IBM MQ
And
IBM Integration Bus

Explored on

IBM Cloud Private

V3.0

Contents

Introduction

This document will explore the basics of deploying and connecting to IBM MQ and IBM Integration Bus on IBM Cloud Private (ICP).

The ICP environment I used is located on the IBM Sydney (IIC) Pure Application System (Bluemix Local) rack in St Leonards.

This instance of ICP on the PureApp is not public and VPN access to the IIC is required. However, the instructions (VPN connection aside) have been verified against ICP running on non-VPN accessible instances such as laptops.

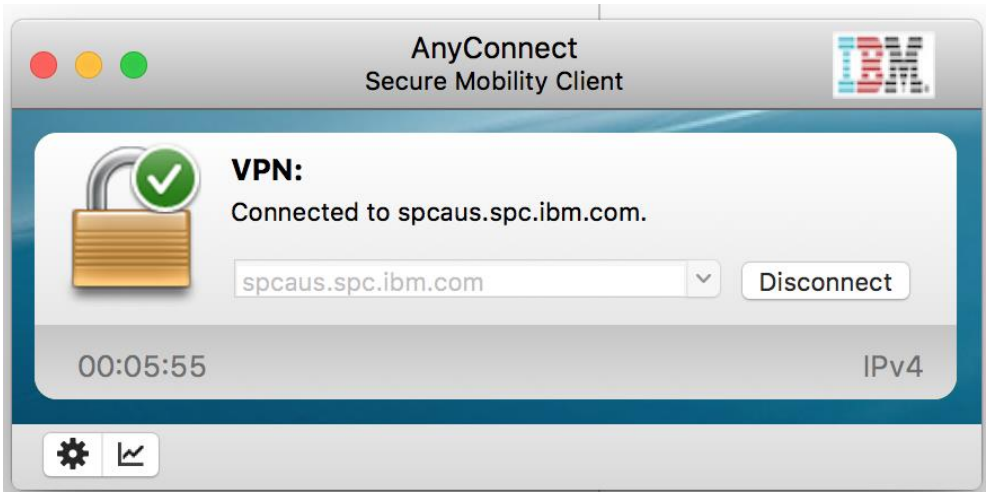
During the construction of this document the ICP instance in the Sydney IIC Pure Application system was upgraded and rebuilt. Therefore, IP address and port numbers in screen shots will have changed. For example

Details that changed	Before Upgrade/Rebuild	After Upgrade/Rebuild
ICP Console IP	https://172.23.49.17:8443/console/	https://172.23.50.112:8443/console/
ICP deployment ingress IP	172.23.49.19:*portnumber*	172.23.50.111:*portnumber*

Where possible in the text I have made these IP address changes, however I have not recaptured all the screen shots.

