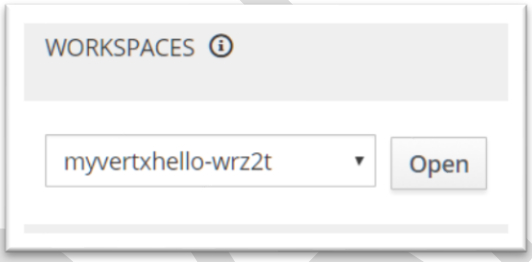
Normally, a new browser tab would now open with Che running in it. However, when you first create a workspace, it is likely that a browser pop-up blocker will prohibit the new tab from opening.



If you see a pop-up blocker icon such as this, click on the blocked pop-up icon.

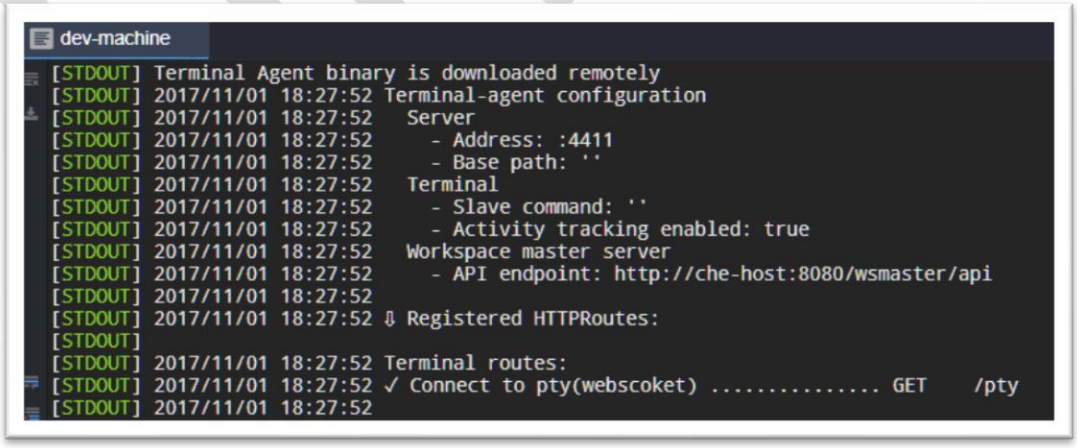


In the dialog which appears, change the option to “Always allow” and click Done. You will now see a Workspace associated with the Codebase.

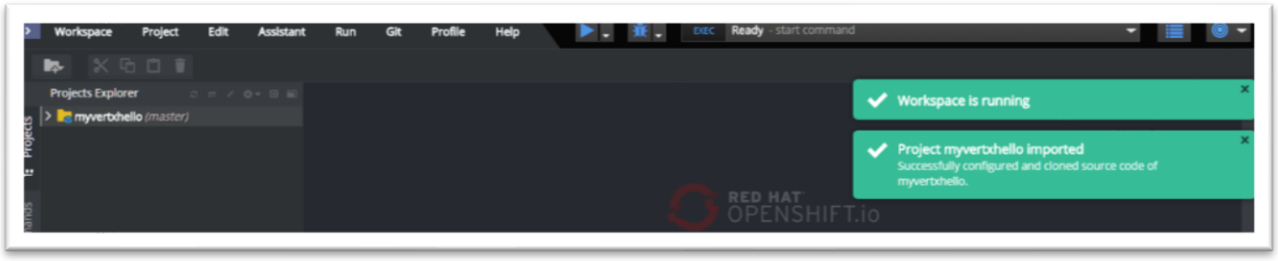


Click the Open button to launch the Che Workspace in a new browser tab.

Starting a workspace is very similar to booting a computer, and, as such, it takes a little bit of time. You will be able to boot logging information during the process.



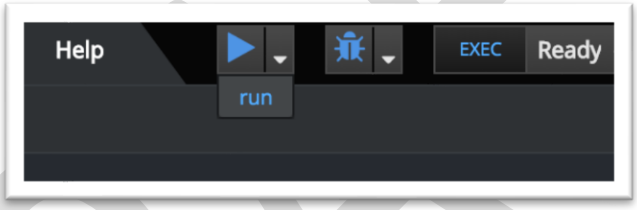
It may take a couple of minutes for the Che Workspace to fully start. A successful start looks like this:



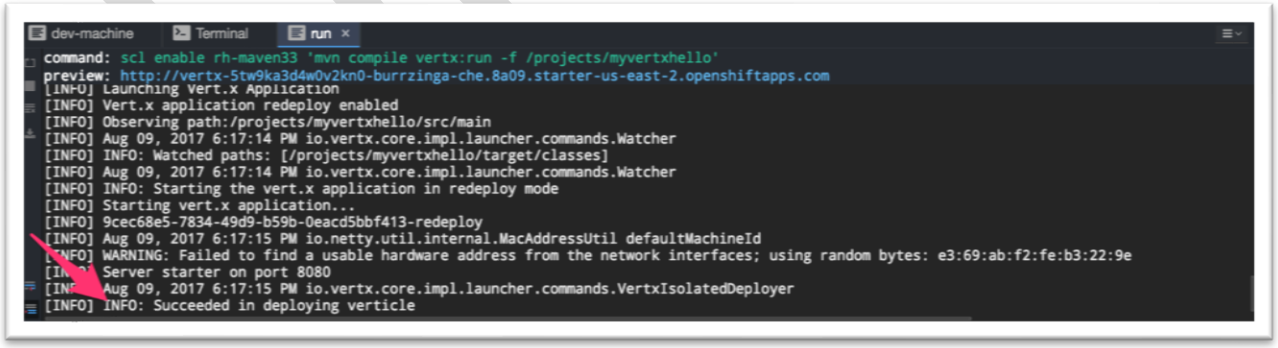
If nothing appears to be happening, refresh the page. If you get an error, please use the chat channel to get assistance.

Running the application in Che

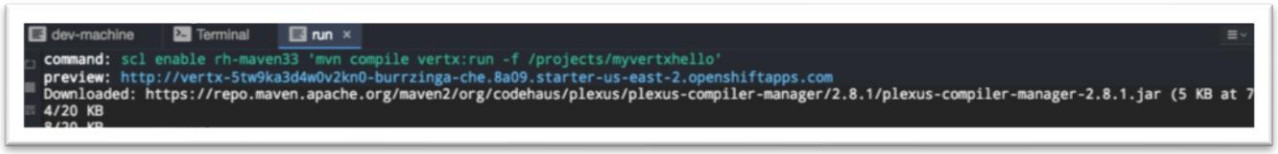
Verify that the application has been properly imported into Che by running it. In the top menu, click on the 'Play' icon  , which will reveal a pull-down menu with one choice – Run. Select Run.



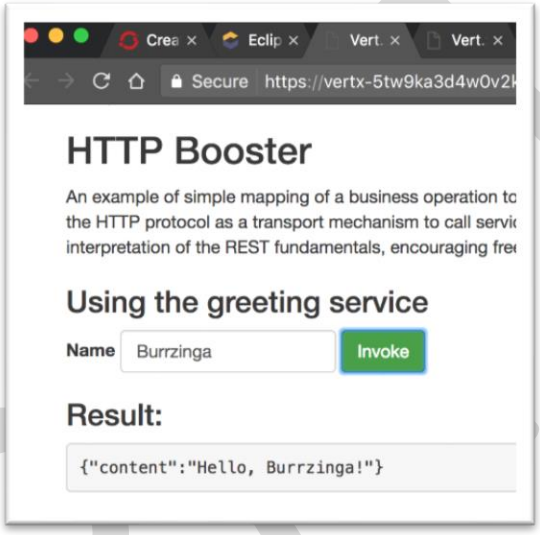
Maven will download any necessary dependencies, compile the application, and start the "verticle," the name used by Vert.x for deployable code. It may be necessary to manually scroll the contents of the 'run' pane at the bottom of screen to see the full output. If everything has worked correctly, the last line of output will read "INFO: Succeeded in deploying verticle."



Just above the output you will see a ‘command’, the command that Che used to run your application, and ‘preview’, a URL to the application which was just deployed.



Click the preview URL to see your application, as running in Che. In will open in a new browser tab.

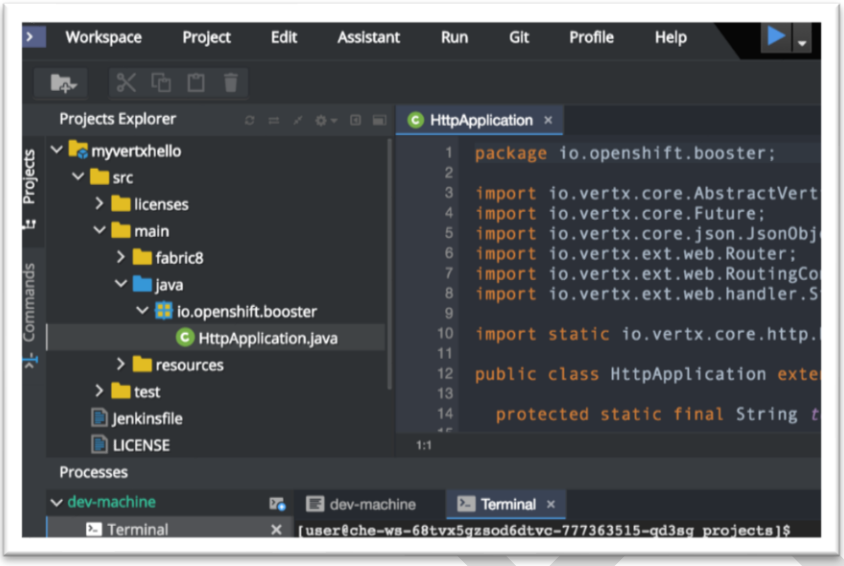


This should look very familiar, as it is the exact same version of the application as the one that the pipeline deployed to the Stage environment and you promoted to the Run environment. Note that this URL is different from the Stage and Run URLs provided by OpenShift Online; this is your private sandbox, hosted within Che. This does not limit you from sharing the URL, though. You can send this URL to someone and interactively debug the application while they run it in their browser.

Modifying the application

You can now customize the application. Return to the Che browser tab. In the Projects Explorer plan on the left-hand side, expand `myvertxhello/src/main/java/io.openshift.booster`. (The last part of that path – the package name – may be different depending upon what you

chose for a Group Id in the Application Wizard.) Double-click `HttpApplication.java` to open it in the editor.

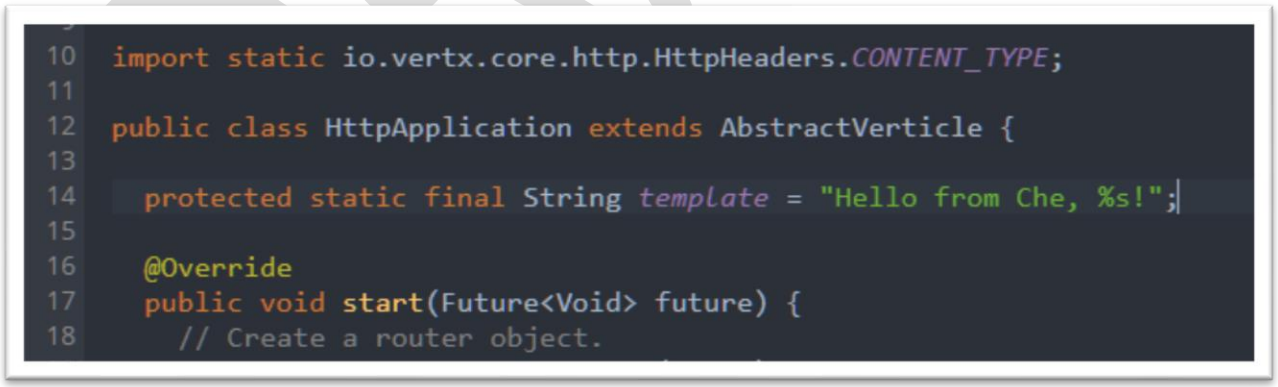


Find the line, on or around line 14 in HttpApplication.java, which reads:

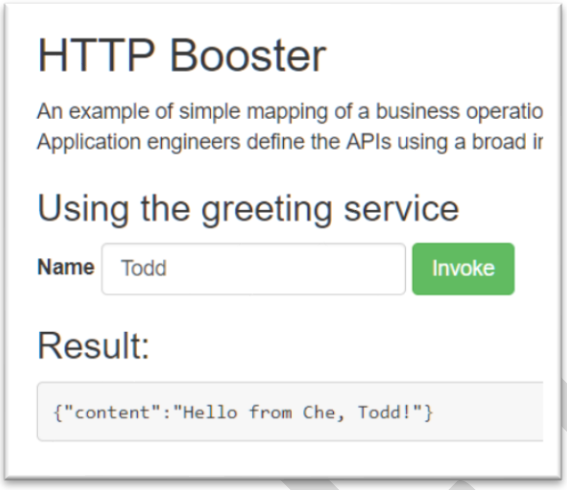
```
protected static final String template = "Hello, %s!";
```

This is the template which defines the message which is returned from the API. Change the message to be `"Hello from Che, %s!"`:

```
protected static final String template = "Hello from Che, %s!";
```



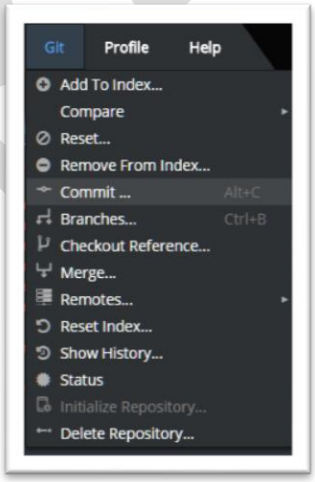
Note that you do not need to save the file – it auto-saves (you can turn off this behavior). Moreover, Maven uses the hot-deploy capabilities of Vert.x to automatically update the running application. Return to the browser tab with the running application, and invoke the service again.



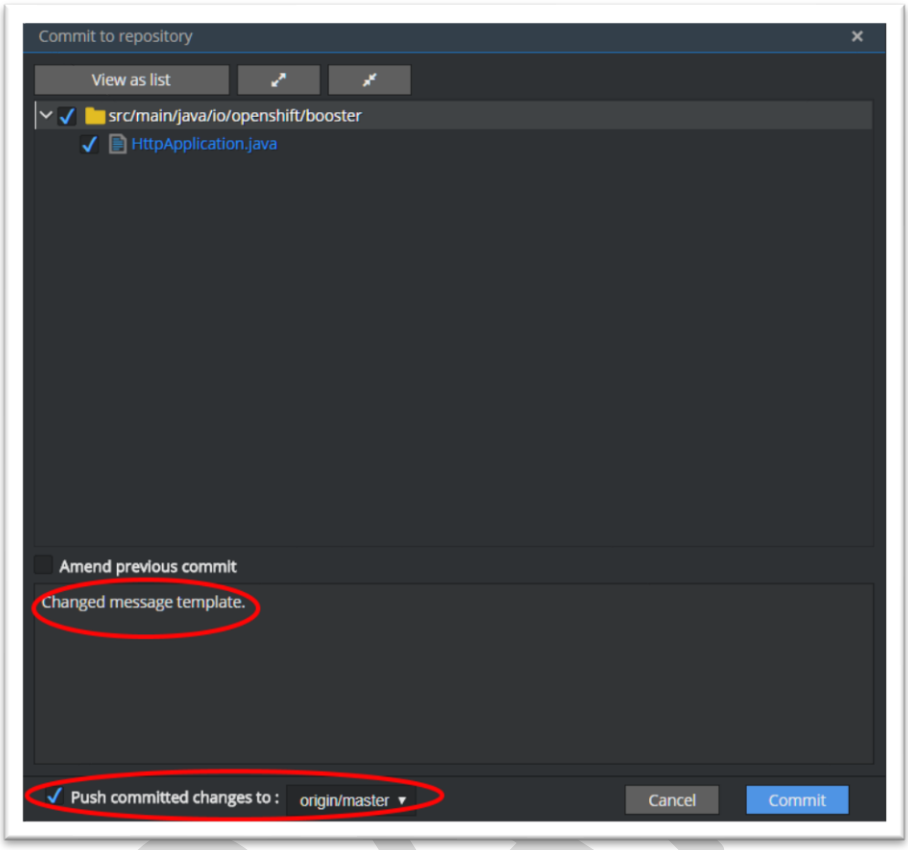
By simply typing in the new message, and without having to save the file or manually recompile and redeploy the application, the code change is live in your sandbox environment.

Committing your code changes to Git

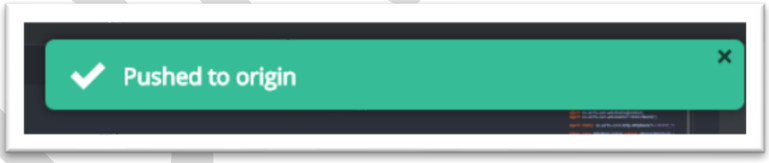
Return to the Che browser tab and select Commit… from the Git menu.



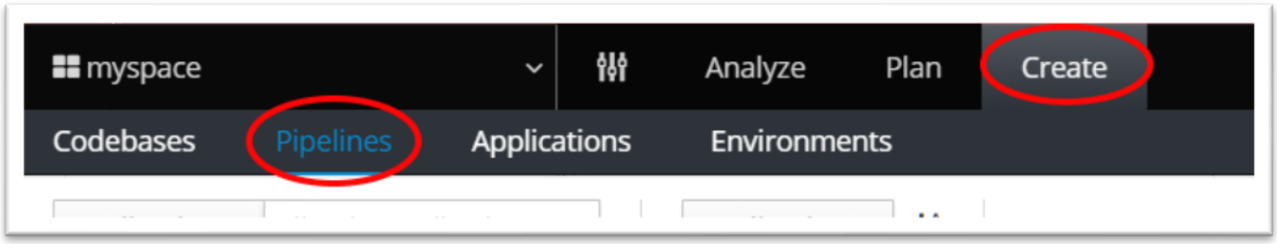
A ‘Commit to repository’ dialog will appear.



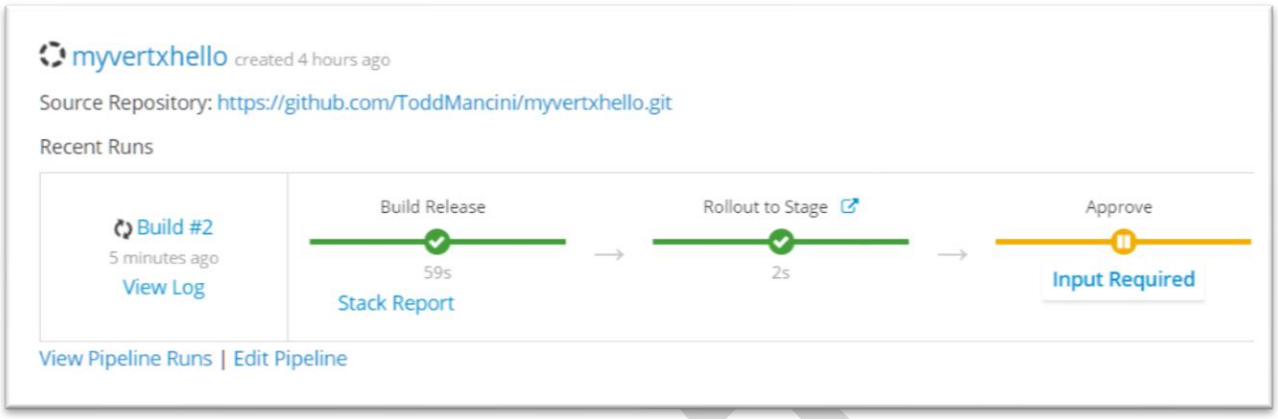
Check the “Push committed changes to” checkbox and put a commit comment such as “Changed message template” in the lower pane. **Do *not* check the “Append previous commit” checkbox**. Click the “Commit” button. In a second or two, a notification should inform you that your change has been pushed to origin.



