Integrate Slack

Objective

Bluemix DevOps services include toolchains working together to support your development and deployment tasks. The integrations across a toolchain help ease the workflows between the tools so they work in synergy. This lab demonstrates how to set-up and use Bluemix toolchains with Slack to make your move to DevOps easier. **Note:** Toolchains are currently available in the US South region only and the instructions in this lab are written for the US South region.

This lab uses the simple toolchain for the Orders API microservice.

Teaming

Software development is a team activity. The lab scenario shows how Bluemix Continuous Delivery tool integrations can be used to alerts teams when activities, such as builds or deployments, occur.

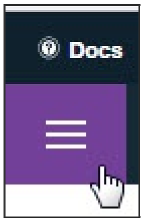
Tasks

- Task 1: Integrate Slack into Toolchain
- Task 2: Test Slack Integration
- Task 3: Break the application build
- Task 4: Fix application

Task 1: Integrate Slack into Toolchain

The visual display of the Delivery Pipeline is great if you want to watch it. But if you don't want to watch or if you did not even start the build, how do you get notified of events that you are interested in? One way is Slack. Slack provides real-time messaging for team communications. You can integrate Slack with your Bluemix DevOps Services project so that notifications about build results from your Delivery Pipeline are posted on a Slack channel.

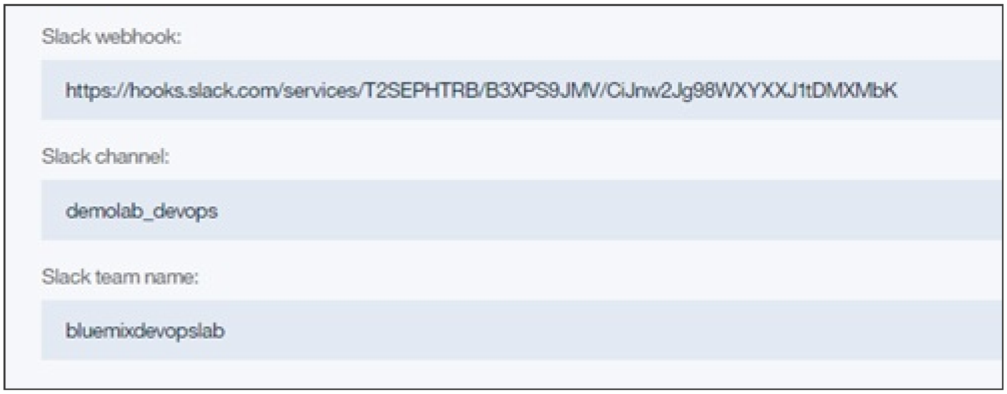
1. If you are not on the Toolchains page (if you don't see a button called *Create a Toolchain*), cick on the **Bluemix menu bar** in the upper left corner.

   and click on **Services** then **DevOps**.
2. Click **simple-order-toolchain-*timestamp***.
3. On the **simple-order-toolchain-*timestamp*** Toolchain page, click **Add a Tool**.
4. Select **Slack**
5. On the Slack Configuration page:
   - Enter the following all as one string as the Slack webhook. This is also available for copying and pasting at: http://ibm.biz/SlackWebHook.

```
https://hooks.slack.com/services/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
```

- Enter **#demolab_devops** as the Slack channel.
- Enter **BluemixDevOpsLab** as the Slack team name.