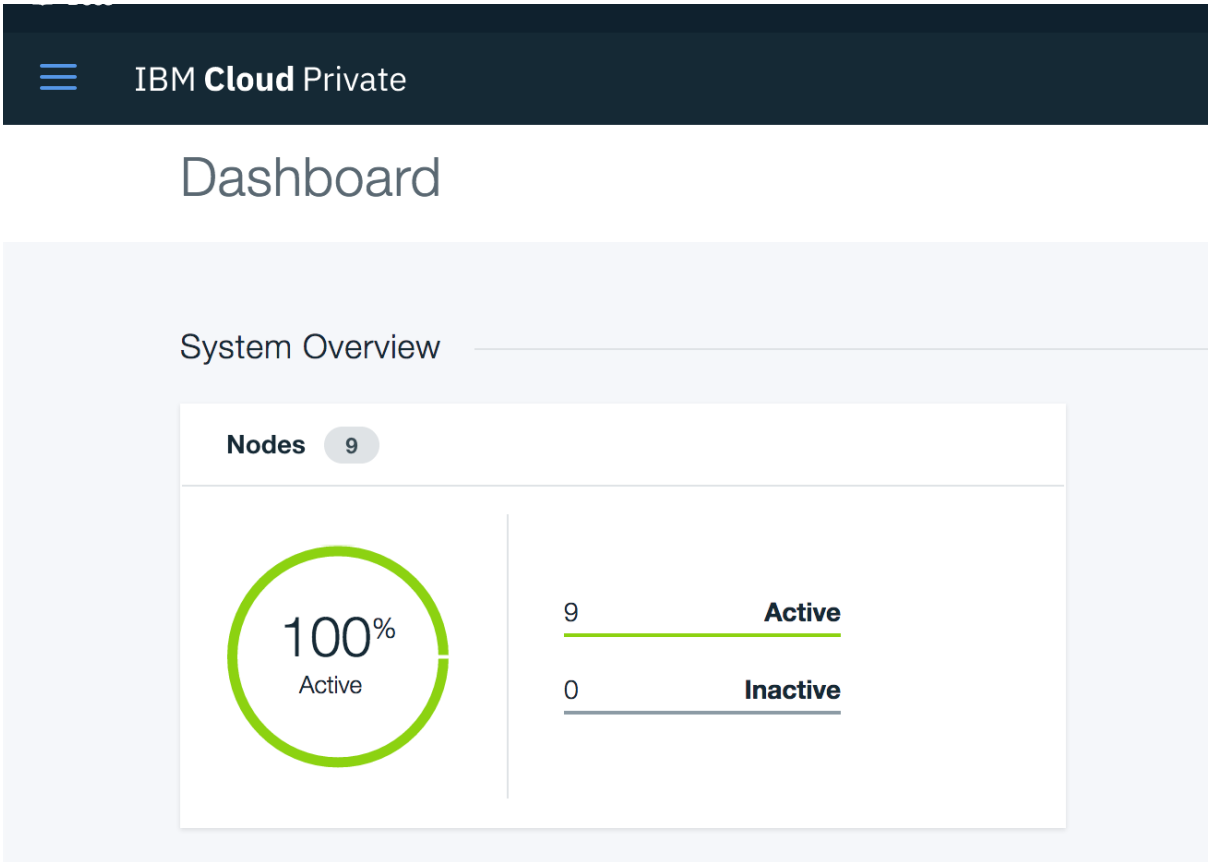
Connecting to the IBM Sydney IIC VPN

Connect to IIC with syd9185 (you will use your own ID)



Point your favourite Browser at : https://172.23.50.112:8443/console/

At the login screen admin/admin to log into ICP console



Connecting to ICP from the command line.