Since a change has been pushed to the git repository, this will kick off a new build pipeline. Return to the OpenShift.io browser tab, and navigate to the Pipelines screen.



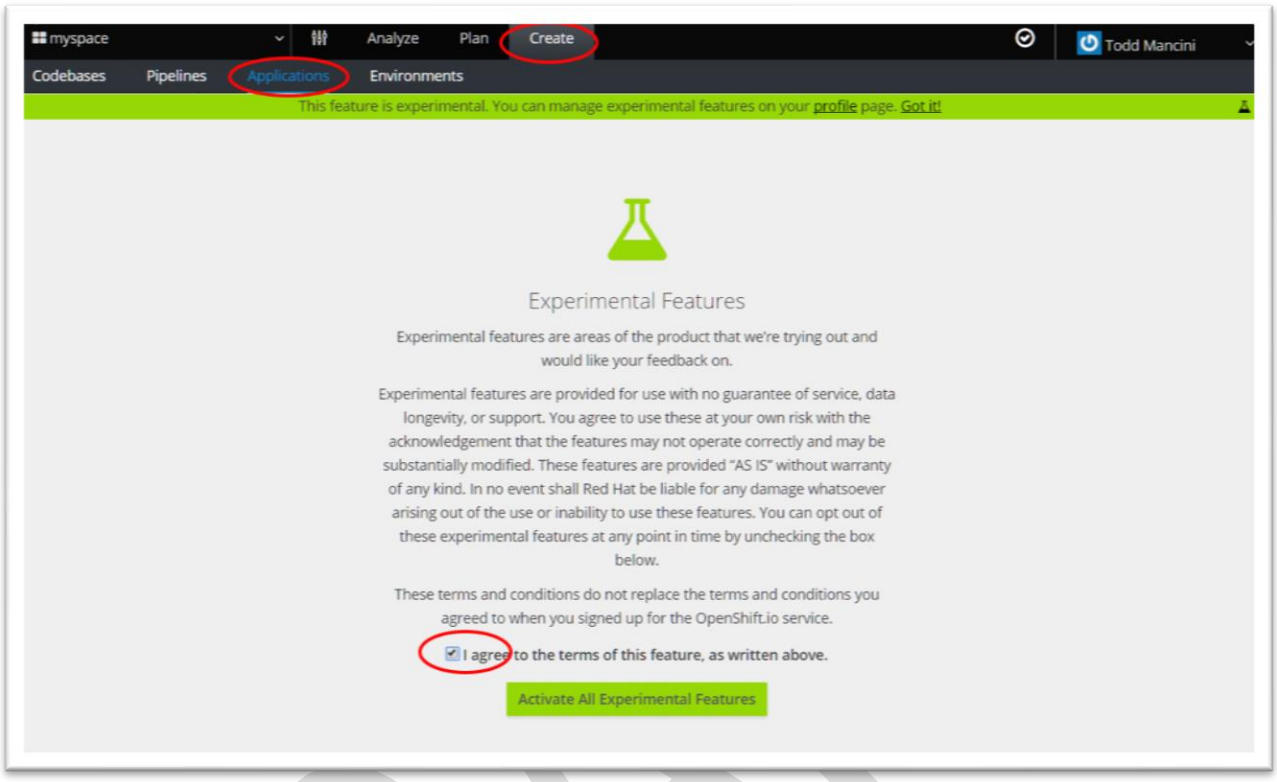
Within a few minutes, you will see Build #2 start. Let it run to the Approve stage.



Comparing Stage to Run

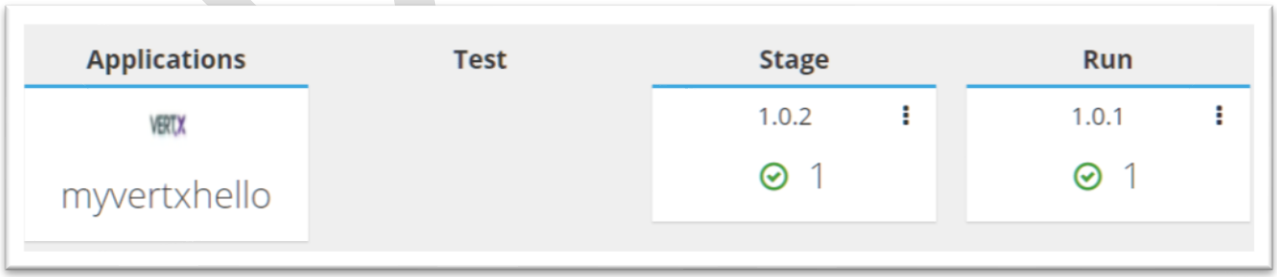
You now have two versions of your application deployed – version 1.0.1 in the Run environment, simulating a production system servicing your users, and version 1.0.2 in the Stage environment, ready for inspection and approval. And while it is true that stakeholders could have also reviewed the application via the preview URL in Che, the Stage environment is meant to closely match the production environment, whereas the Che environment may have both subtle and not so subtle differences in support of its needs as a development environment.

Navigate to the Applications screen. You may see a warning about “Experimental Features.”



During the Early Access Program, OpenShift.io is under extremely active development. Updates are pushed put to production routinely – often dozens of times per day. Many features are approaching “beta” quality. A small number of features are labelled as experiments. These experimental features might not make it into the beta of the service. More likely, these experimental features will have dramatic changes to their appearance before reaching beta.

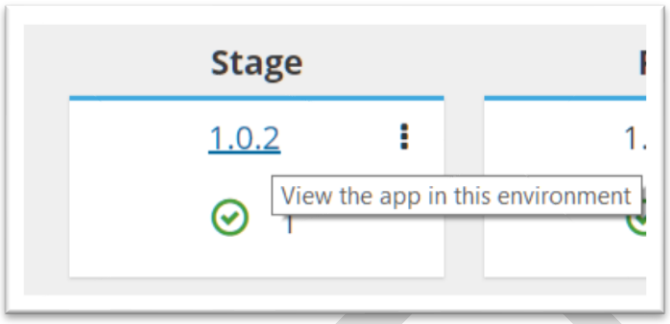
Read the warning and, if you desire, click the “I agree” checkbox and then click the “Activate All Experimental Features” button. The Applications screen will appear.



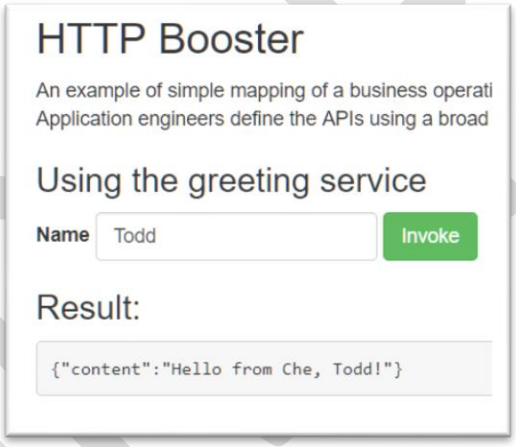
Here you can see that your application “myvertxhello” is deployed to two environments – Stage and Run. Version 1.0.2 in deployed to Stage and version 1.0.1 is deployed to Run. Both are operational (the green checkmark) and each has been scaled to 1 pod. The number of pobs

indicates the number of instances of the application running, in support of scalability and fault tolerance.

The version numbers are, in fact, links to the running applications. For example, if you hover over 1.0.2, you will see this.

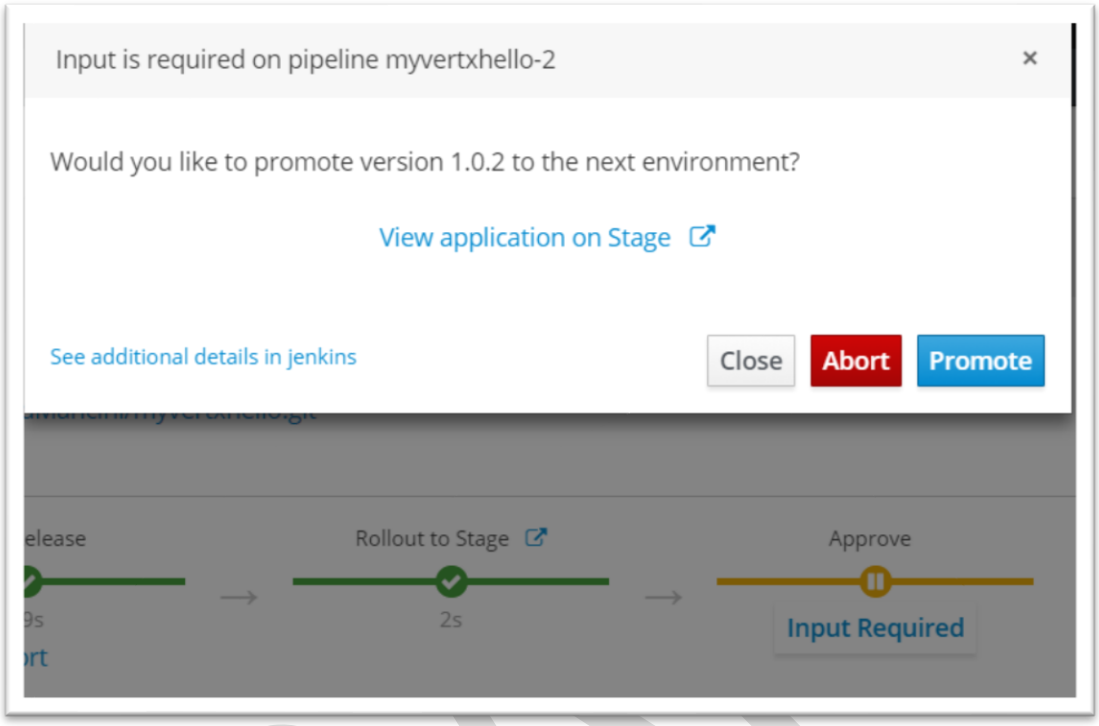


Click the link and you will see the new version of your application, running in Stage.



You can share both URLs to other stakeholders, so that interested parties can compare the new version of the application in Stage to the current version of the application in Run. Once a decision has been made on whether or not to release the new version of the application, return

to the Pipelines screen, click the “Input Required” button, and either Promote the version from Stage to Run, or Abort the pipeline, leaving version 1.0.1 in Run.



Congratulations! You have completed the OpenShift.io Hello World exercise!

Analyze the application

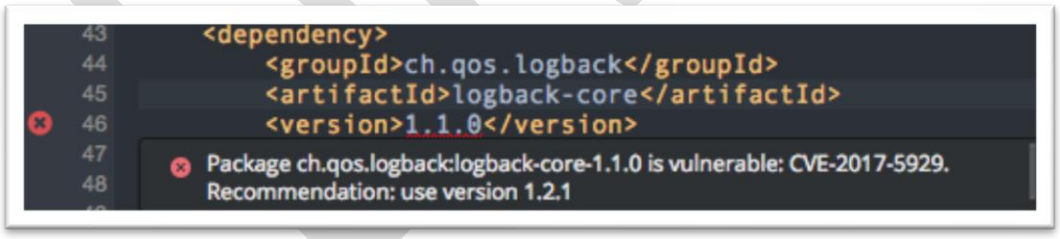
CHE dependency analyzer

Return to CHE tab and open the pom.xml file.

Now go to the line 43 and insert the following XML snippet.

```
<dependency>
    <groupId>ch.qos.logback</groupId>
    <artifactId>logback-core</artifactId>
    <version>1.1.0</version>
</dependency>
```

Note that the version will become red and a red sign will appear. When you place your mouse over this red sign you will see the following message.

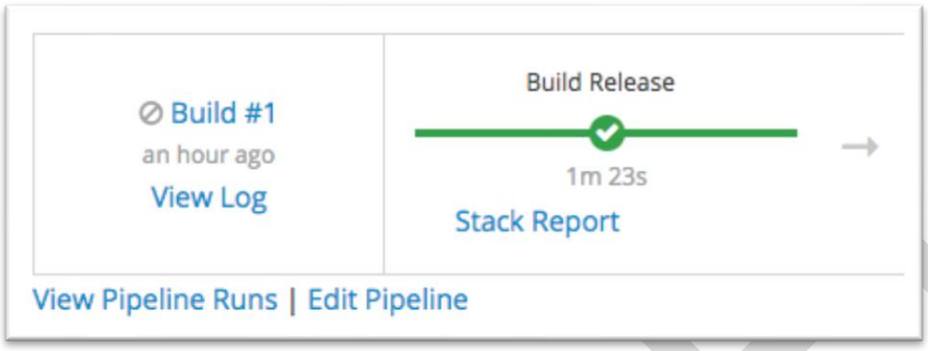


This indicates that this dependency has a CVE (Common Vulnerabilities and Exposures) and the version 1.2.1 should be used instead.

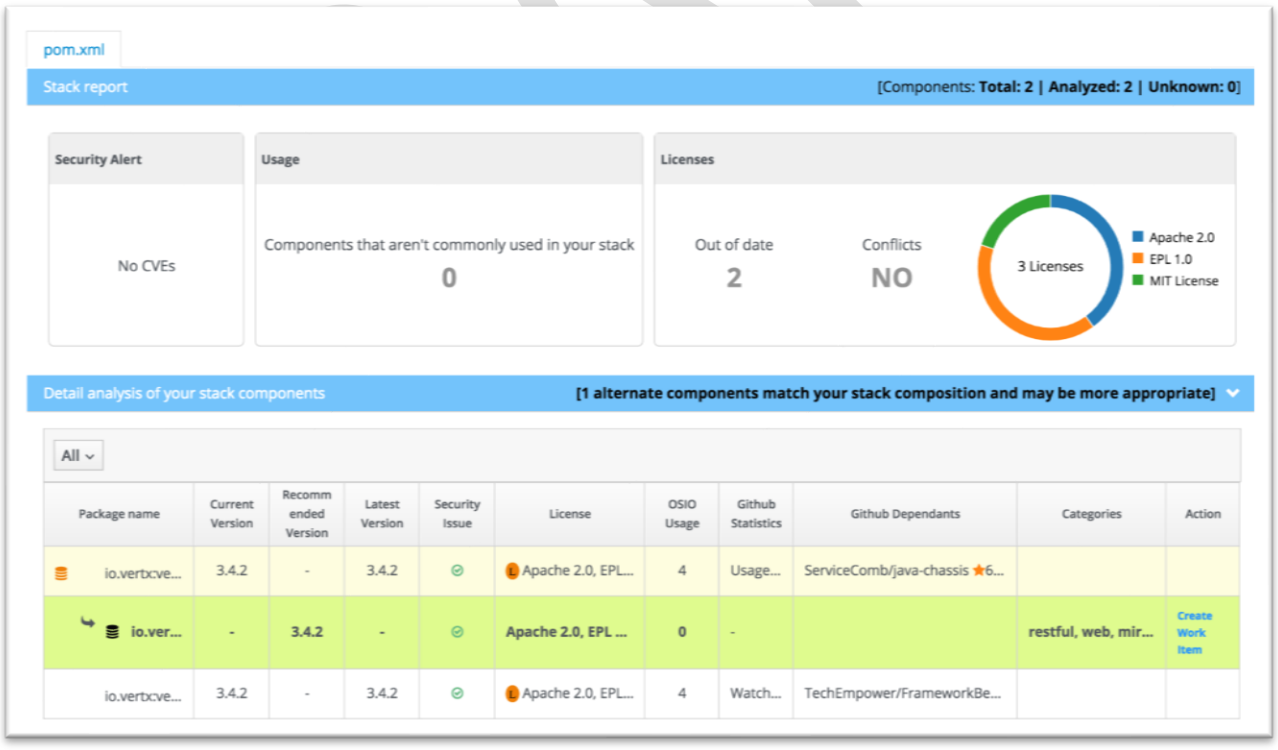
Replace the version 1.1.0 by the version 1.2.1 and note that the red sign will disappear.

Stack reports

Go to the Openshift.io dashboard. In the pipelines menu (, you may see a Stack Report Recommendations section.



You can click on "Stack report" link. The Recommendations can appear AFTER the execution of your first successful build.

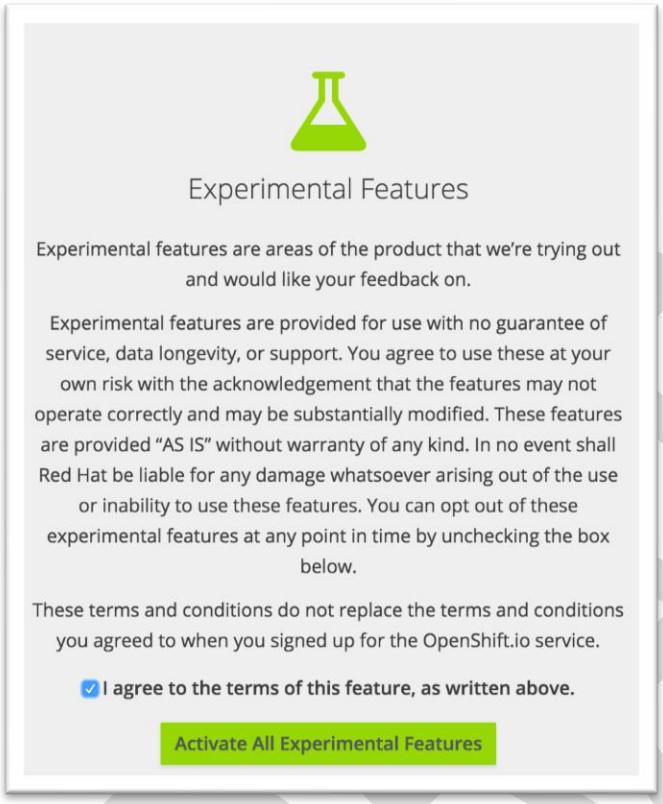


Note that it also provides some suggestions for improvements for your application.

Now, let’s create a work item to track our work.