Slack webhook:

https://hooks.slack.com/services/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

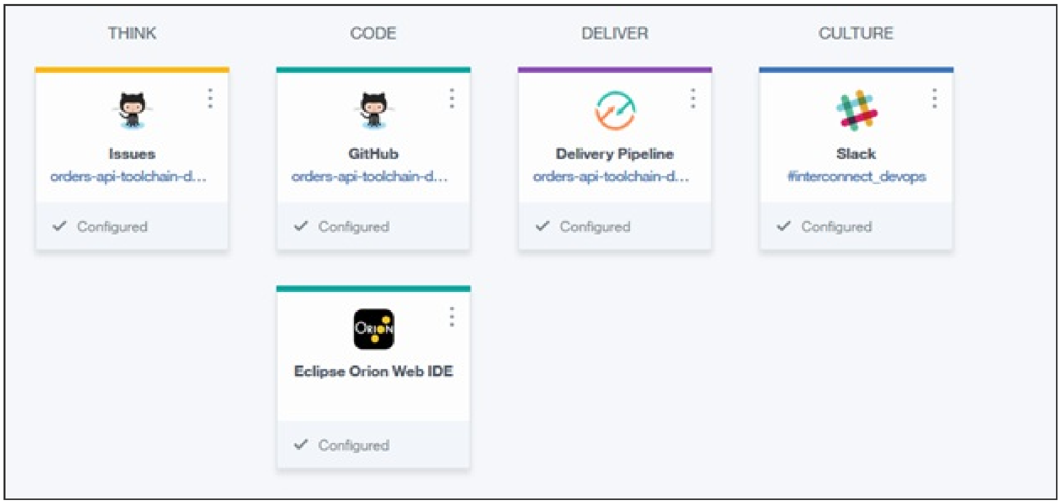
Slack channel:

demolab_devops

Slack team name:

bluemixdevopslab

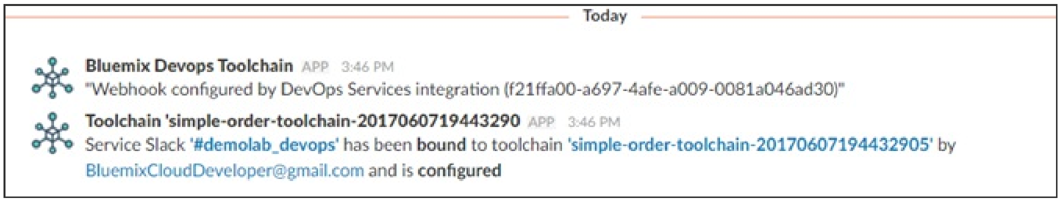
6. Click **Create Integration**
7. The integration with Slack is added.



8. Switch to the browser tab that has Slack open. If the tab is closed:

- Open a new browser tab
- Go to the following URL to go to the Slack team https://bluemixdevopslab.slack.com
- If asked, enter the following information:
  - Email address: **BluemixDevOps@gmail.com**
  - Password: **bluemix4me** and click **Sign in**.

9. Slack notifies us that the integration was added.



Today

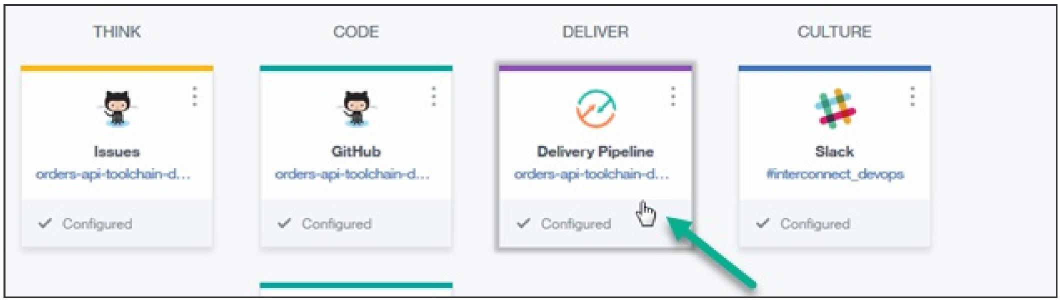
Bluemix Devops Toolchain APP 3:46 PM
"Webhook configured by DevOps Services integration (f21ffa00-a697-4afe-a009-0081a046ad30)"

Toolchain 'simple-order-toolchain-2017060719443290 APP 3:46 PM
Service Slack '**#demolab_devops**' has been **bound** to toolchain '**simple-order-toolchain-20170607194432905**' by BluemixCloudDeveloper@gmail.com and is **configured**