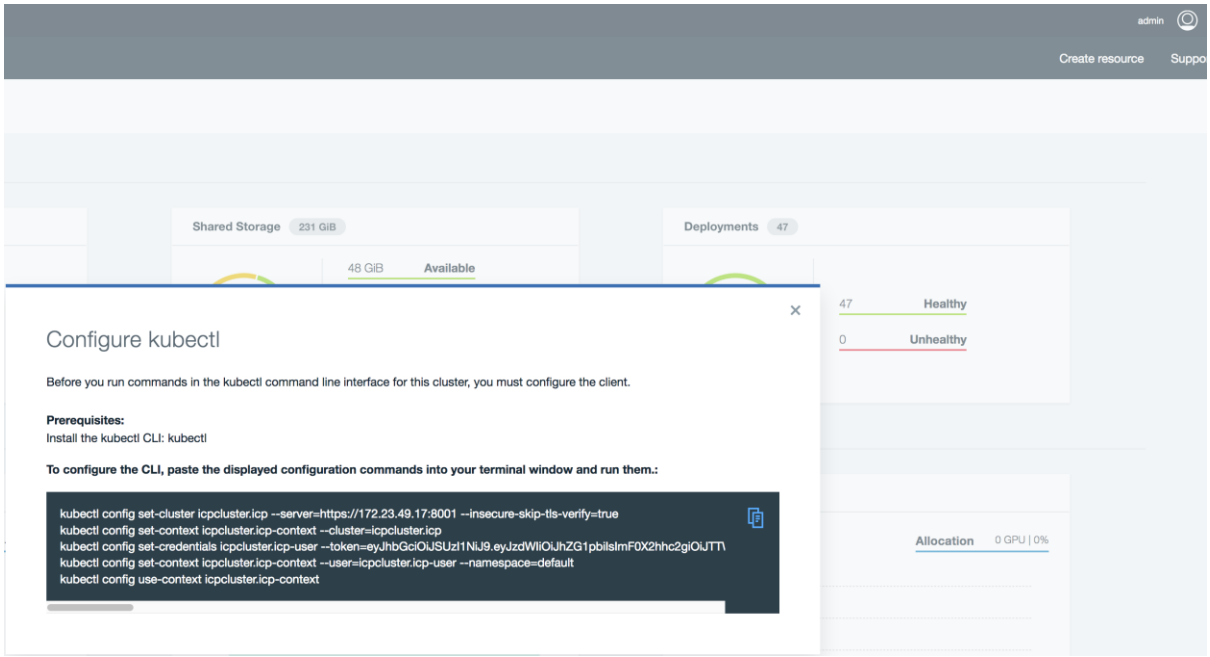
ICP Knowledge Center link

Knowledge Center can be found here if you need it
https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/manage_cluster/cfc_cli.html

This link takes you to the kubernetes CLI install and set up

Get the icp cluster details

Select User Name > Configure client, which is in the upper right of the window. The cluster configuration details display, resembles the following code:



Use the “Copied” icon and run in a terminal on your machine where kubernetes cli is installed and set up

Connect the command line environment to the ICP cluster

kubectl config set-cluster icpcluster.icp --server=https://172.23.50.112:8001 --insecure-skip-tls-verify=true

kubectl config set-context icpcluster.icp-context --cluster=icpcluster.icp

kubectl config set-credentials icpcluster.icp-user --token=eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJhZG1pbiIsImF0X2hhc2giOiJTTW1raV80Yk9xTlRTZmVXWnBoN1F3IiwiaXNzIjoiaHR0cHM6Ly9pY3BjbHVzdGVyLmljcDo5NDQzL29pZGMvZW5kcG9pbnQvT1AiLCJhdWQiOiJkMzFkZDgxMjVmNjg3ZjIxMzE1Y2ZjYTEyYTY5ZjI4MiIsImV4cCI6MTUxMTk3MTI2OSwiaWF0IjoxNTExOTI4MDY5fQ.XAOX5mFAeADXuvmENY01N0Rk46MSTjBdwPkKfpFSEoPJRjpTCsFZ1uh6rjuNfFZiZLMZO-VZti3V4725_qyzkZGCNMlsqX9V6t3lJtkTDahKDQDwiisqQ2JvY1O07qJQXxgheGih2UnR3NLQj_A5H01BLUSgts5kN4Njh_0DuD6QojsVMxh-kOS9Y352peRcpUyg5lBJdPbLHicAeNodbvVglYw-

amqnqnmIdy_v09lsTV_rl7vBfLr0RrikuzBGLOVcXXpDAGc7UIfnb2txcNco4T1qsfWbJzw2Rwr4Ta2PcdQ
gFg3hckBx6-degxmeSMYj3cjsrw1UU39bhzmpHA

kubectl config set-context icpcluster.icp-context --user=icpcluster.icp-user --namespace=default