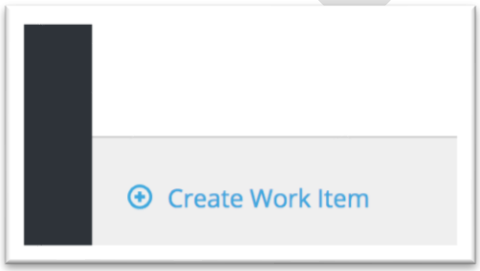
Accessing the Planner - In the Plan Tab

From the Plan tab, Activate All Experimental Features to gain access to the work item Planner:



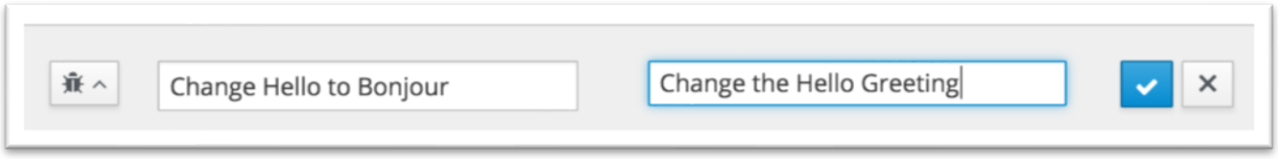
Create Work Item

In the Planner, tasks are tracked as work items. We will change the deployed quickstart project to display a different string. Create a new work item (on the bottom of the page) now to track this change:

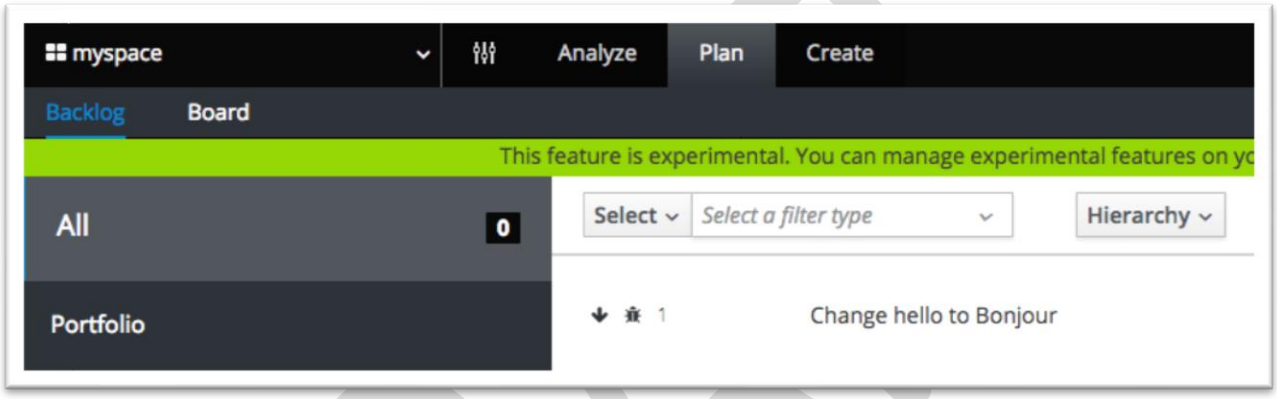


Click on Create Work Item

“Change Hello to Bonjour” and a description of “Change the Hello Greeting” and hit enter or click the checkmark.

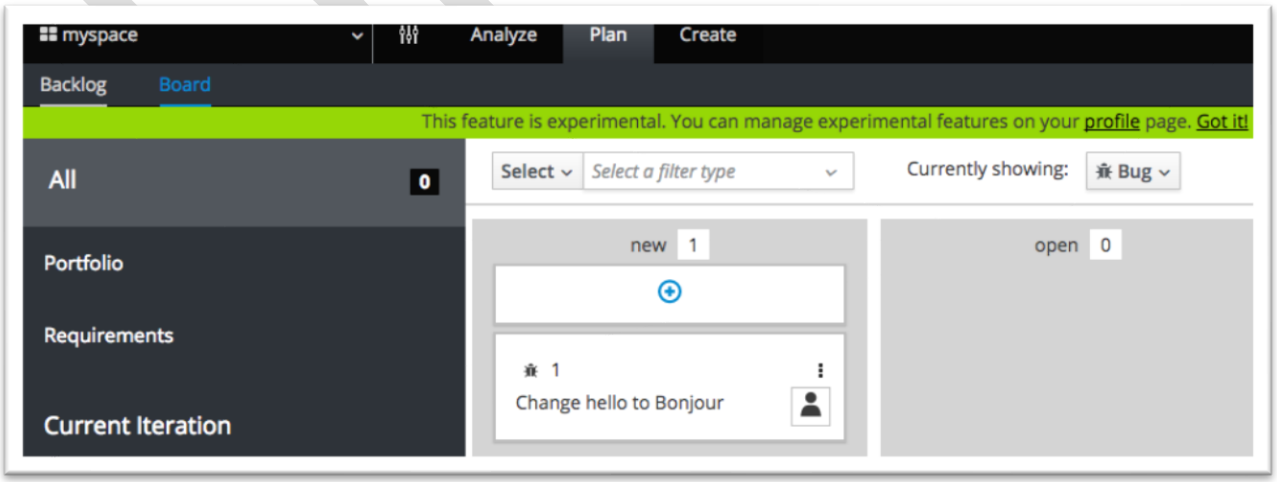


View the new work item in the Planner’s Board view



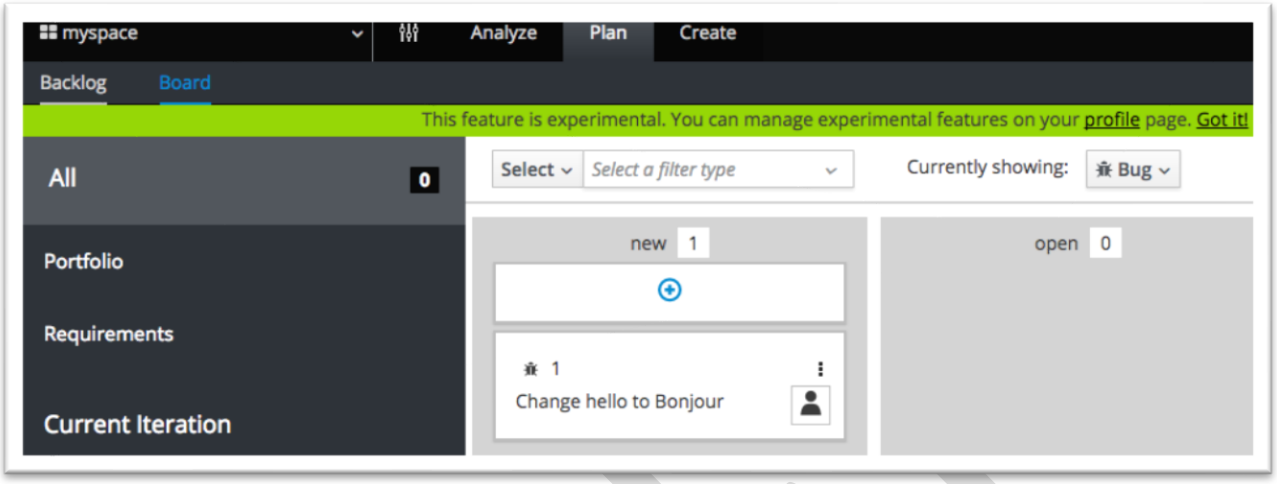
Click on **Board**:

Go to the board view tab (under Plan):

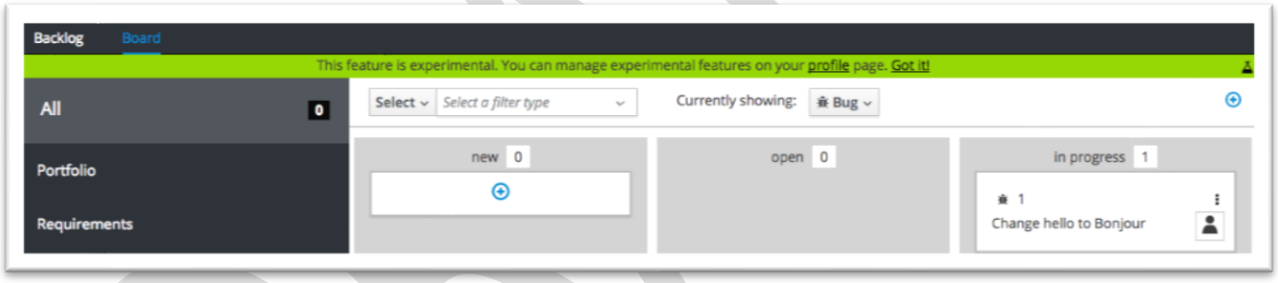


Drag to In Progress

To set the work item's status from "new" to "in progress," drag the work item from "new":



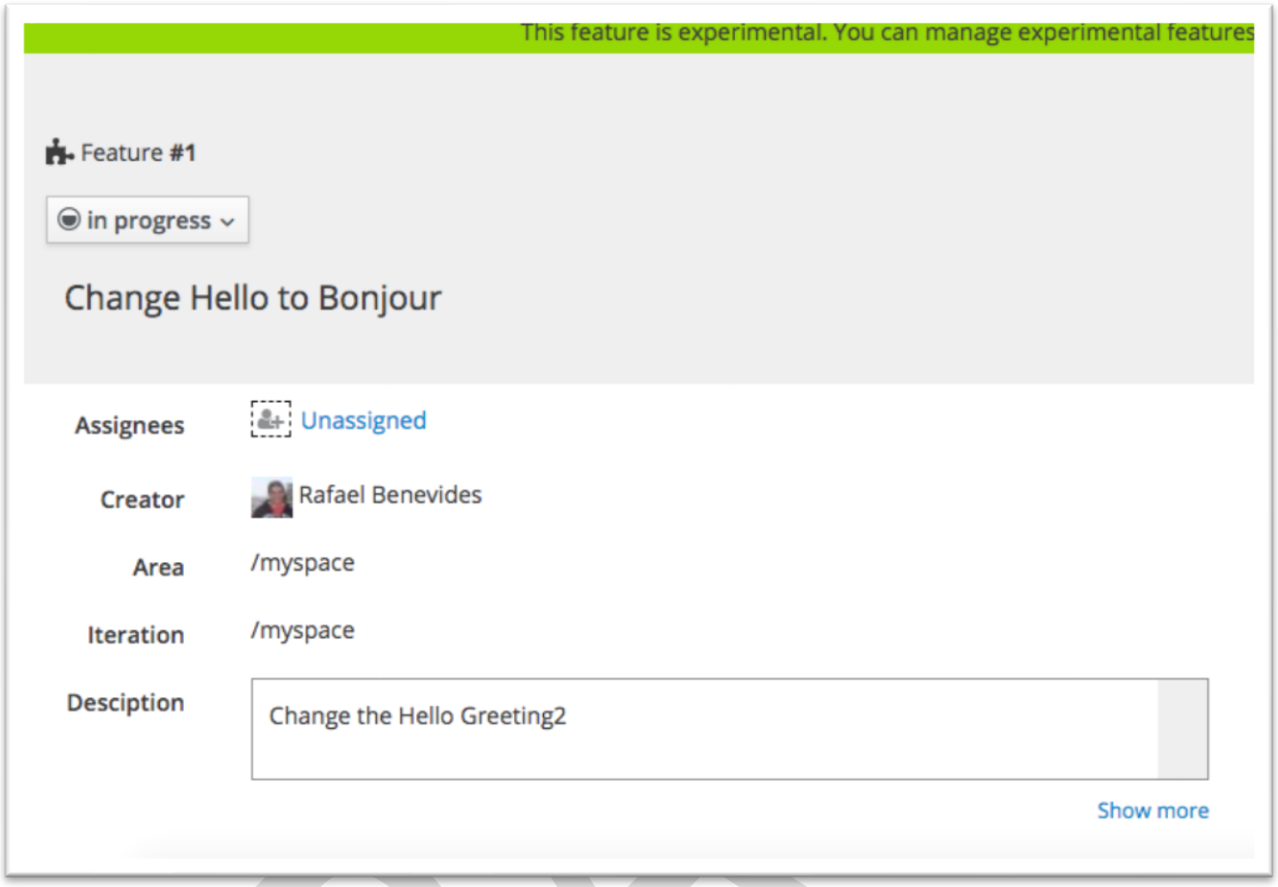
To "in progress":



Assign to Self

A work item represents work for a person. Assign this work item to yourself:

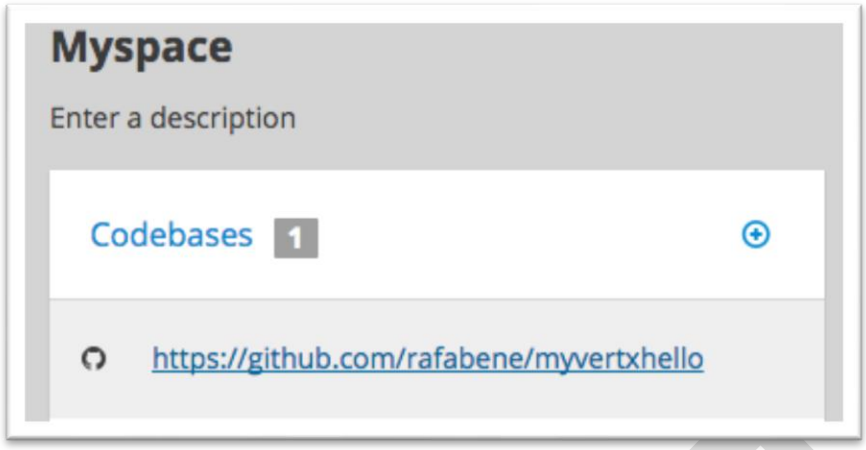
Click on the work item card to open up a detailed view. From there you can assign it to yourself on the "Assignees" line.

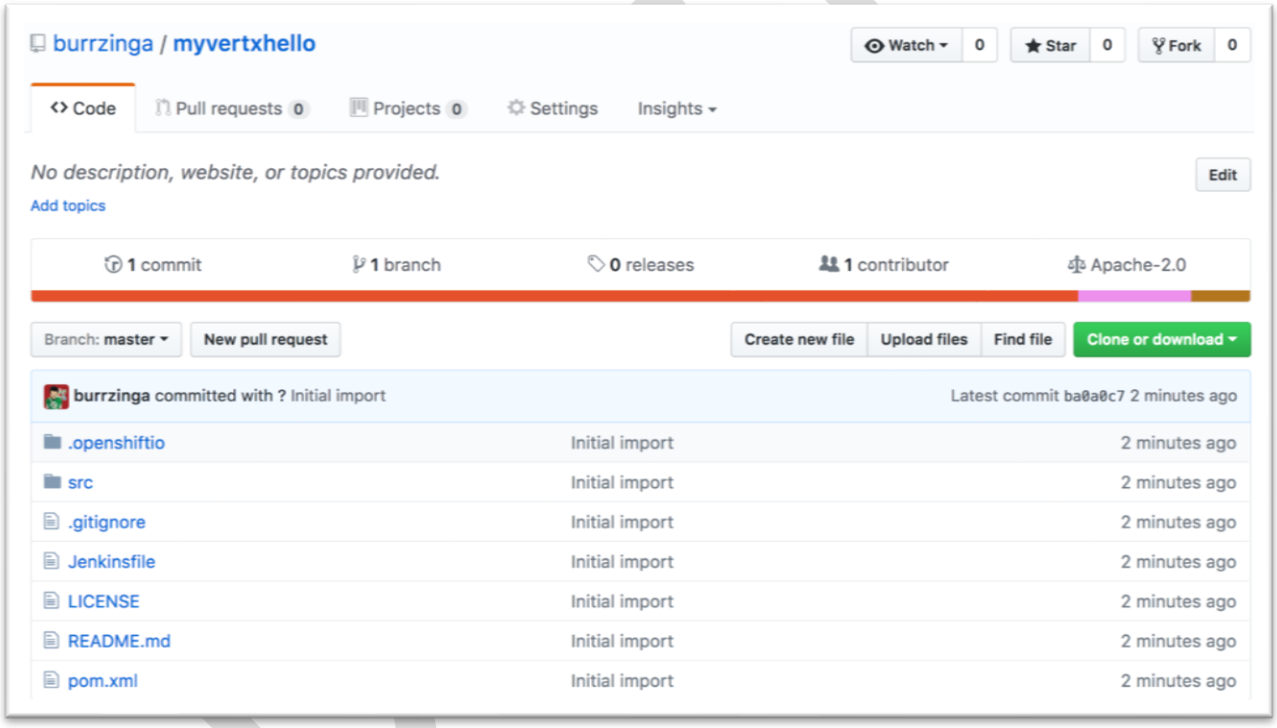


The assignment results in an automatic email notification. Go check your email as a notification should be received based on this work item addition/change. Note: depending on your mail server/account, there may be a delay of a few minutes before you receive the notification email.

Viewing the project files in Github

Visit github to see the generated project files by clicking on the codebase link (Analyze tab)



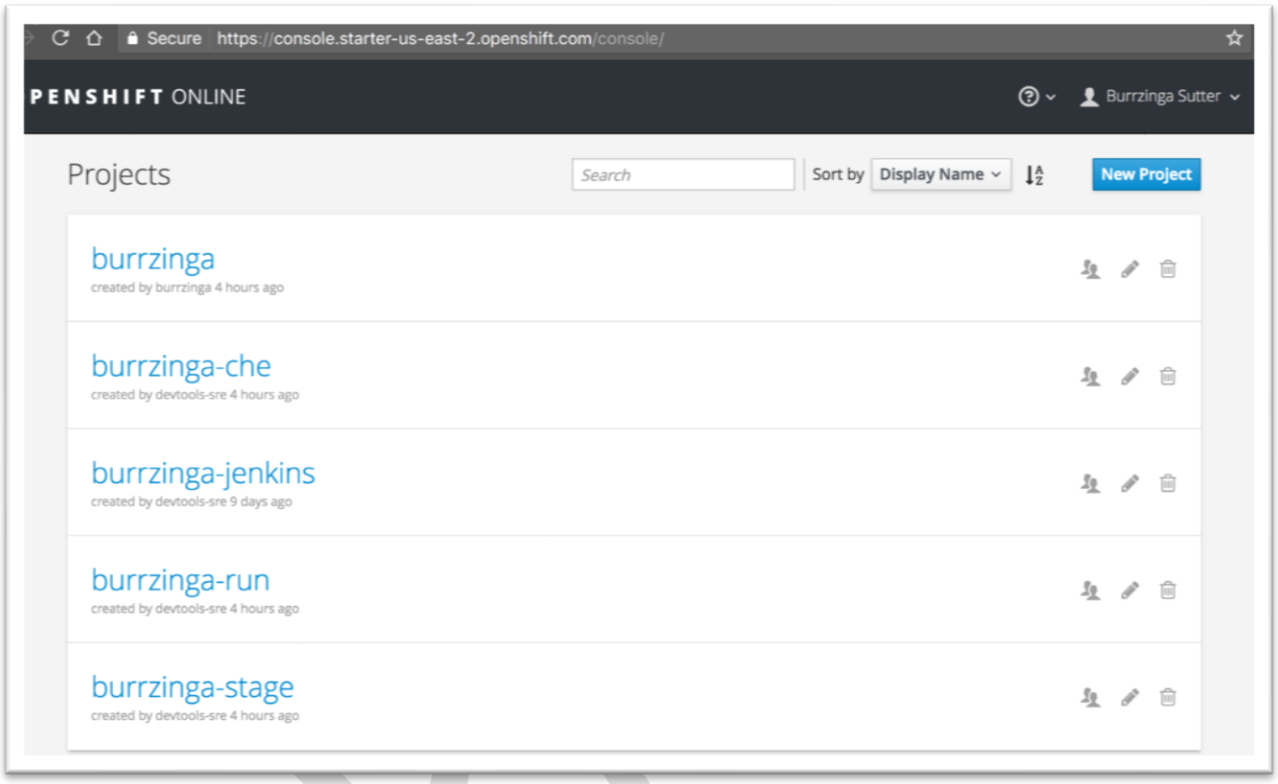


There's a file called Jenkinsfile that implements the staging and rollout as defined in the pipeline:

```
44
45        stage('Build Release'){
46          mavenCanaryRelease {
47            version = canaryVersion
48          }
49        }
50
51        stage('Integration Testing'){
52          mavenIntegrationTest {
53            environment = 'Test'
54            failIfNoTests = localFailIfNoTests
55            itestPattern = localItestPattern
56          }
57        }
58
59        stage('Rollout to Stage'){
60          kubernetesApply(environment: envStage)
61          //stash deployments
62          stashName = label
63          stash includes: '**/*.yml', name: stashName
64        }
65      }
66    }
67  }
68
69  if (deploy){
70      node {
71          stage('Approve'){
72            approve {
73              room = null
74              version = canaryVersion
75              console = fabric8Console
76              environment = 'Stage'
77            }
78          }
79
80          stage('Rollout to Run'){
81            unstash stashName
82            kubernetesApply(environment: envProd)
83          }
84      }
85  }
```