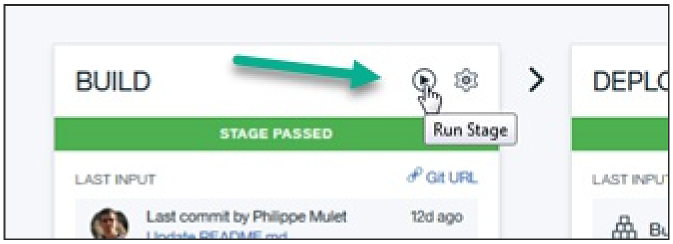
Task 2: Test Slack Integration

Looking at the *demolab_devops* Slack channel, we might see that other team members have been busy. Let's see what happens when we initiate a build.

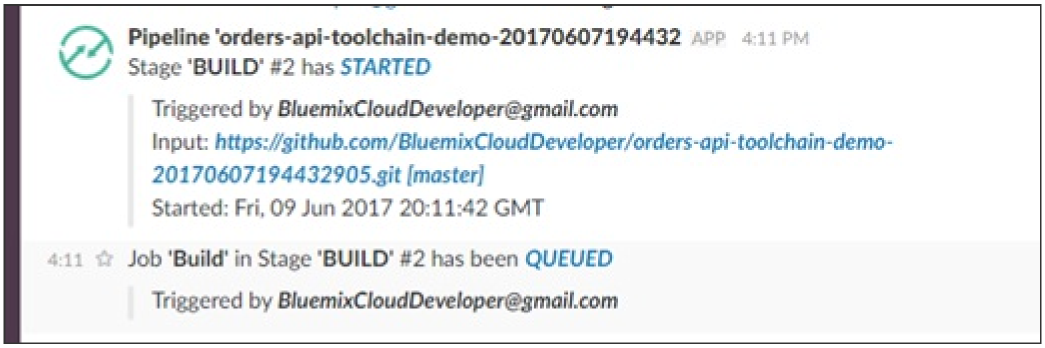
1. Switch to the browser tab displaying the Bluemix console. It should be displaying the **simple-order-toolchain-*timestamp*** Toolchain page. If not, click on the Bluemix menu bar in the upper left corner, click on Services then DevOps. Click on Toolchains. Then click on **simple-order-toolchain-*timestamp***.
2. Click on the **Delivery Pipeline** tile for the orders-api-toolchain-demo Delivery Pipeline.



3. Start an application build by clicking on the *BUILD* stage **Run Stage** icon.



4. Switching to the Slack browser tab, the Slack bluemixdevopslab channel displays the progress of this activity.



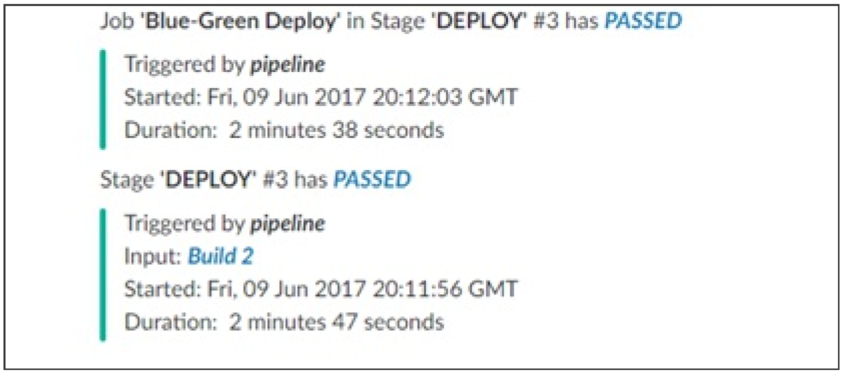
Pipeline 'orders-api-toolchain-demo-20170607194432 APP 4:11 PM
Stage 'BUILD' #2 has *STARTED*