kubectl config use-context icpcluster.icp-context

Get the Kubernetes Pod details – see what's running

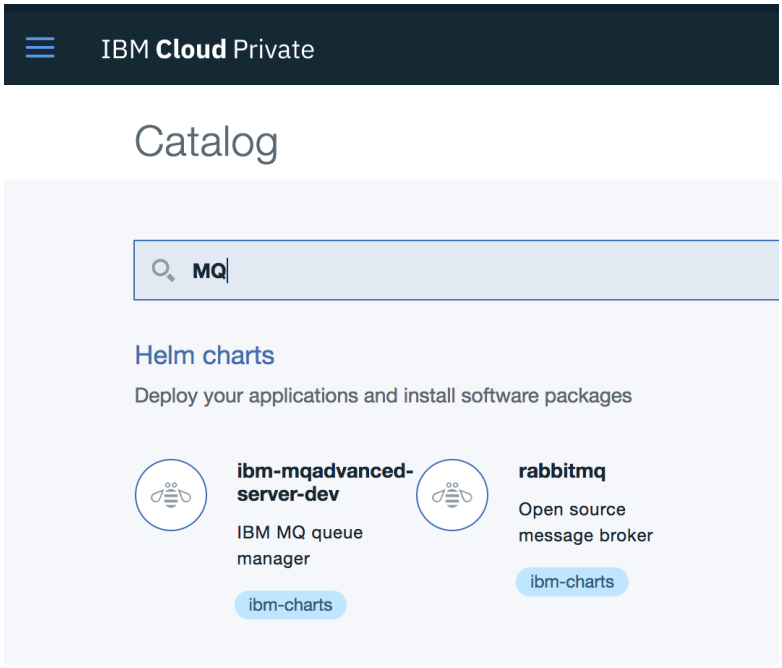
Daves-MacBook-Pro-2:deploy-ibm-cloud-private-master davearno$ kubectl get pods

```
NAME                                                              READY     STATUS    RESTARTS   AGE
blockchain-ca-2437173632-sfhlw                                    1/1       Running   0          18h
blockchain-orderer-1019217726-jc9vs                               1/1       Running   0          18h
blockchain-org1peer1-3689184489-9jcj8                             1/1       Running   0          18h
blockchain-org2peer1-2132069490-bshv0                             1/1       Running   0          18h
composer-playground-943199952-4tb02                               1/1       Running   4          18h
composer-rest-server-1507354567-m993w                             1/1       Running   0          18h
db2server1-ibm-db2oltp-dev-2982159975-hxq2s                       1/1       Running   0          1d
ibm-msb-pipeline-ibm-microservicebuilder-pipeline-146566793qkl5   1/1       Running   0          15h
ibm-msbf-zipkin-3680589052-cgfjw                                  1/1       Running   0          16h
icpmsbdemo-deployment-3190653047-r9c8r                            1/1       Running   0          14h
javacloudtrader4db2-3324903285-9v2lm                              1/1       Running   1          1d
```

IBM MQ on ICP via the ICP Console

Deploying MQ via Helm

Go to the Catalog (filter on MQ)



Select ibm-mqadvanced-server-dev and the following page details how to install the helm chart from the command line (listing all the parameters further down the page)

OR

Use the Configure button

ibm-mqadvanced-server-dev V 1.0.2

Helm Chart Configuration

From the Configure button

Tick the accept license

Set the release name : mq-da-icpqm1 (don't use any capitals or a number for 1st character)

Set Persistence to false

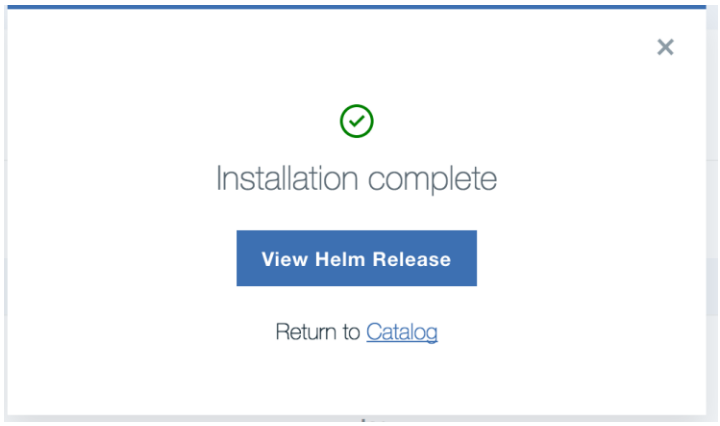
Service ->Type: NodePort (not clusterIP)

Set a queue manager name : icpQM1

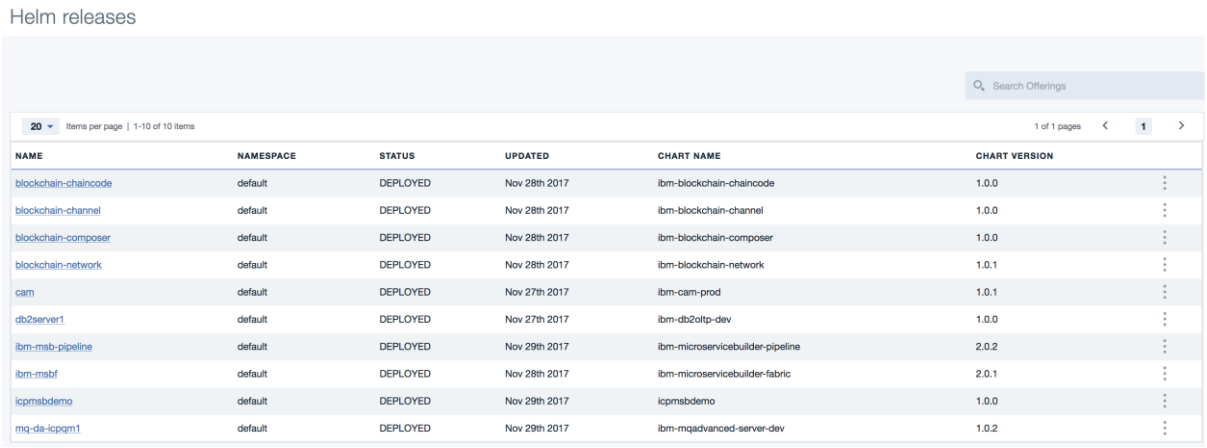
Admin.password : passw0rd (you must set a password)