Viewing the pipeline's projects in OpenShift Online

Visit https://console.starter-us-east-2.openshift.com/ to view the OpenShift projects that support the pipeline:

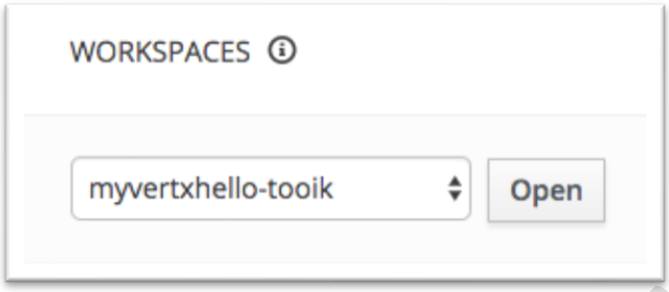


The following (5) projects/namespaces are created in OpenShift Online:

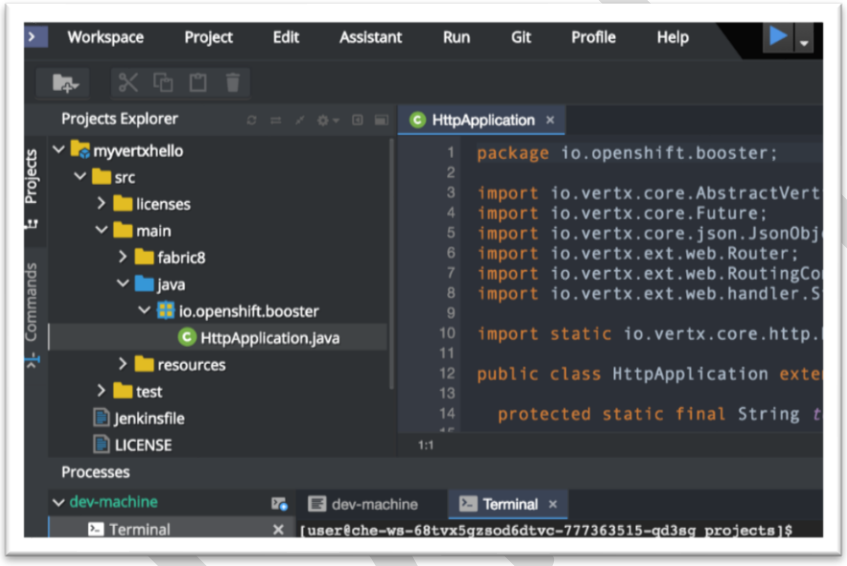
- Username project: where your pipelines run
- username-che project: This is your Che Host and Workspaces
- username-jenkins project: This is your Jenkins Master for your Jenkins Slaves - access to the Jenkins console is available and you would like to watch the Monitoring Tab
- username-stage project: is yours to work with, pods will show up as pipelines run. You will often wish to come here and shut down pods that are unnecessary for your task at hand.
- username-run project: identical to -stage, your place to experiment with pods in OpenShift

Advanced Che

Return to the Create -> Codebases and select the existing workspace and click in the "Open" button. That will open the CHE workspace previously created.



Open Source file: HttpApplication.java



Explore Code Assistant

In the imports section add a new import by typing:

```
import io.
```

Now hit ctrl+space - you will see an autocomplete menu. Select any of the options. You will also see a red “x” appear in the gutter warning you that the import is unused. Delete it to clear the error.

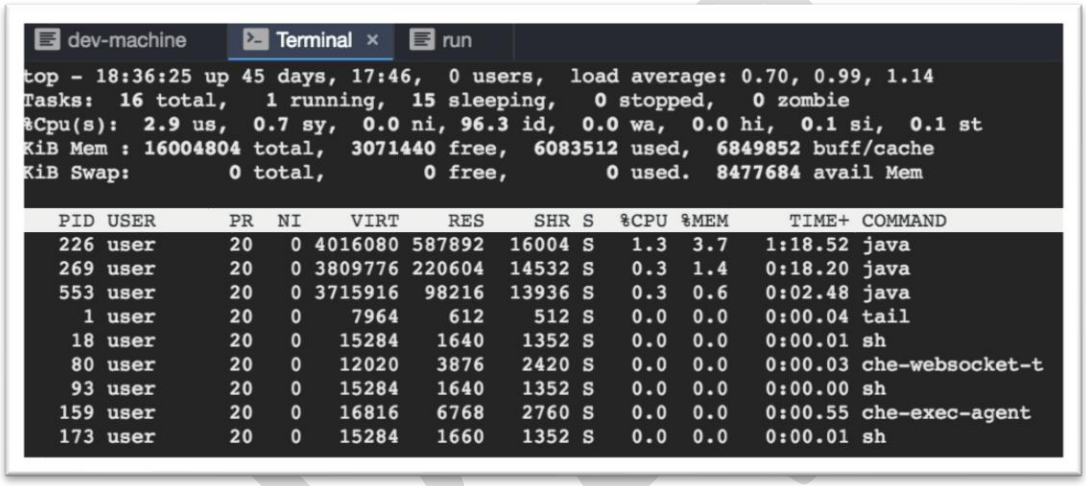
If you want to explore other IDE features try the “Assistant” menu.

Explore the Terminal

Open the Terminal (this is your private linux container, your workspace)

Execute these Unix commands:

```
cat /etc/os-release

top
```

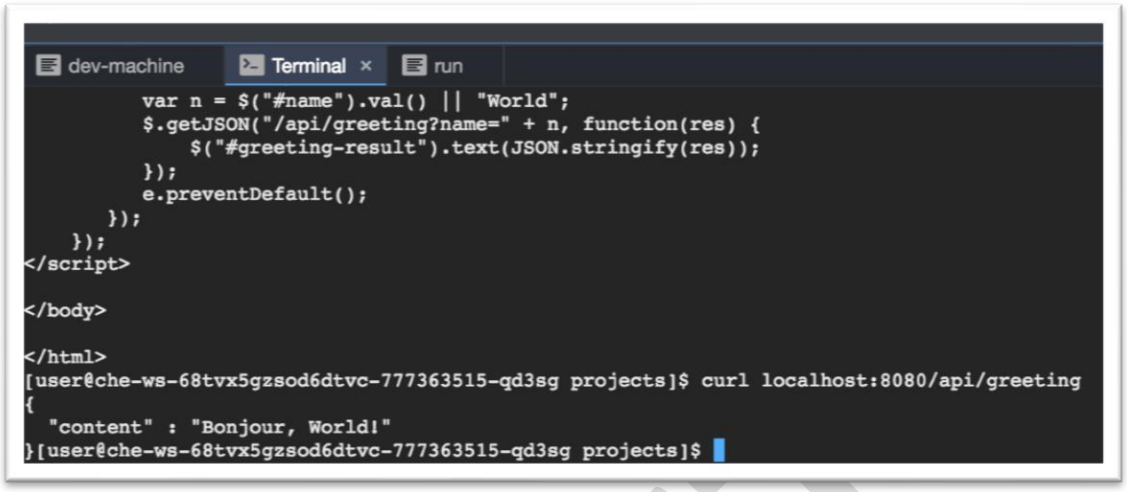


Yes - It is Linux!

Execute these Unix commands to see the code change that you just made:

```
$ ctrl-c (to stop top)

$ curl localhost:8080

$ curl localhost:8080/api/greeting
```

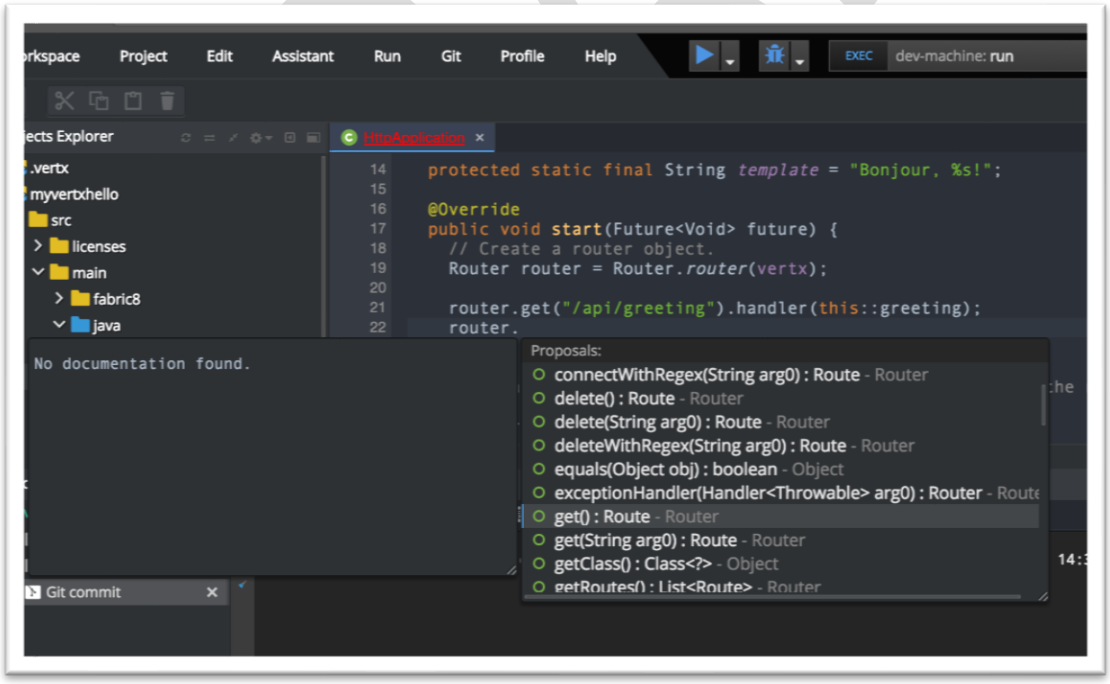
NOTE: Remember that you left your application running inside CHE



Che Debugging

At this point, let's perform debugging tasks for the quickstart.

Insert a new line between the line 21 and 22. Type "router." and hit ctrl-space to see the context-aware assistance, we want the Vert.x Router's get method (maps to the HTTP verb GET):



Add a "goodbye" endpoint:

```
router.get("/api/goodbye").handler(this::goodbye);
```

And its method:

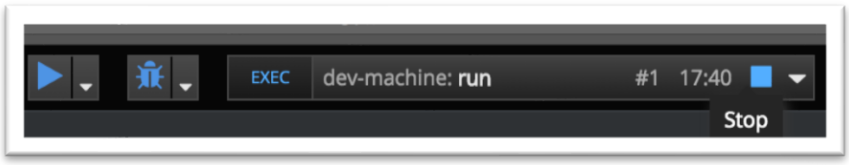
```
private void goodbye(RoutingContext rc) {
    String name = rc.request().getParam("name");
    if (name == null) {
      name = "World";
    }
    JsonObject response = new JsonObject()
        .put("content", "Goodbye " + name);

    rc.response()
        .putHeader(CONTENT_TYPE, "application/json; charset=utf-8")
        .end(response.encodePrettily());
  }
```

Save, let the change hot redeploy and then execute this Linux command in the terminal to verify that the “goodbye” API endpoint has been added:

Execute the following command in the Terminal and check its output

```
$ curl localhost:8080/api/goodbye


{
  "content" : "Goodbye World"
}
```

And, stop the current run by clicking in the "stop" button:



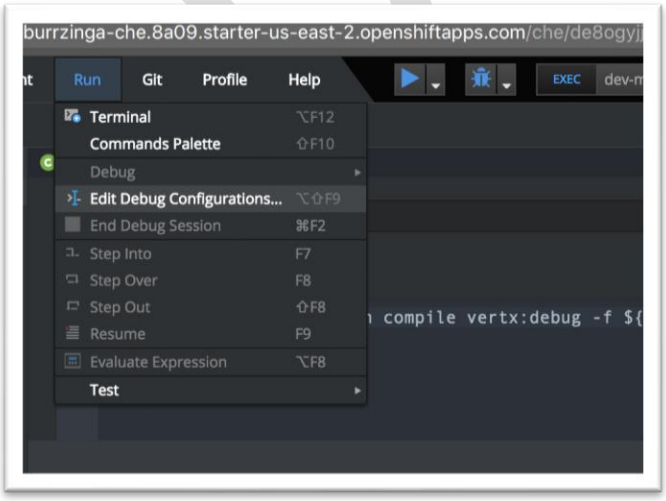
Setup the Debugger

Click "Debug" from the Che menu bar.

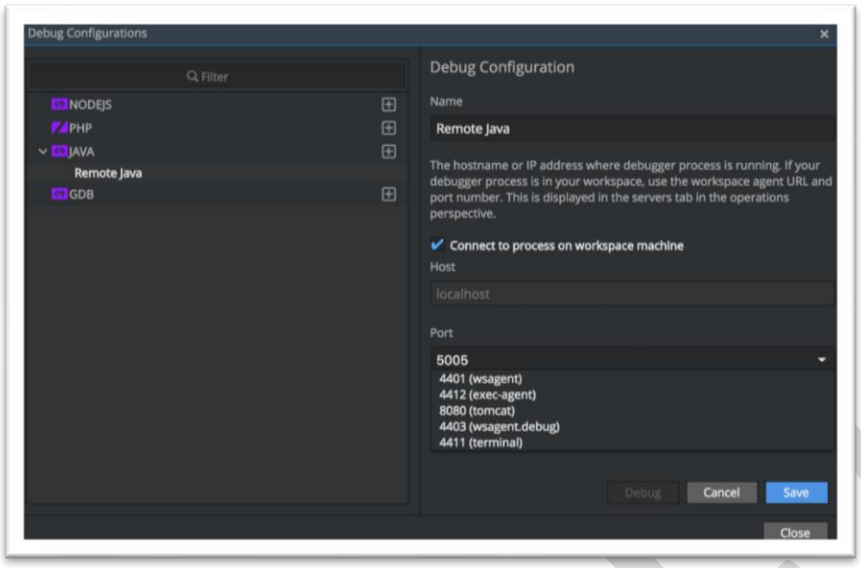
Some key lines to look for in the output are:

```
[INFO] The application will wait for a debugger to attach on debugPort 5005

[INFO] Launching Vert.x Application

[INFO] Listening for transport dt_socket at address: 5005
```

Next, from the top menu, select Run and Edit Debug Configurations:

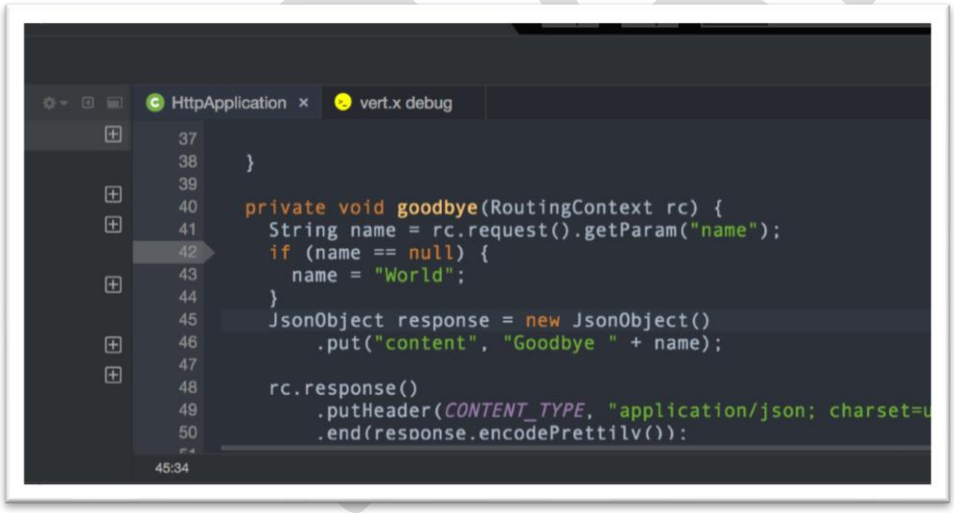


Hit the + to add a “Remote Java” and make its port 5005

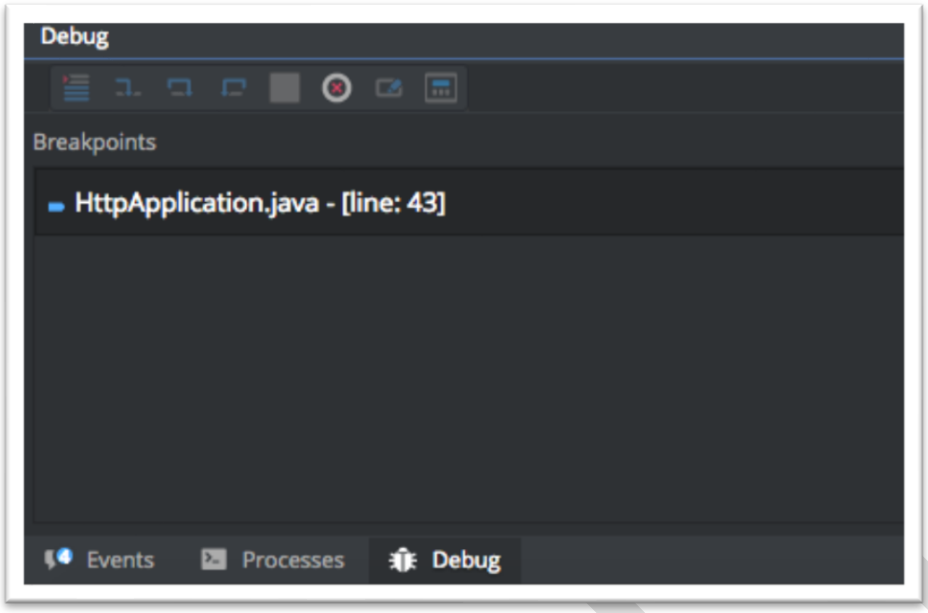


Click Save and then Close to save the new Debug Configuration.

Next, create a breakpoint on your new method’s “if (name == null)” line by clicking on its line number:

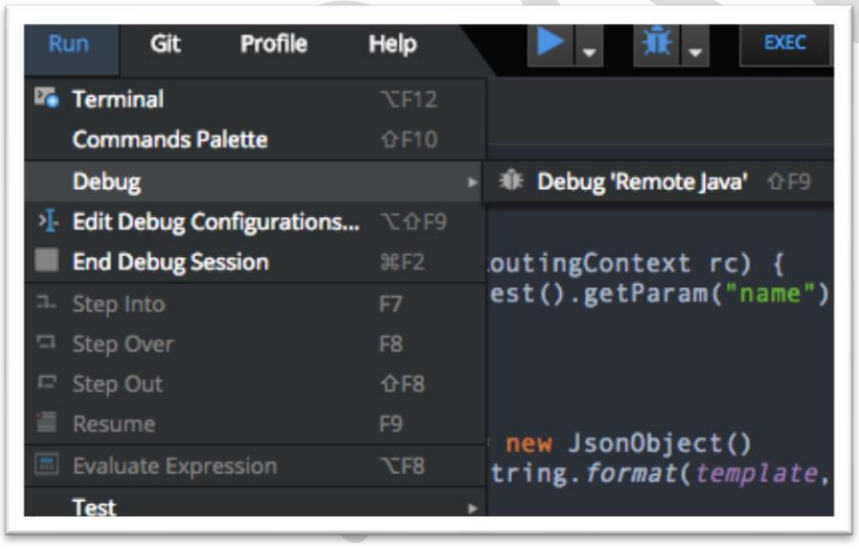


A successful breakpoint addition must be confirmed by accessing the Debug > Breakpoints panel at the bottom of the Che workspace.