Triggered by *BluemixCloudDeveloper@gmail.com*
Input: *https://github.com/BluemixCloudDeveloper/orders-api-toolchain-demo-20170607194432905.git [master]*
Started: Fri, 09 Jun 2017 20:11:42 GMT

4:11 ☆ Job 'Build' in Stage 'BUILD' #2 has been *QUEUED*

Triggered by *BluemixCloudDeveloper@gmail.com*

5. And the completion.



Job 'Blue-Green Deploy' in Stage 'DEPLOY' #3 has *PASSED*

Triggered by *pipeline*
Started: Fri, 09 Jun 2017 20:12:03 GMT
Duration: 2 minutes 38 seconds

Stage 'DEPLOY' #3 has *PASSED*

Triggered by *pipeline*
Input: *Build 2*
Started: Fri, 09 Jun 2017 20:11:56 GMT
Duration: 2 minutes 47 seconds

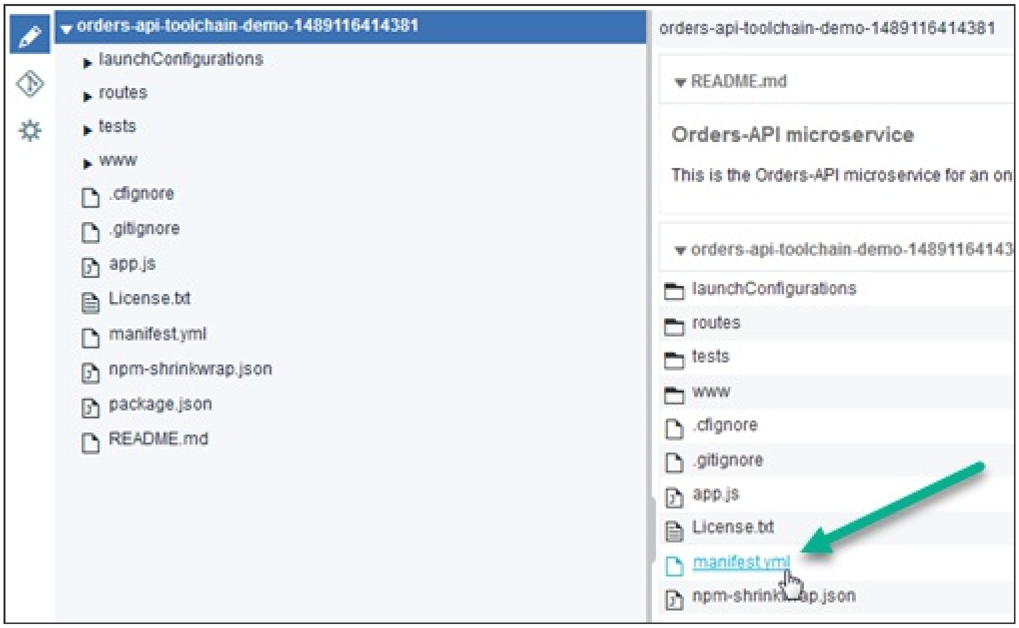
6. Through the Bluemix Continuous Delivery integration with Slack, teams get notified of events in the Delivery Pipeline.
7. Switch to the browser tab displaying the Bluemix console. Click the blue arrow in the upper left hand corner to return to the

Toolchain.

Task 3: Break the application build

Let's see what happens when the application build is broken.

1. On the **simple-order-toolchain-*timestamp*** Toolchain page, click the **Eclipse Orion Web IDE** tile.
2. In the file directory, click manifest.yml to open the file.