Hit install

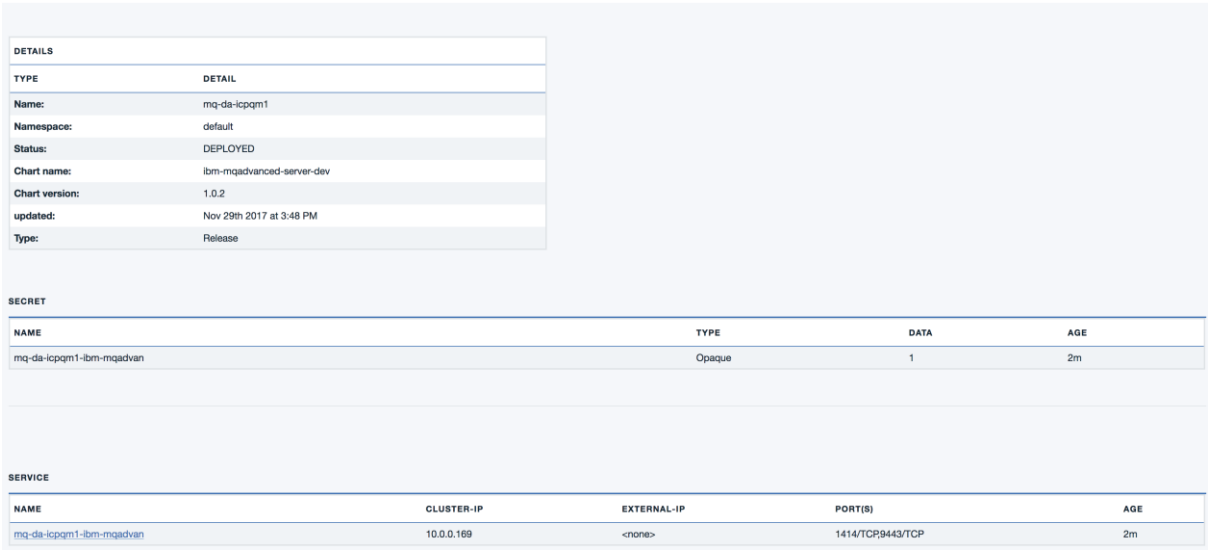
View the Helm Release



Click on View Helm Release

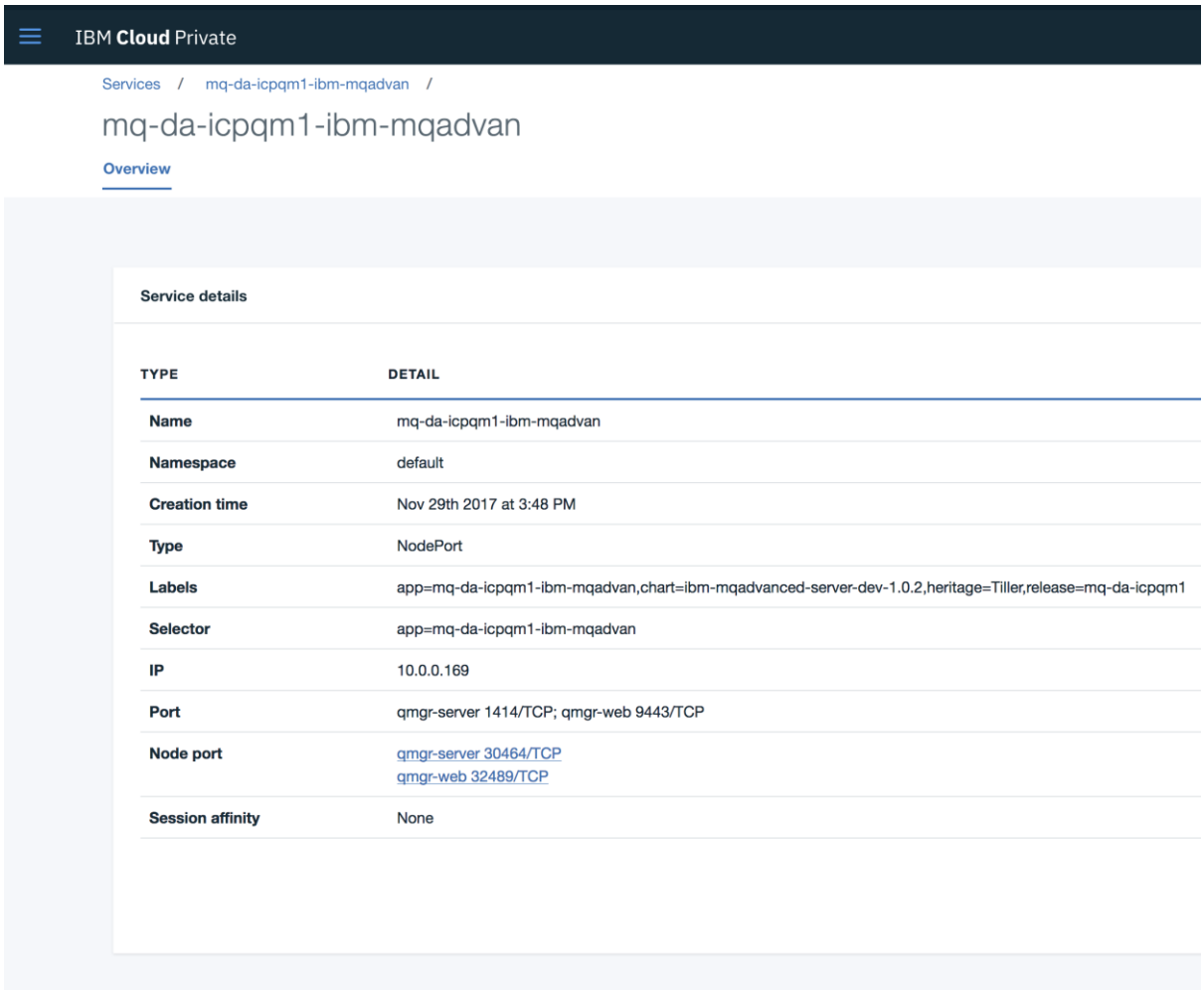