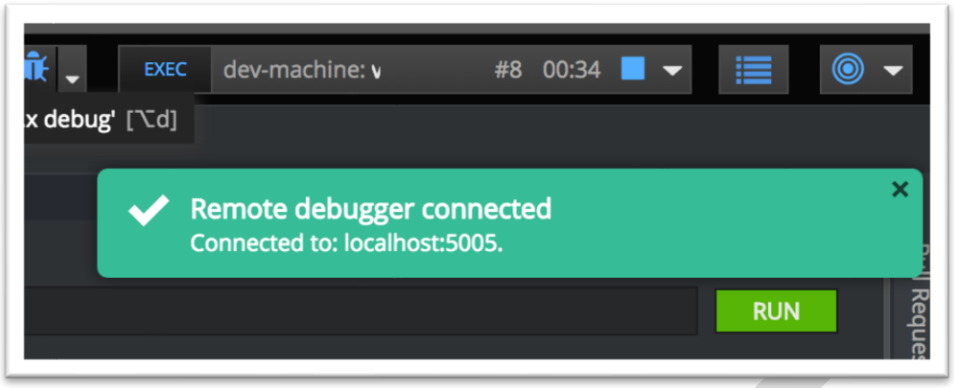


Run the Debug command

Now, go the Run menu and select Debug -> 'Debug Remote Java':



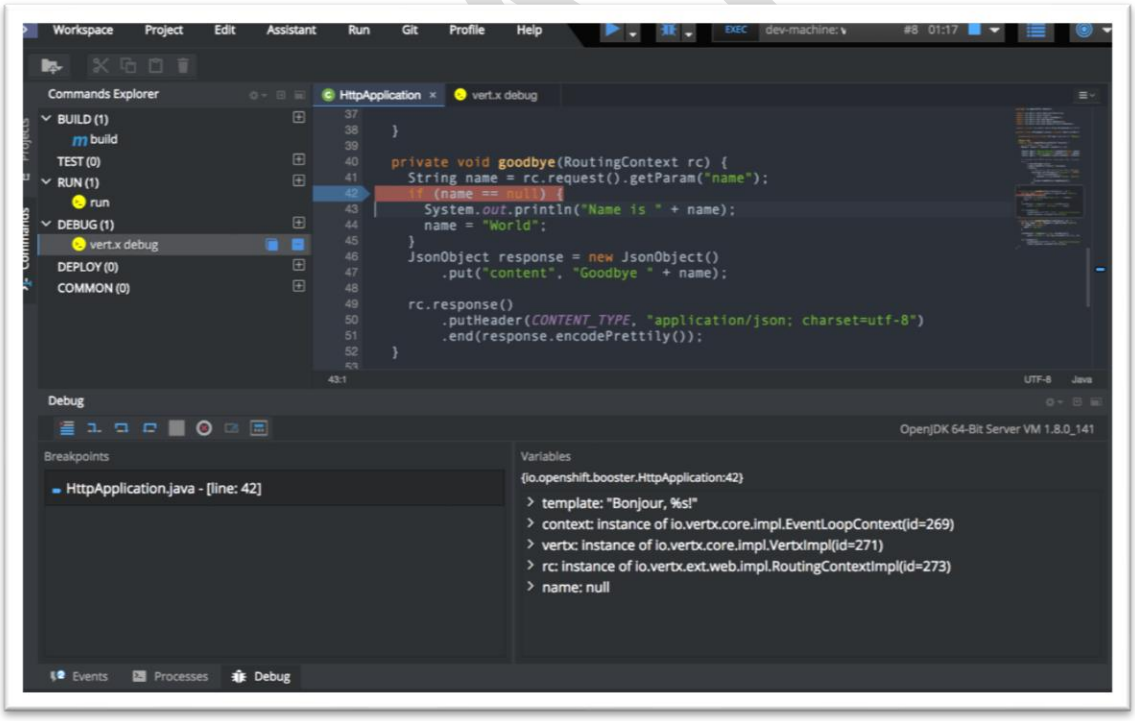
You will see a notification when the remote debugger is connected:



If it fails to connect, it is best to restart your Chrome Incognito session/browser window.

Execute the following command in the terminal and you will be debugging.

```
$ curl localhost:8080/api/goodbye
```



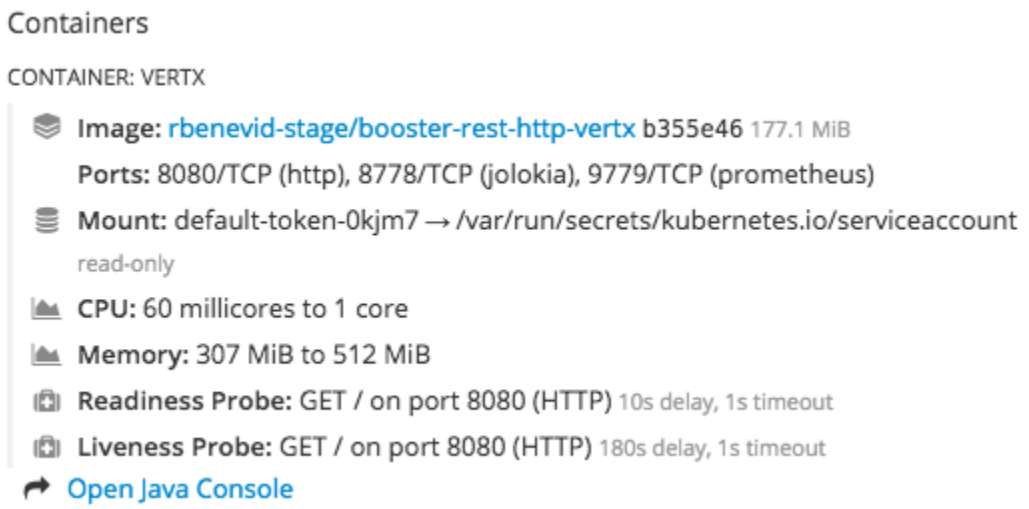
You can check the "Debug" tab and watch the variables. Note that you can also "Resume", "Step Into, "Step Over", and "Step Out".

Finish your debugging

Save Memory

Each OpenShift POD has a memory limit of 512Mi. That means that the one single application will consume almost 1Gb of memory for Stage and Run environments.

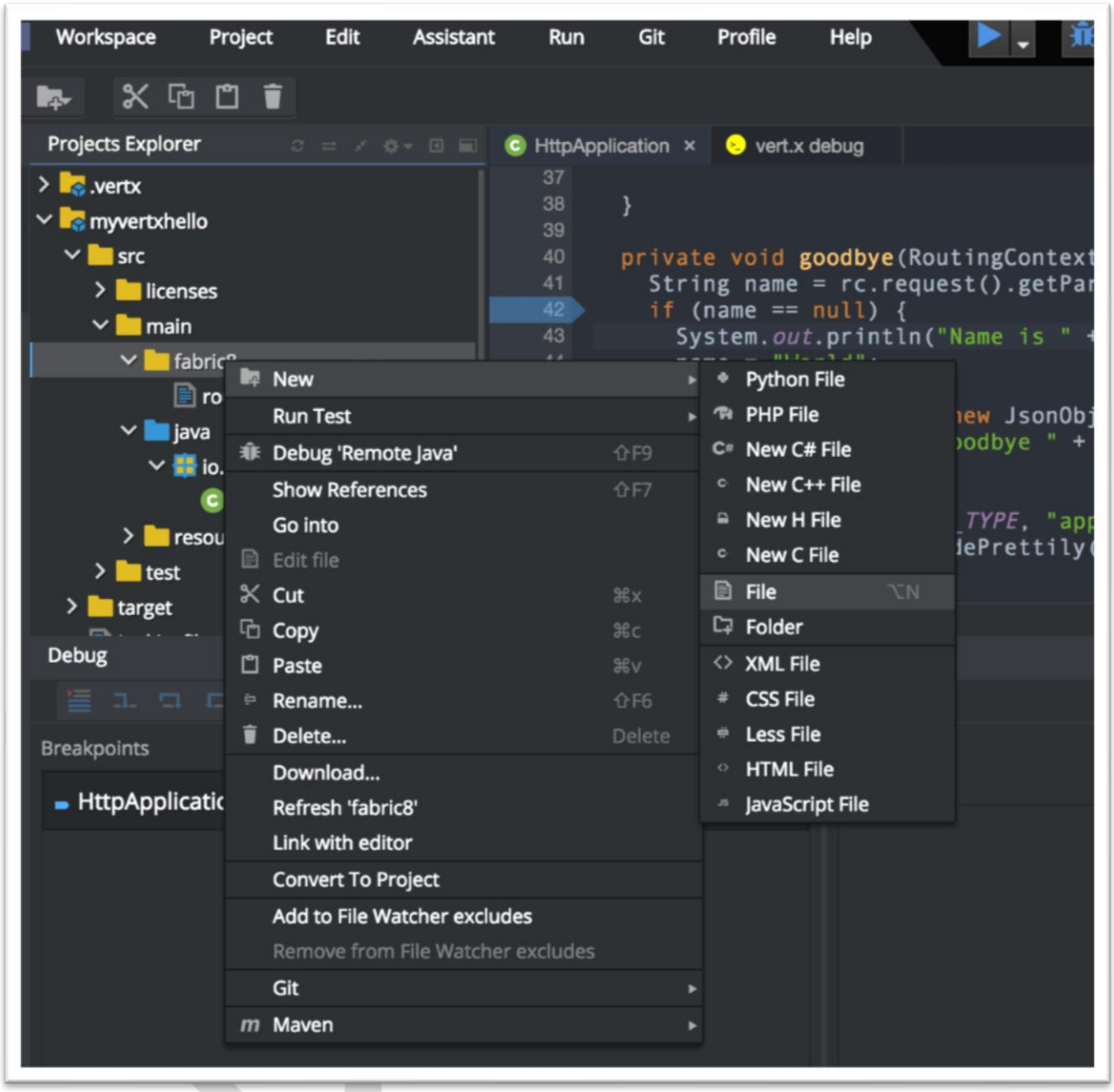
You can check the memory limit by opening the OpenShift Online console and visiting the pod detail.

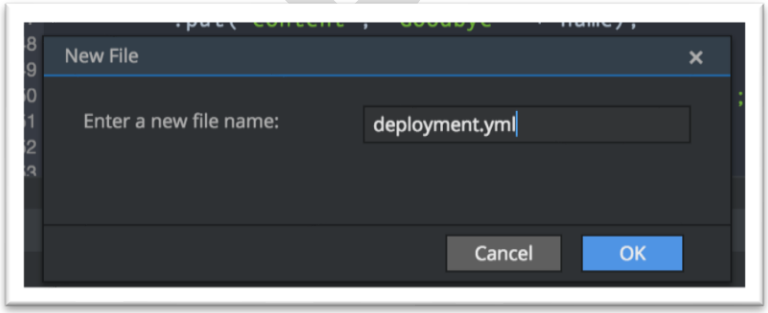


It's easy to reconfigure the quickstart to use less memory. This can be important as memory is limited in OpenShift.com's Free Tier quota.

To limit the memory use, we'll add a new deployment.yml file to constrain the memory to 250Mi

To create the file, right click on src/main/fabric8:





And add the following text (note: the spaces really matter in yaml):

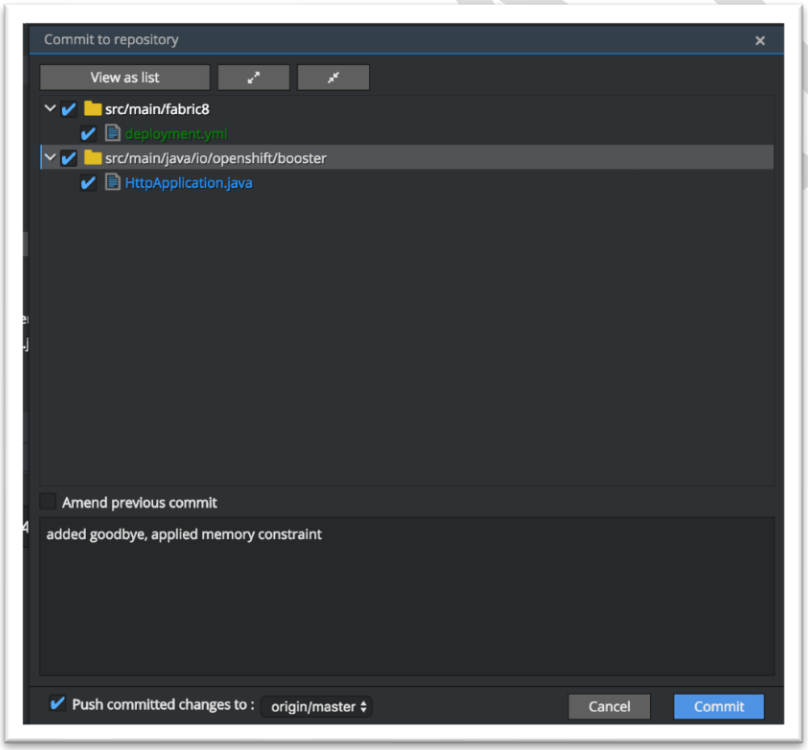
```
spec:
  replicas: 1
  template:
    spec:
      containers:
        - resources:
            limits:
              memory: '250Mi'
```

It is best to copy and paste from here:

https://raw.githubusercontent.com/burrsutter/vertx-eventbus/master/src/main/fabric8/deployment.yml

Save your edits in the new file.

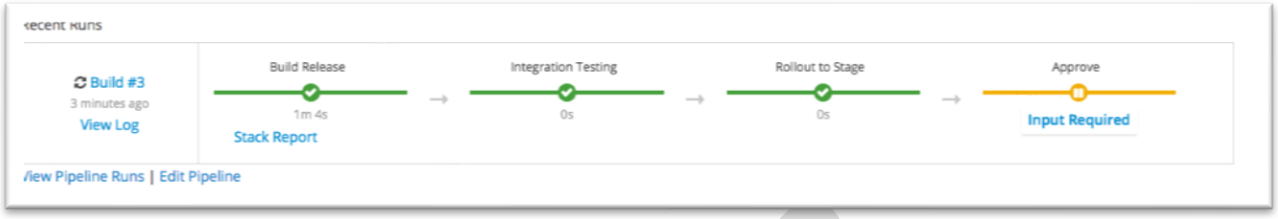
To make the changes active, perform a git commit and push:

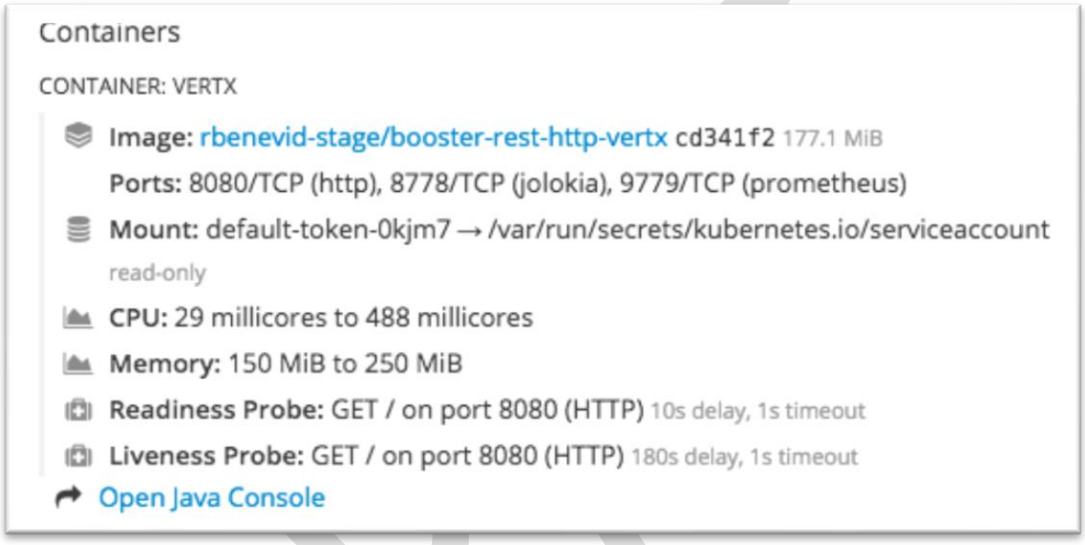


The memory constraint will give you more headroom inside of OpenShift.com's Free Tier quota.

It is important that you understand how quota is managed inside of OpenShift.com.

If you want to monitor, wait the pipeline to deploy the application in the Stage environment. Once that the application is in the Stage environment,open the OpenShift Online console and visiting the new pod detail.





Don't forget to approve the pipeline to send this version to the Run environment.

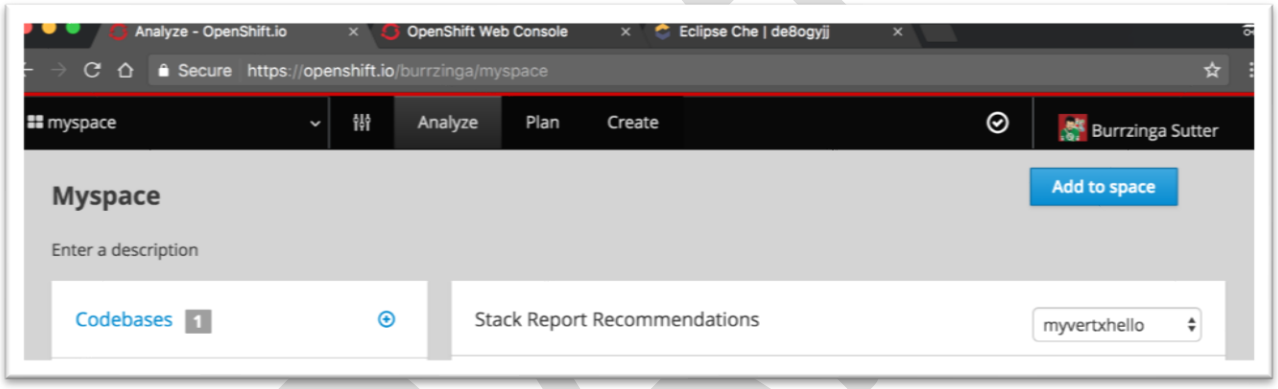
Spring Boot Quickstart

We'll now create a 2nd quickstart project. The first quickstart that we created was a Vert.X application. This second quickstart will be based on Spring Boot.