- orders-api-toolchain-demo-1489116414381
  - launchConfigurations
  - routes
  - tests
  - www
  - .cfignore
  - .gitignore
  - app.js
  - License.txt
  - manifest.yml
  - npm-shrinkwrap.json
  - package.json
  - README.md

orders-api-toolchain-demo-1489116414381

README.md

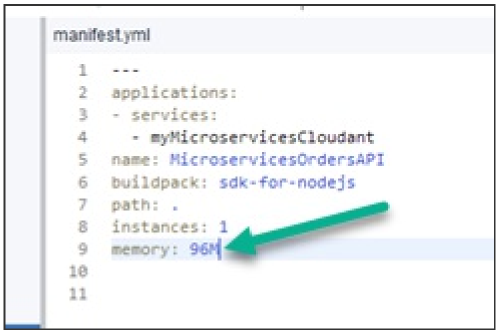
Orders-API microservice

This is the Orders-API microservice for an on

orders-api-toolchain-demo-14891164143

- launchConfigurations
- routes
- tests
- www
- .cfignore
- .gitignore
- app.js
- License.txt
- manifest.yml
- npm-shrinkwrap.json

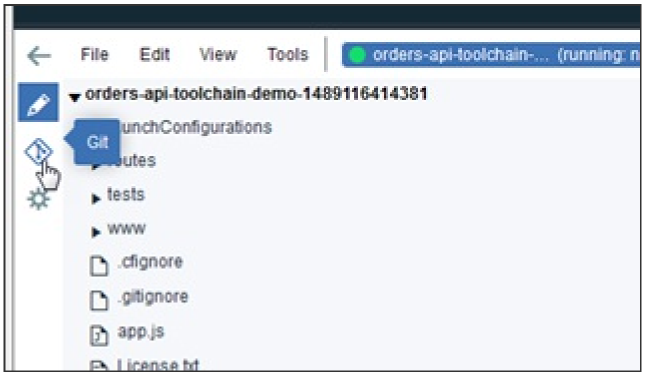
3. Update the value for memory to 96G. This setting intentionally increases your memory to exceed the quota for your org. Your changes are automatically saved.



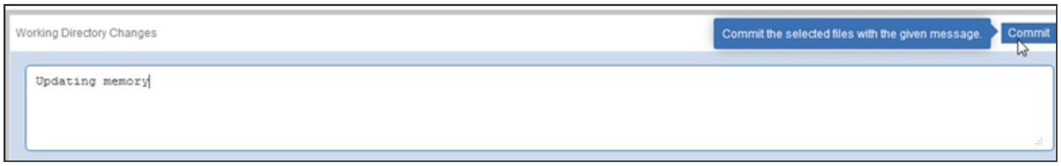
manifest.yml

```
---
applications:
- services:
  - myMicroservicesCloudant
name: MicroservicesOrdersAPI
buildpack: sdk-for-nodejs
path: .
instances: 1
memory: 96M
```

4. From the Eclipse Orion Web IDE menu, click the **Git** icon.

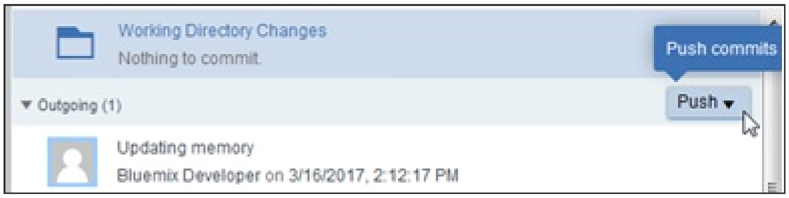


5. Enter a relevant comment and click **Commit** to push the changes in the local master branch.


Working Directory Changes
Commit the selected files with the given message.
Commit