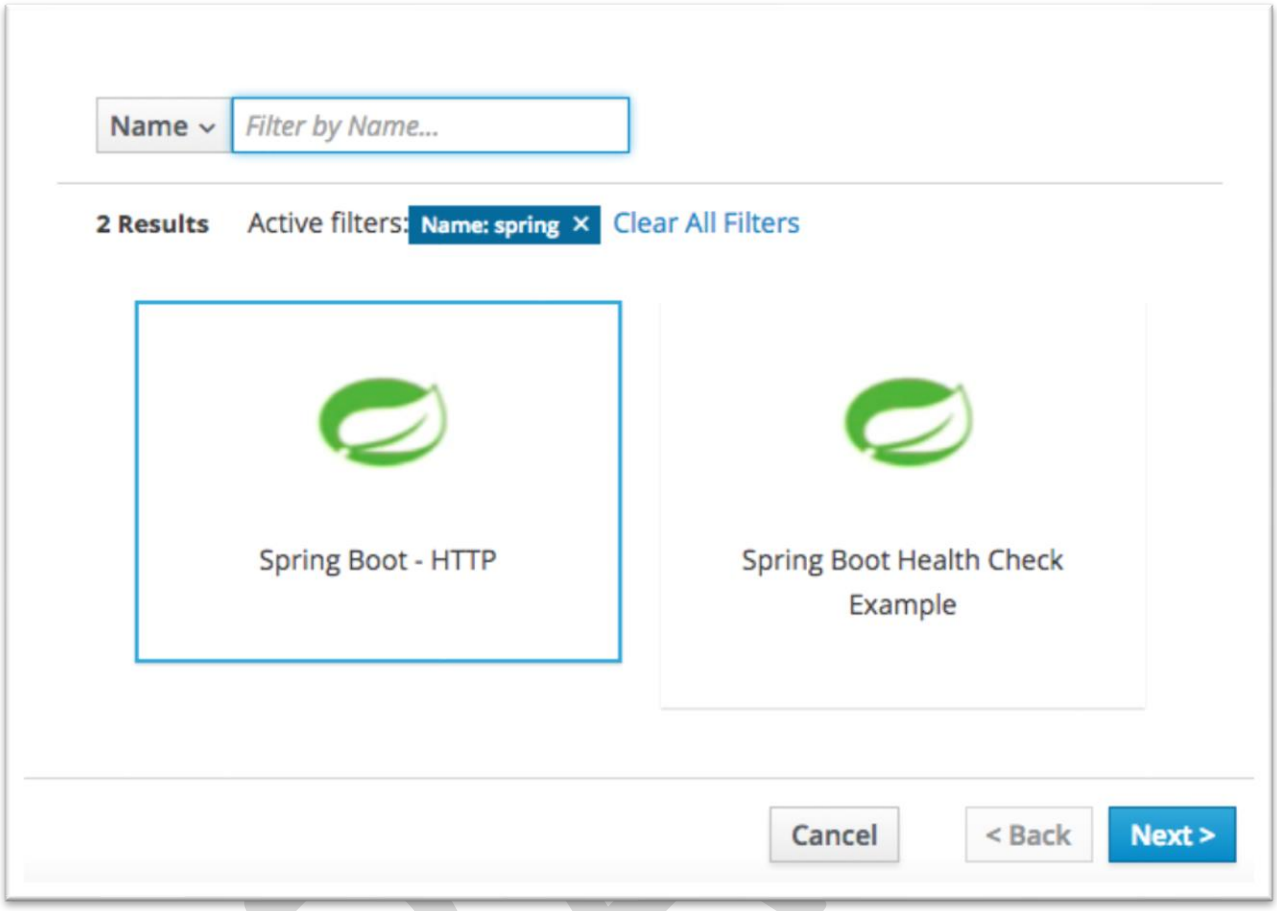
Many of the steps that we'll follow for this second quickstart will be similar to those that we performed when we created the first quickstart, so we'll cover those steps in less detail.

Creating a Spring Boot quickstart

In OSIO, in the Analyze Tab, click on "Add to Space" to start creating the new quickstart:

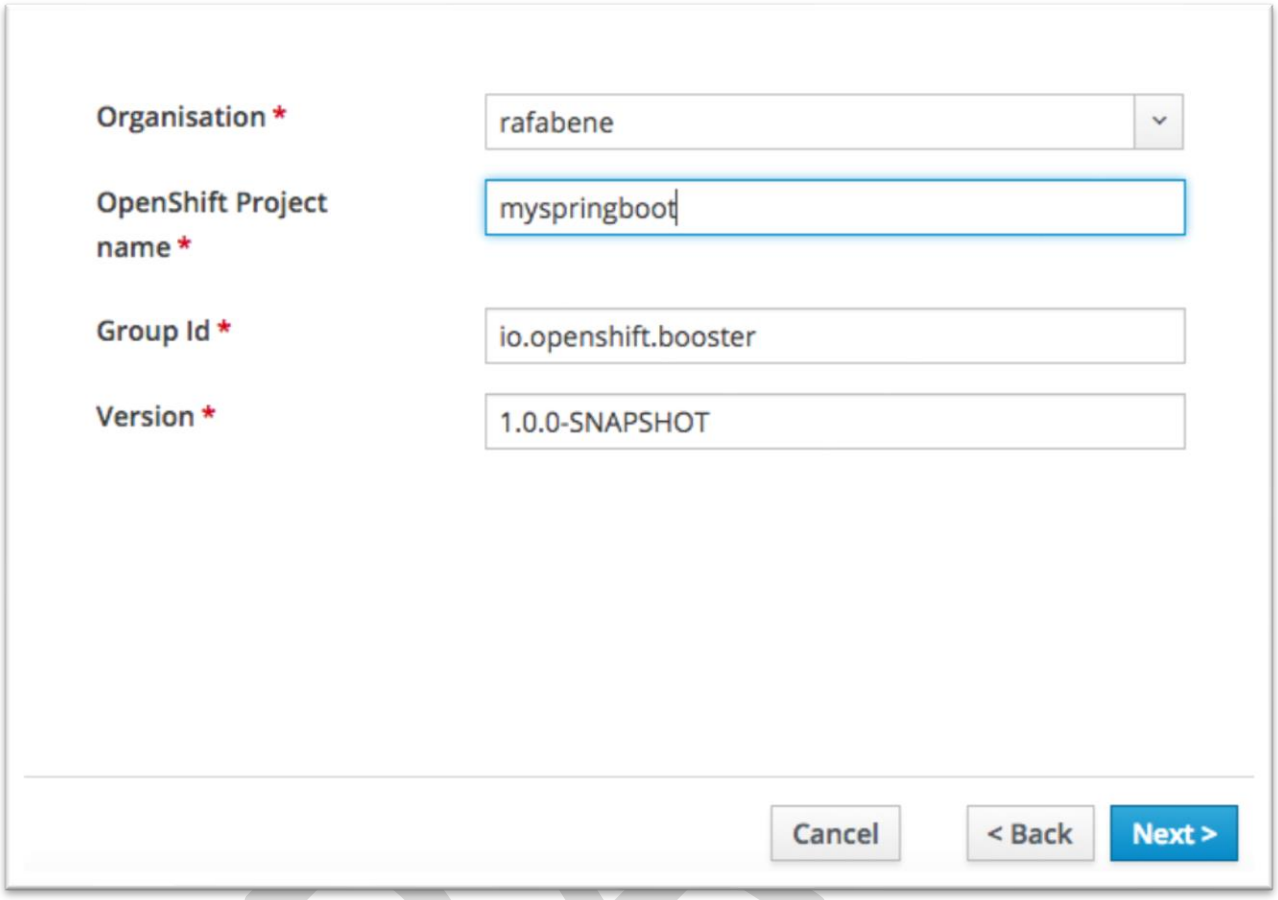


In the new quickstart dialog, type "spring" in the filter and type ENTER. Then select "Spring Boot - HTTP" for the project type, click the Next button:



In the "Step 1B" change the name of the application, accept the default options and click the Next button.

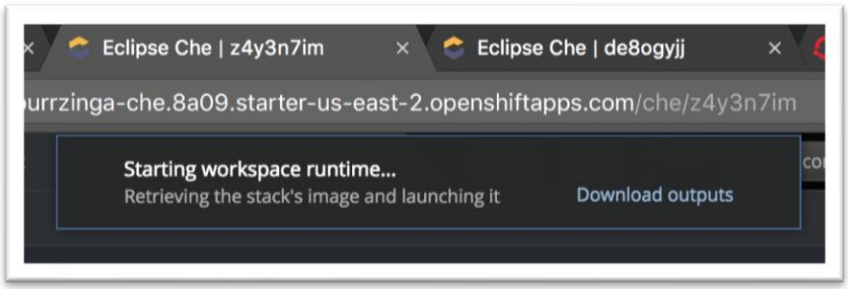
- Name: myspringboot
- Accept the defaults for the other fields.

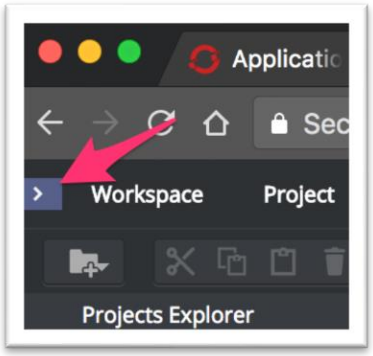


select the “Release, Stage, Approve and Promote” pipeline and click the Next and then the Finish button.

Create Tab - Codebases - to Create a Che Workspace

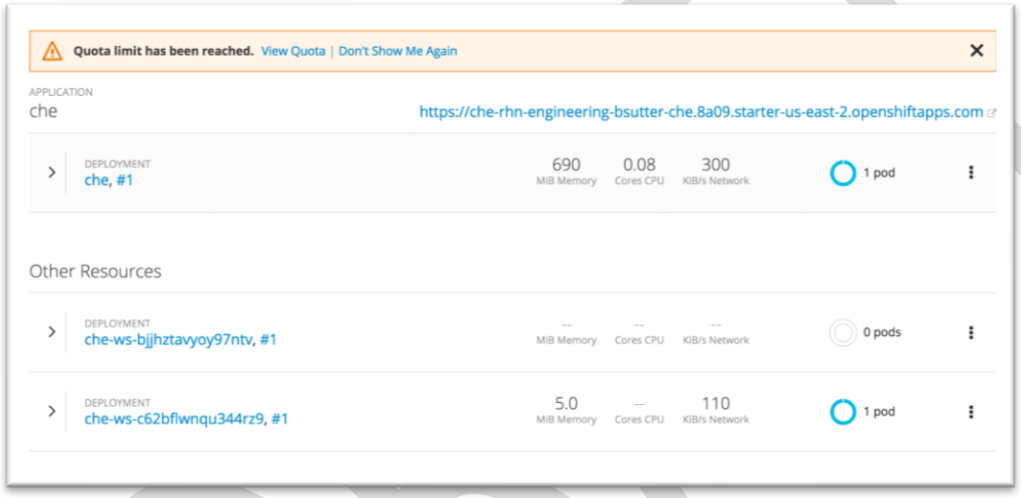
As we did for the first quickstart, the next step is to create a Che workspace. Note that the OpenShift Online Free tier has resources to support one running workspace at a time. When you start your new workspace for the Spring Boot quickstart, your first workspace should be automatically stopped. If it is not stopped, go to the Che dashboard and stop it before proceeding.





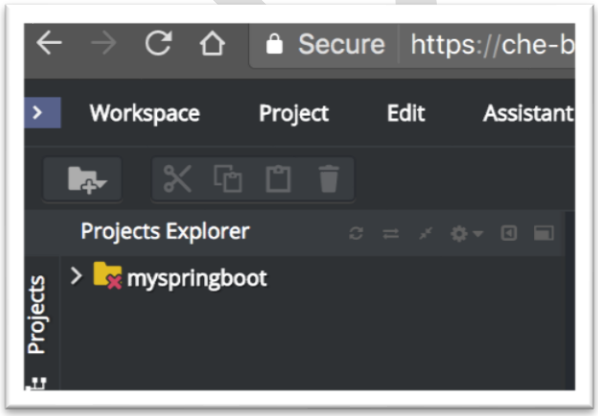
Che Dashboard link inside of a Che Workspace

If you lose track of your workspaces, visit the Che Dashboard which is an exposed route for the “che” in username-che via the OpenShift.com console:



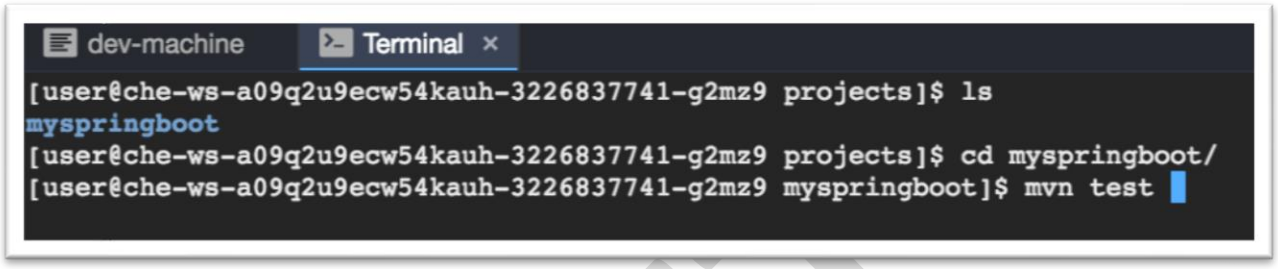
https://che-**<USER ID>**-che.8a09.starter-us-east-2.openshiftapps.com/dashboard/#/workspaces

Once the workspace is created and opens, If you get the Red X error just hit the browser refresh:



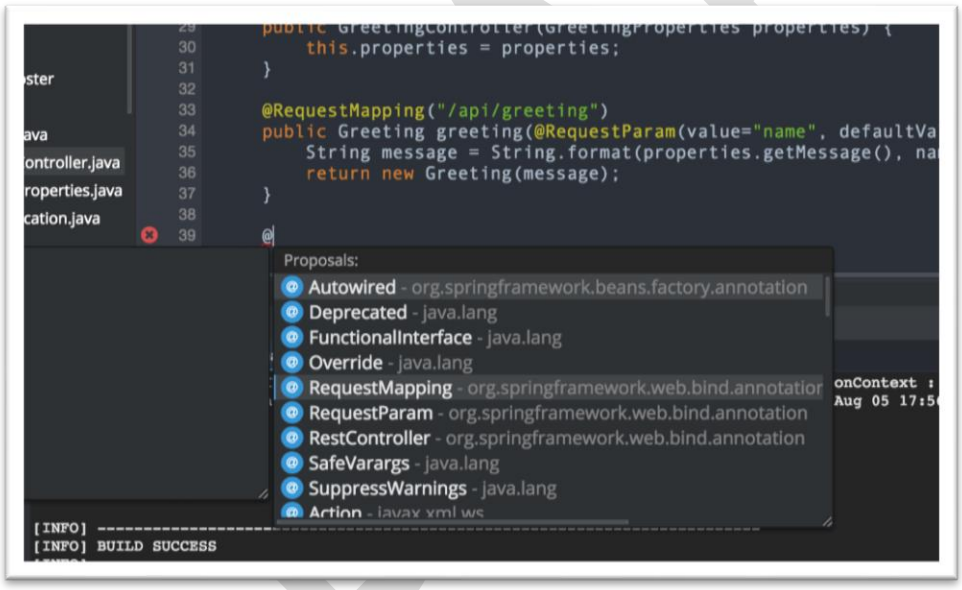
In the Che workspace, go to Terminal option and and execute this command:

```
cd myspringboot

mvn test
```



Watch out for failing tests, you do not wish to git commit/push your code change if it fails tests!

Open the file at src/main/java/io.openshift.booster/service/GreetingController.java and add a goodbye endpoint inside of GreetingController.java by typing “@” and hitting Ctrl-Space:

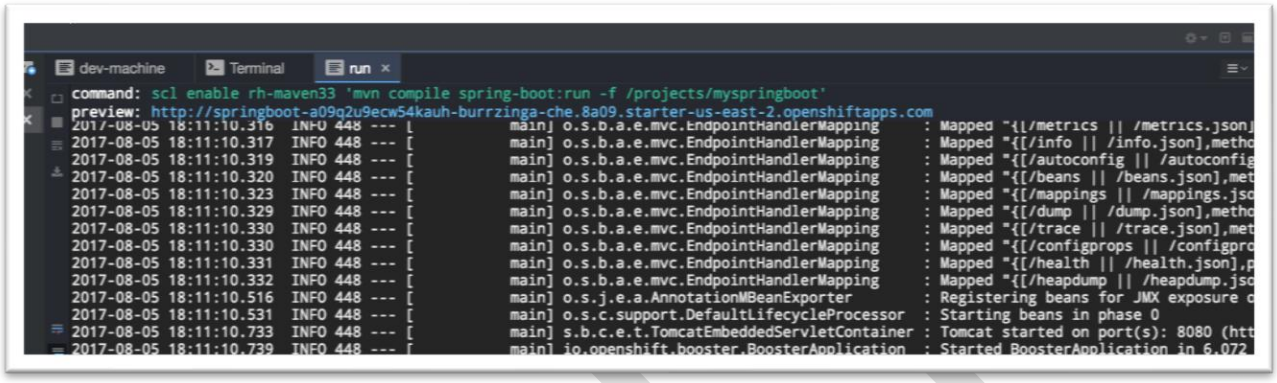


Select RequestMapping and add this code:

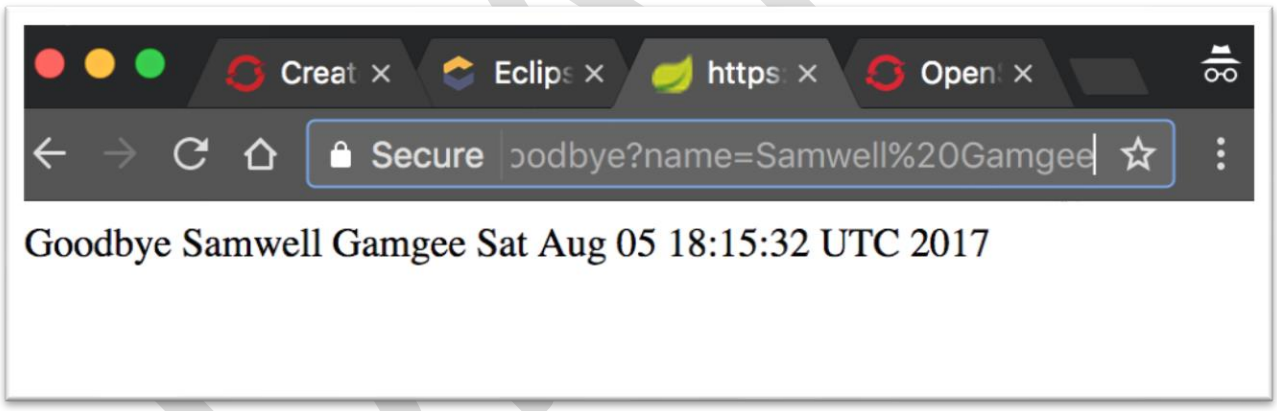
```
@RequestMapping("/api/goodbye")

public String goodbye(@RequestParam(value="name", defaultValue="World") String name) {

    return "Goodbye " + name + " " +  new java.util.Date();

}
```

After you make this change and save it, run quickstart. You'll have to run the quickstart manually as by default Spring Boot does not have hot redeployment.

Again see it in the Preview URL (the blue text):



Add the "/api/goodbye?name=Samwell%20Gamgee", without the quotes to the end of the Preview URL:



Reconfigure the quickstart to make it small enough to fit alongside the previously deployed Vert.x pod by adding a new deployment.yml under src/main/fabric8/ containing:

```
spec:
  replicas: 1
  template:
    spec:
      containers:
        - resources:
            limits:
              memory: '250Mi'
```

Again, you can copy and paste from https://raw.githubusercontent.com/burrsutter/vertx-eventbus/master/src/main/fabric8/deployment.yml

And finally, perform a git commit and git push to kick off the Spring Boot quickstart's pipeline.

At this point, you have successfully run two quickstarts/apps/microservices but you are starting to push the boundaries of your available quota for OpenShift.com!

Troubleshooting

Asdsadasd

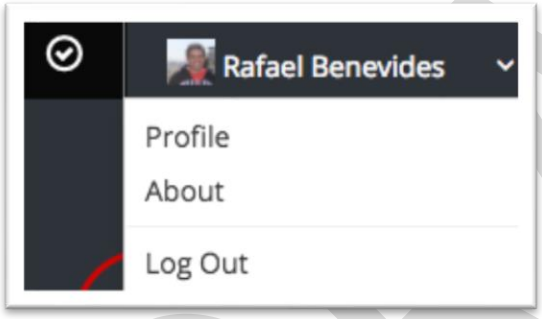
Update Tenant Services

Sdasdasd

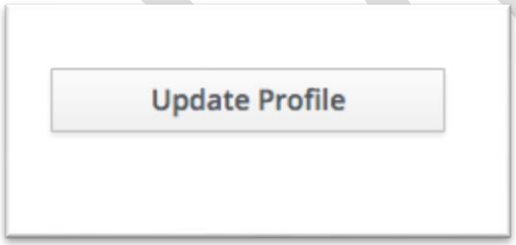
“Factory Reset”

OpenShift.IO generates a large number of artifacts across 5 different OSO Projects (aka Kubernetes Namespaces). All of these OSO Projects have substantial limits based on their assigned quotas. The best way to stay ahead of the curve is to periodically clean things up.

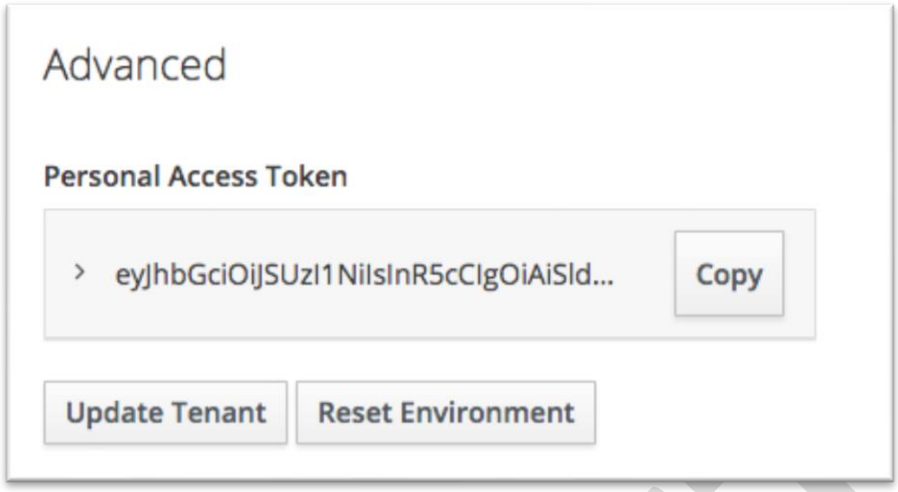
To do this, open your Profile options by clicking in the menu with your username and select "Profile"



At the following page, there will be a button called "Update Profile". Click on it, to edit your profile::

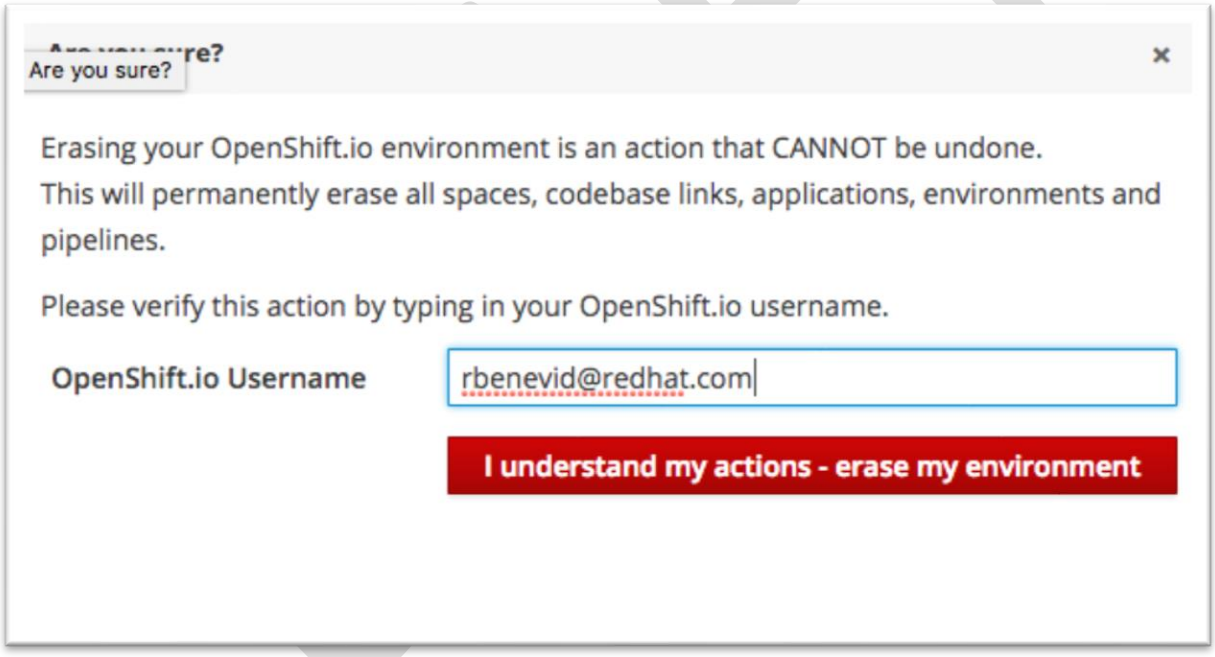


In the "Edit Profile" page, roll to the "Advanced" and click on "Reset Environment" button



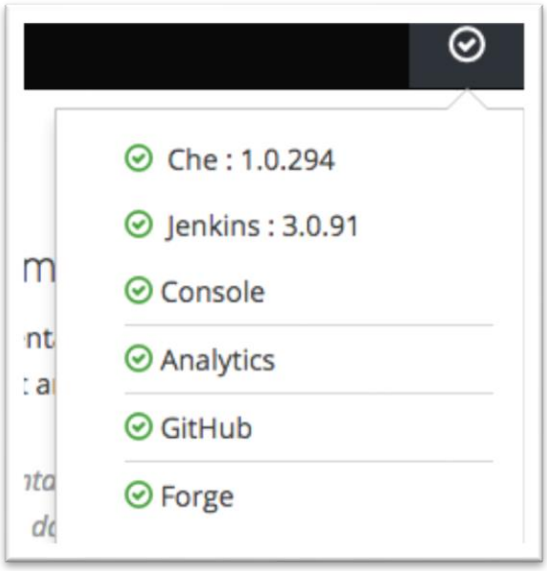
In the next page, roll the page and click on the red "Erase My OpenShift.io Environment"

You will be prompted to confirm your action by typing your username.



Once that you confirm the operation, this will reset your environment, cleaning out the various pods/services/dcs/bcs/cms/etc from your 5 OSO projects, it does NOT remove the projects but cleans inside them. Return to "Account Home" and wait for the environment to be ready again. It should take 5 minutes to load your newly updated tenant.

Check on the Che and Jenkins happiness by visiting them in the OpenShift.com (OSO) Console and/or checking the status widget



Import Code

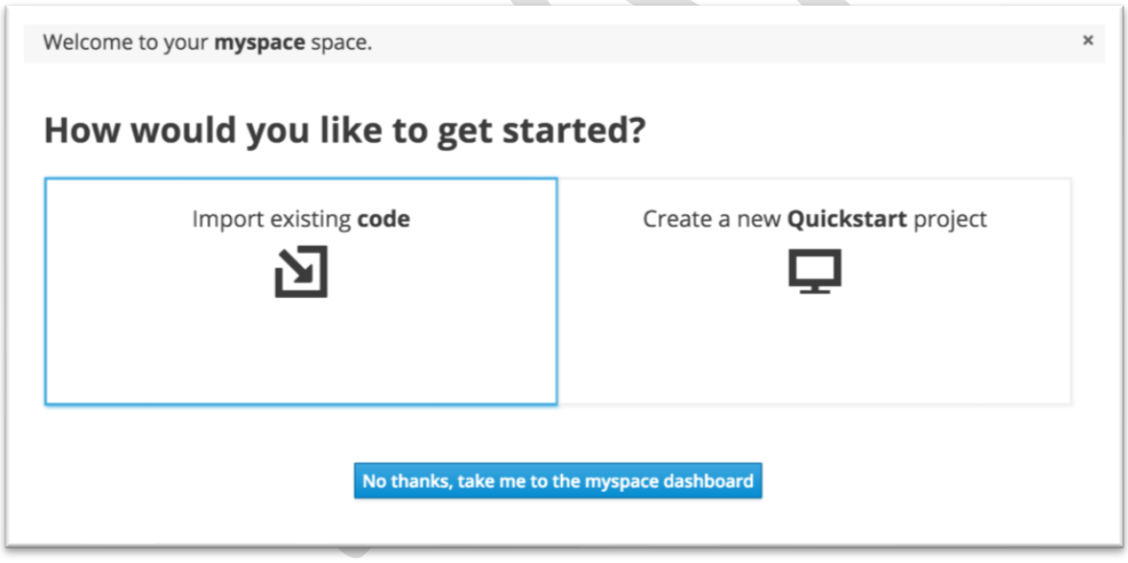
In addition to creating quickstarts from scratch, you can also add another project via the Import from github feature.

We can import the following sample github repos. To import these repos, first create your own fork of each of these. (Remember that projects == pods, and 2 pods run within the OpenShift Online Free quota.)

- https://github.com/burrsutter/vertx-eventbus
- https://github.com/burrsutter/vertx-paint
- https://github.com/burrsutter/vertx-wiki2

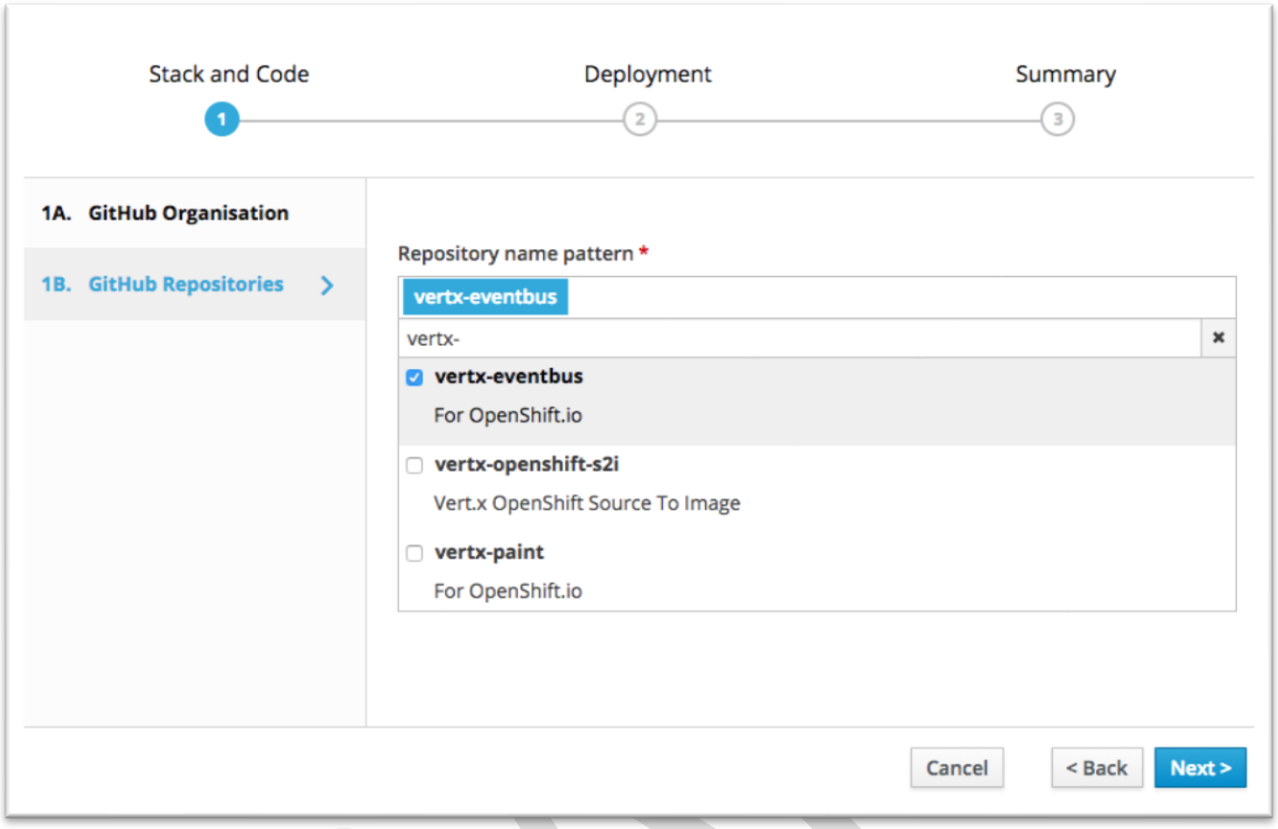
Note: while you can import all 3, it is best if you import them one at a time as it can take 12 to 15 minutes if you attempt all 3 simultaneously.

In OSIO, in the Account Home page, after you have created the “myspace” space, click on “Add to Space”:

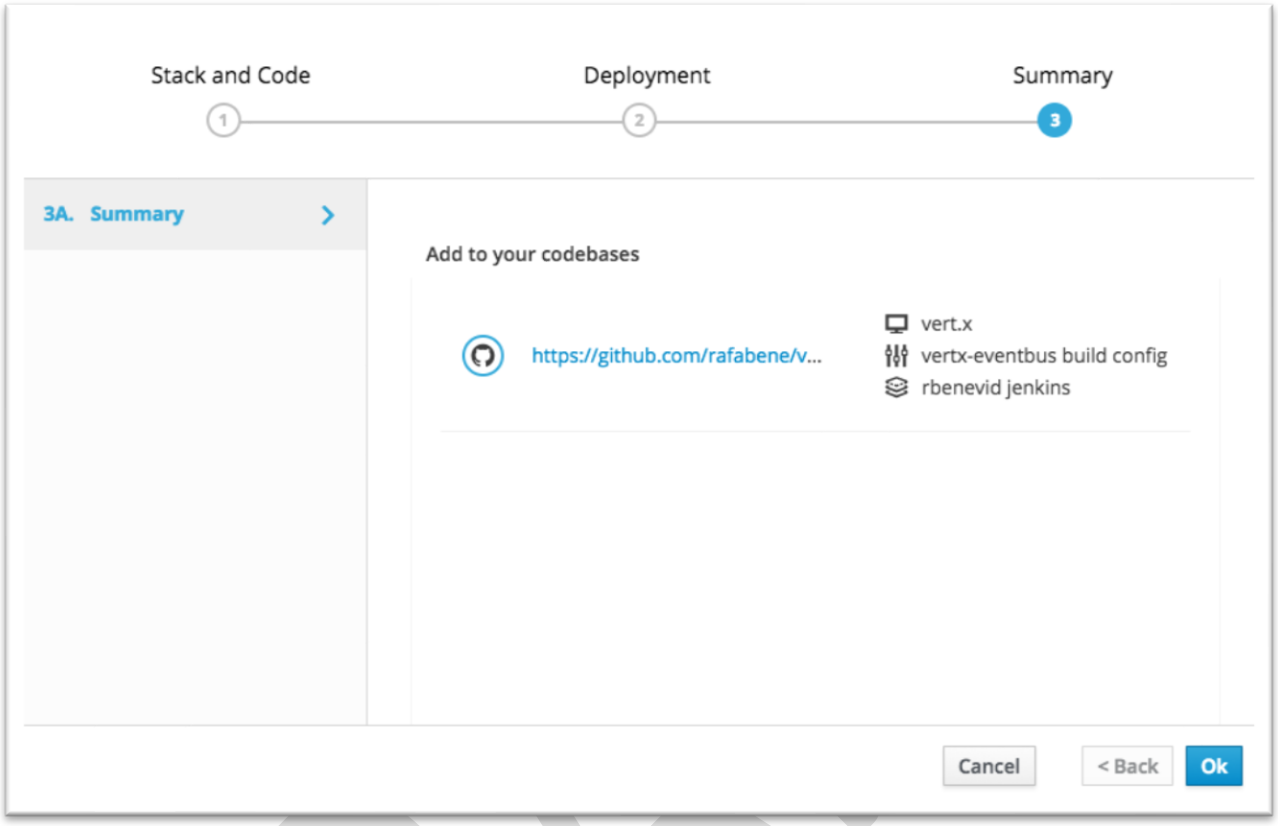


Click on Import existing code. Select the Github Organization. In the next page of the dialog, the “Repository name pattern” field will automatically query Github for all of your repos.

Select “vertx-eventbus”:



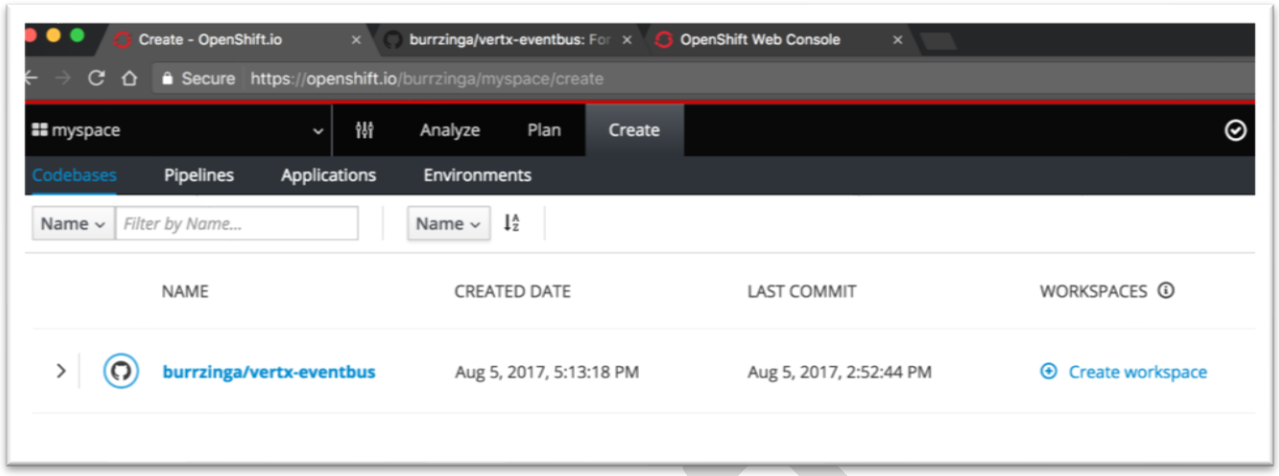
Then click the Next button. Select the "Release, Stage, Approve and Promote" pipeline and click Next and then on the Finish button:



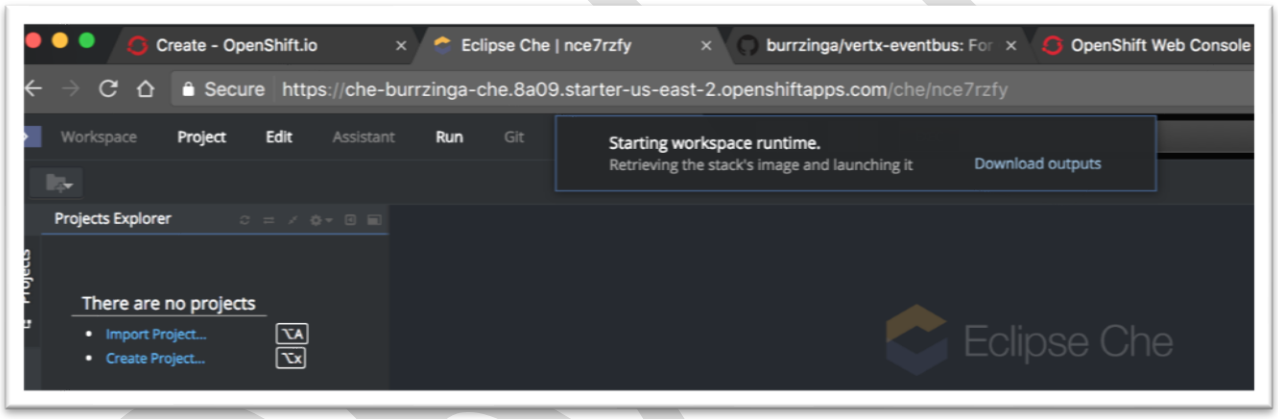
Click the OK button.