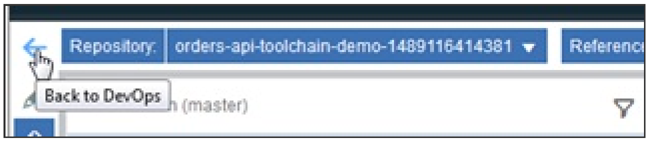
Updating memory

6. Click **Push** to put the changes into GitHub.

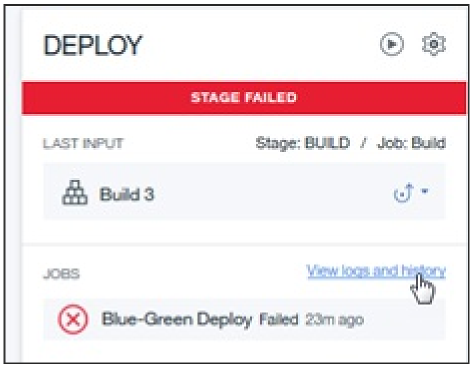


Updating memory
Bluemix Developer on 3/16/2017, 2:12:17 PM

7. Your changes are automatically built and deployed in the pipeline.
8. Click the blue arrow in the upper left hand corner to return to the Toolchain.

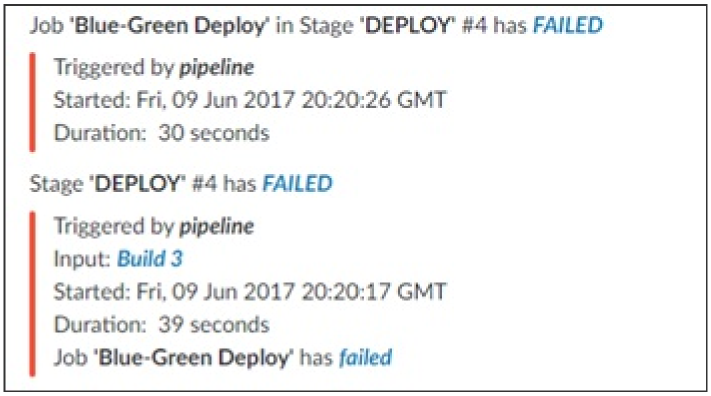


9. Click the tile for the *order-api-toolchain-labtimestamp* Delivery Pipeline. Our change caused the deployment to fail.



Blue-Green Deploy Failed 23m ago

10. Switch to the Slack browser tab, Slack also notifies the team about the Deployment failure.



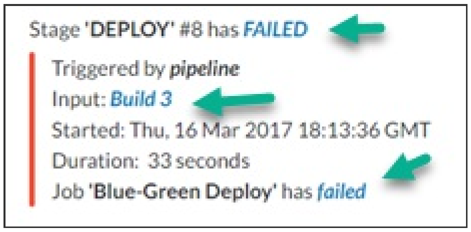
Job '**Blue-Green Deploy**' in Stage '**DEPLOY**' #4 has *FAILED*

Triggered by ***pipeline***
Started: Fri, 09 Jun 2017 20:20:26 GMT
Duration: 30 seconds

Stage '**DEPLOY**' #4 has *FAILED*

Triggered by ***pipeline***
Input: *Build 3*
Started: Fri, 09 Jun 2017 20:20:17 GMT
Duration: 39 seconds
Job '**Blue-Green Deploy**' has *failed*

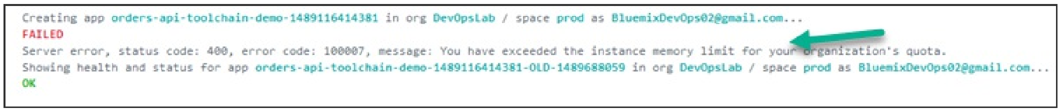
11. In Slack, there are a few links to go to the Stage, Job Input or Job. Click the **Failed** link for the Blue/Green Deploy job.



Stage '**DEPLOY**' #8 has *FAILED*

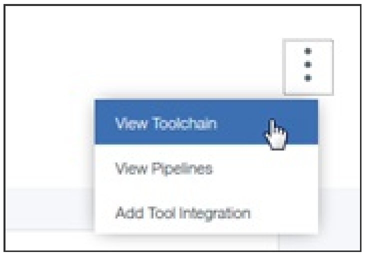
Triggered by ***pipeline***
Input: *Build 3*
Started: Thu, 16 Mar 2017 18:13:36 GMT
Duration: 33 seconds
Job '**Blue-Green Deploy**' has *failed*