Create Tab - Codebases - to Create a Che Workspace

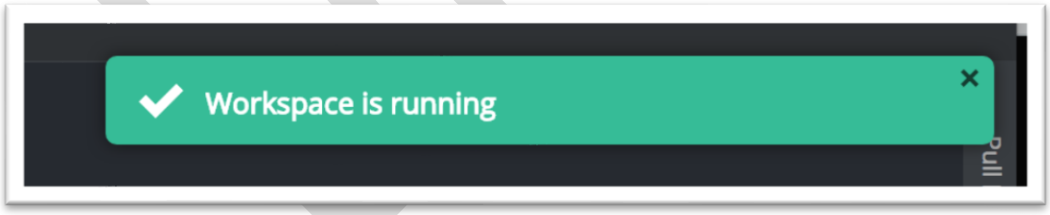
Click on Create Tab and then click on "Create Workspace" to create a new workspace:



If you previously checked the “Always allow pop-ups…” then it should open the new tab right away:



Look for the Workspace Running message:

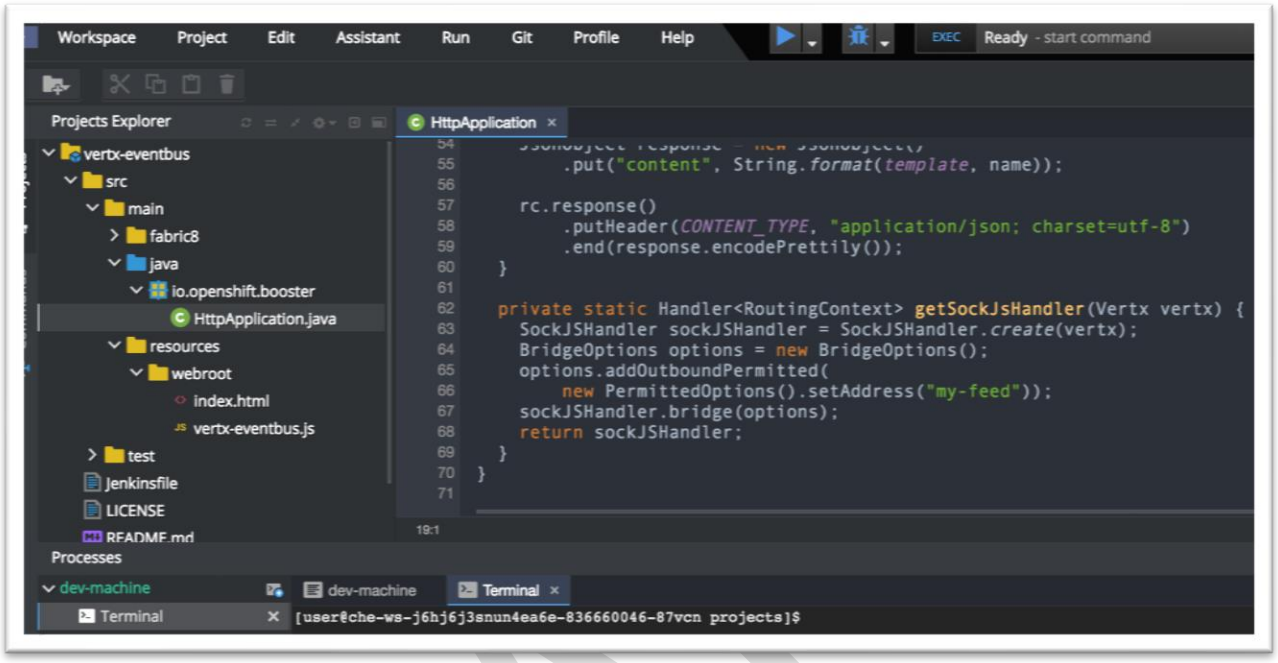


Add a couple of exclamation points to the “Aloha” string in the HttpApplication.java file:

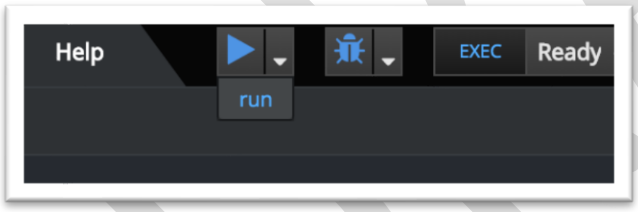
```
protected static final String template = "Aloha !!, %s!";
```

Also, check out the new route for the “eventbus” and its supporting method, this sets up the eventbus to allow outbound-only communications for the “my-feed” channel. Any messages

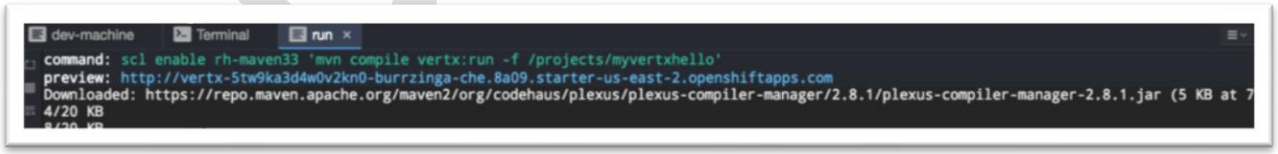
arriving in the “my-feed” channel will be “pushed” to the browser via this SockJS bridge. You will also see a vertx-eventbus.js file alongside index.html and some tweaks to index.html itself.



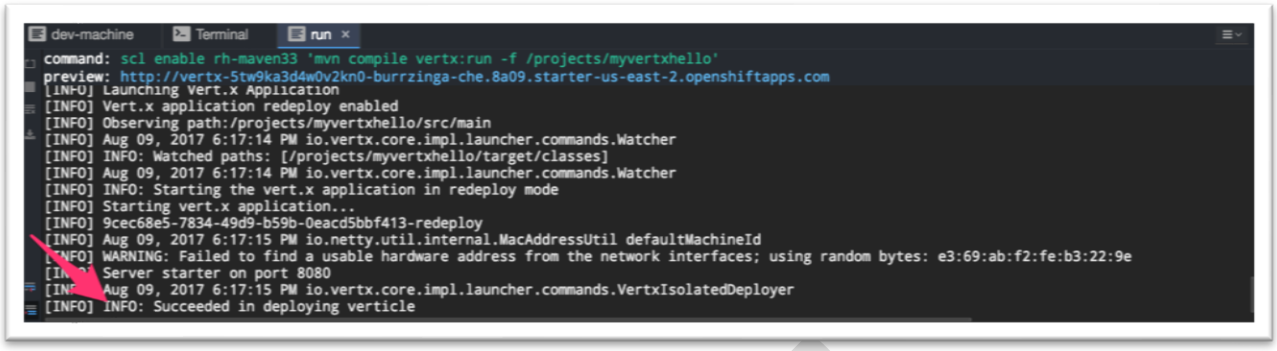
Select Run from the Che menu bar at the top of the screen - this sets up the Vert.x server and gives you hot redeploy:



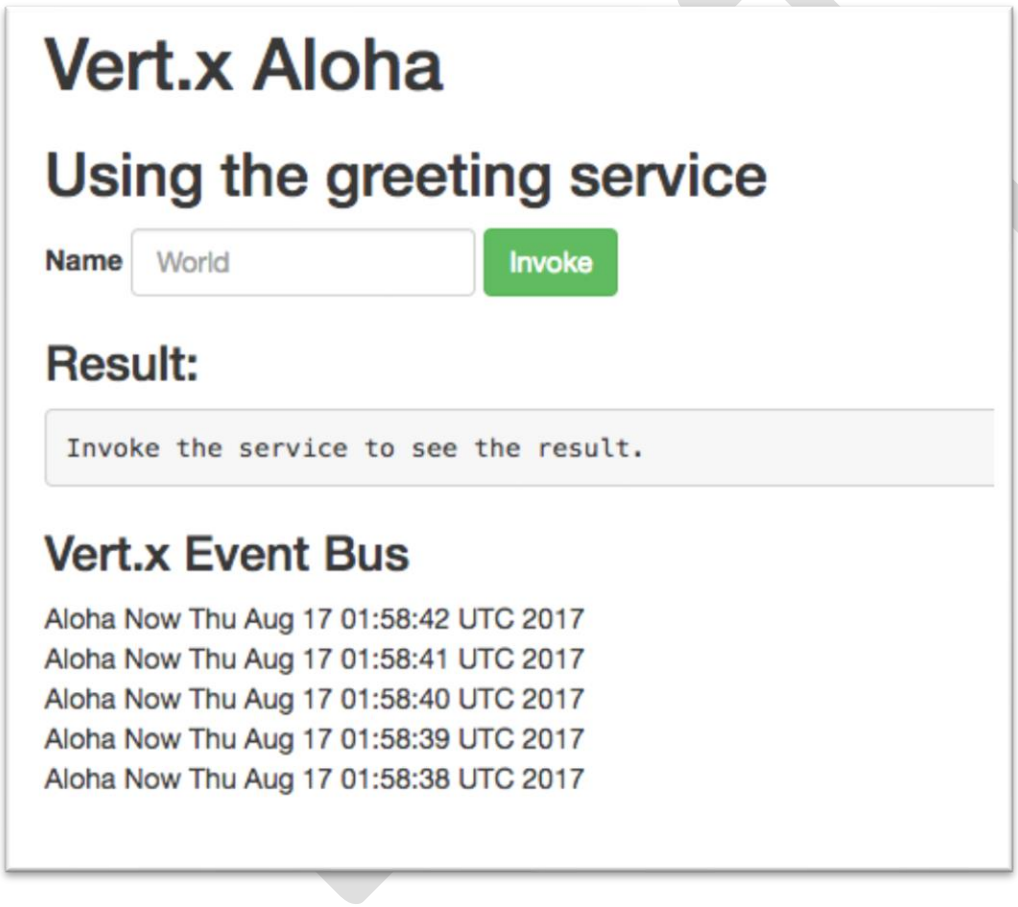
Wait for the mvn build to download its dependencies:



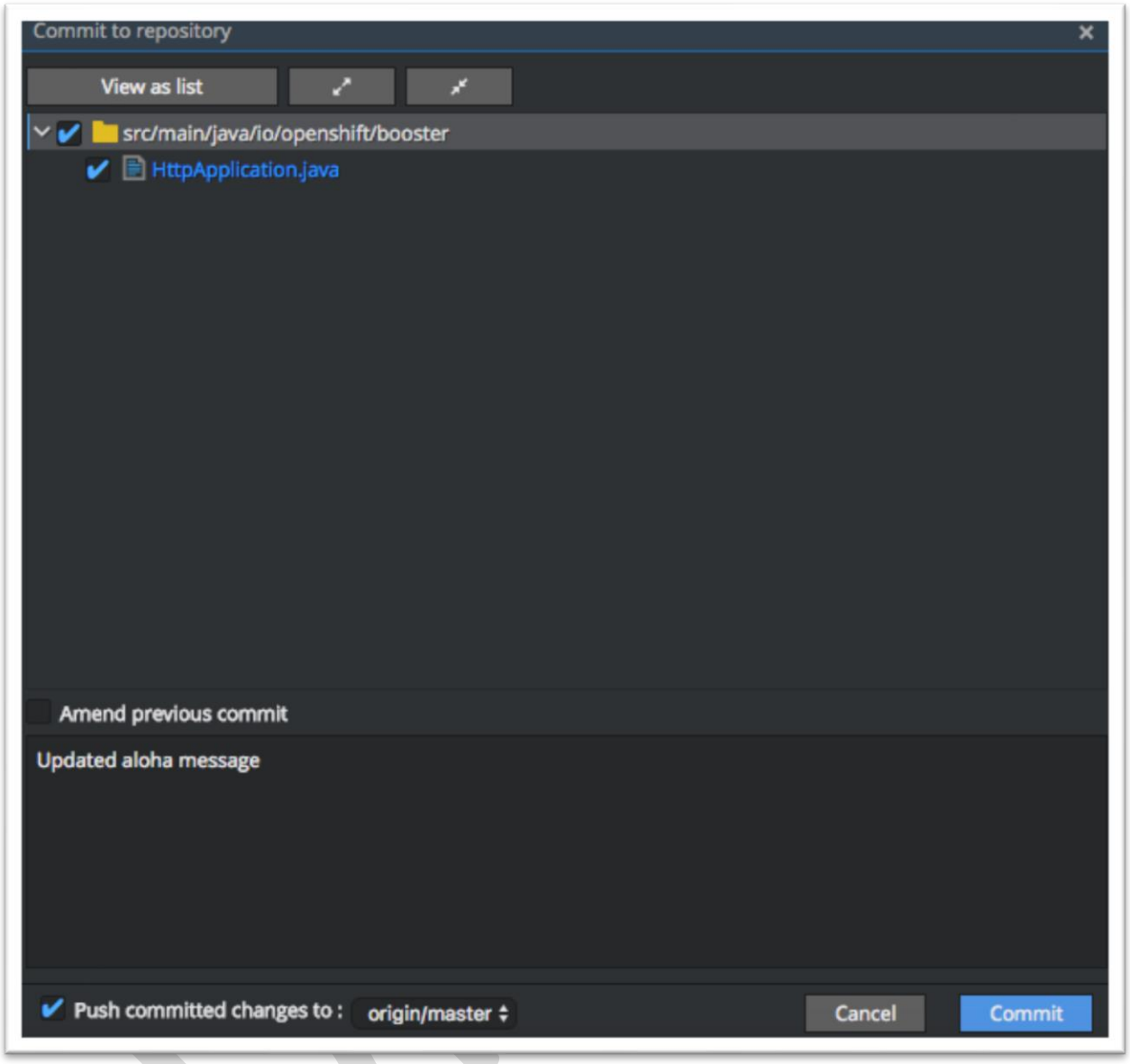
Look for the logging statement: “INFO: Succeeded in deploying verticle”



Click on the Preview URL to invoke the quickstart:



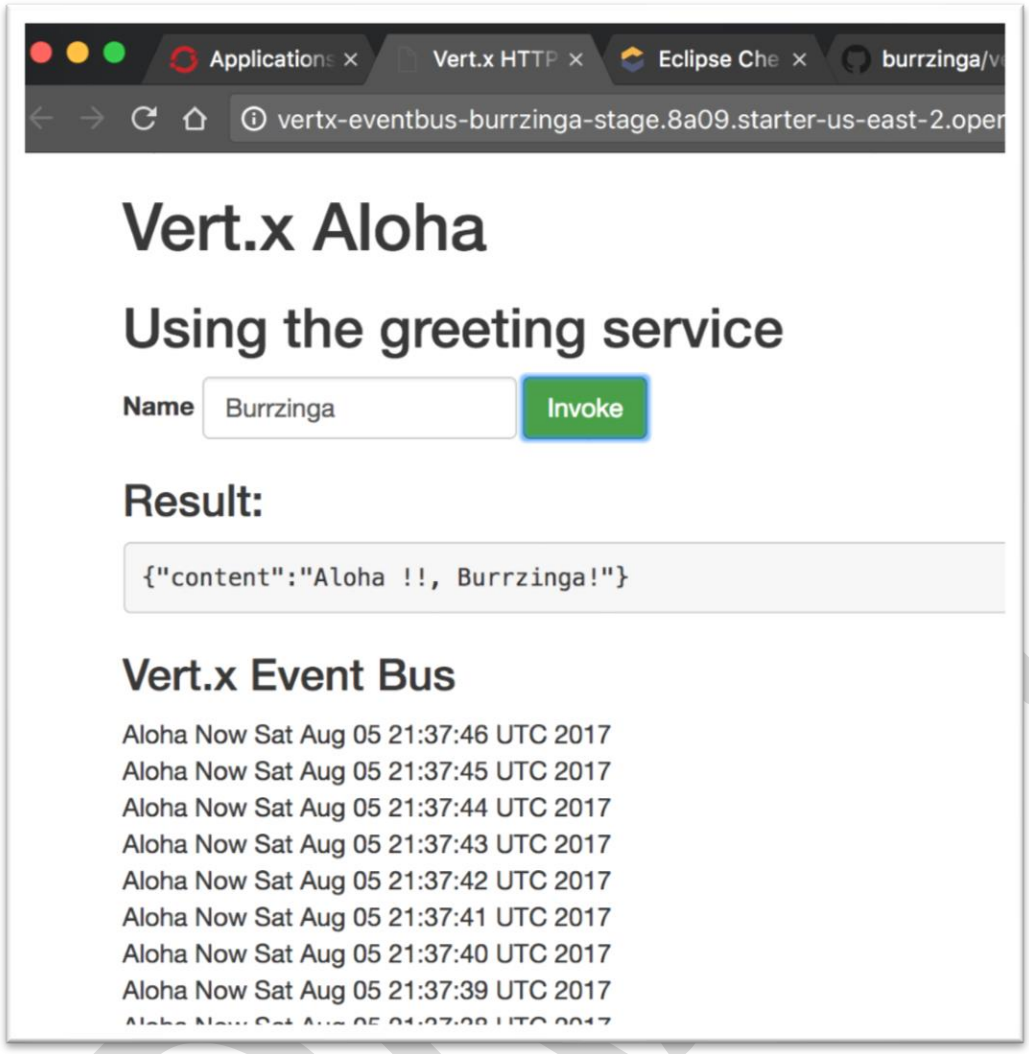
Then, perform a git commit and git push (git->remotes->push). Go to the Che menu "Git -> Commit". You can perform "commit" and "push" at the same time by checking the "Push committed changes to" checkbox.



See the changes flowing through the pipeline

Go to the Pipelines view of OpenShift.io and **Abort the Pipeline #1.** That will allow the Pipeline #2 to start with the modifications that we did.

Open the Stage URL and see the modifications in the Stage environment. Then return to the pipeline and Promote it to the Run environment.


Applications
Vert.x HTTP
Eclipse Che
burrzinga/v
vertx-eventbus-burrzinga-stage.8a09.starter-us-east-2.oper
Vert.x Aloha
Using the greeting service
Name
Burrzinga
Invoke
Result:
{"content":"Aloha !!, Burrzinga!"}
Vert.x Event Bus
Aloha Now Sat Aug 05 21:37:46 UTC 2017
Aloha Now Sat Aug 05 21:37:45 UTC 2017
Aloha Now Sat Aug 05 21:37:44 UTC 2017
Aloha Now Sat Aug 05 21:37:43 UTC 2017
Aloha Now Sat Aug 05 21:37:42 UTC 2017
Aloha Now Sat Aug 05 21:37:41 UTC 2017
Aloha Now Sat Aug 05 21:37:40 UTC 2017
Aloha Now Sat Aug 05 21:37:39 UTC 2017