12. Looking at the *Blue/Green Deploy* Job log (and scrolling down), we can see why it failed.



```
Creating app orders-api-toolchain-demo-1489116414381 in org DevOpsLab / space prod as BluemixDevOps02@gmail.com...
FAILED
Server error, status code: 400, error code: 100007, message: You have exceeded the instance memory limit for your organization's quota.
Showing health and status for app orders-api-toolchain-demo-1489116414381-OLD-1489688059 in org DevOpsLab / space prod as BluemixDevOps02@gmail.com...
OK
```

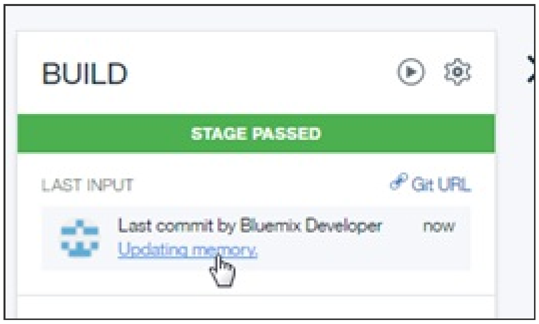
Return to the Toolchain by clicking on the vertical ellipsis on the upper right of the Bluemix console page and selecting **View Toolchain**.



Task 4: Fix application

Now to fix the application.

1. On the **simple-order-toolchain-*timestamp*** Toolchain, click the **Eclipse Orion Web IDE** tile.
2. In the file directory, click manifest.yml to open the file.
3. Update the value for memory to 96M.
4. Now to Commit and Push the changes. From the Eclipse Orion Web IDE menu, click the **Git** icon.
5. Click **Commit** to put the changes in the local master branch (enter a relevant Commit comment).
6. Click **Push**. Your changes are automatically built and deployed in the pipeline.
7. Return to Toolchain page and click the Delivery Pipeline tile to watch the stages run in response to your code changes.
8. You could also display the Slack browser tab to display the Delivery Pipeline progress.
9. The deploy is successful and all the downstream stages run afterwards.
10. The Delivery Pipeline shows the Git URL and Git commit link.



BUILD

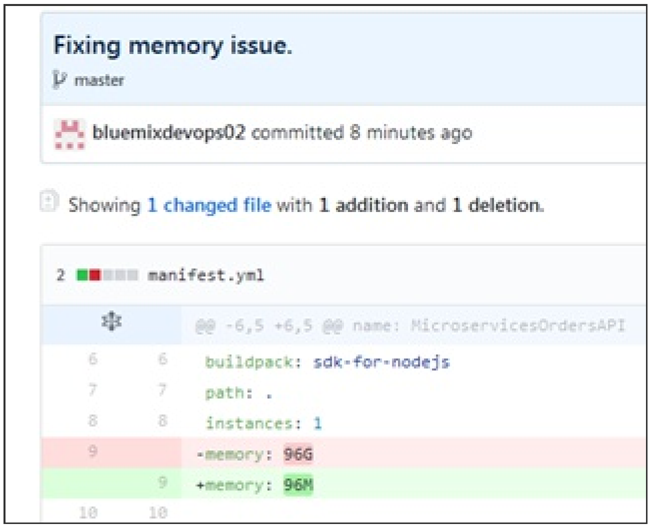
STAGE PASSED

LAST INPUT Git URL

Last commit by Bluemix Developer now

Updating memory.

11. Click the Git commit link to show the changes that were committed that caused the Delivery Pipeline to execute.



Fixing memory issue.

master

bluemixdevops02 committed 8 minutes ago

Showing 1 changed file with **1 addition** and **1 deletion.**

```
2 manifest.yml
@@ -6,5 +6,5 @@ name: MicroservicesOrdersAPI
 6   6    buildpack: sdk-for-nodejs
 7   7    path: .
 8   8    instances: 1
 9       -memory: 96G
     9   +memory: 96M
10  10
```

12. Return to the Toolchains page.