mq-da-icpqm1



Now wait for the service to spin up

Review the MQ Helm Release details in the ICP console

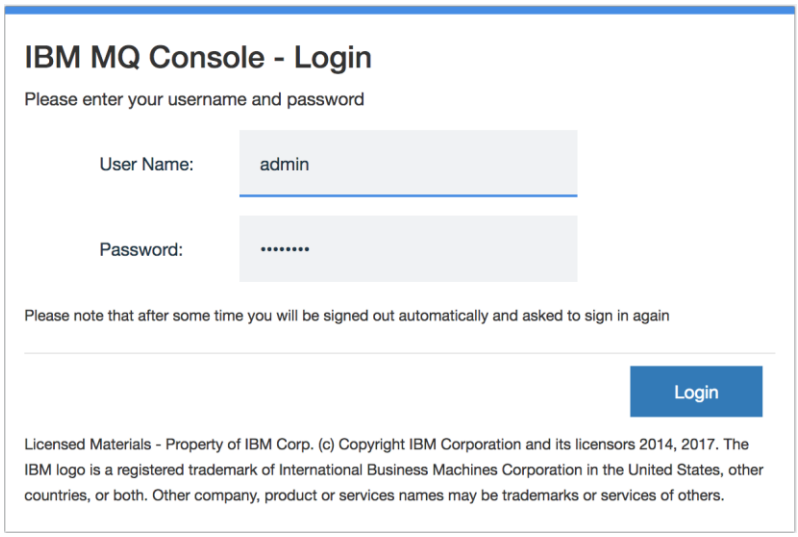


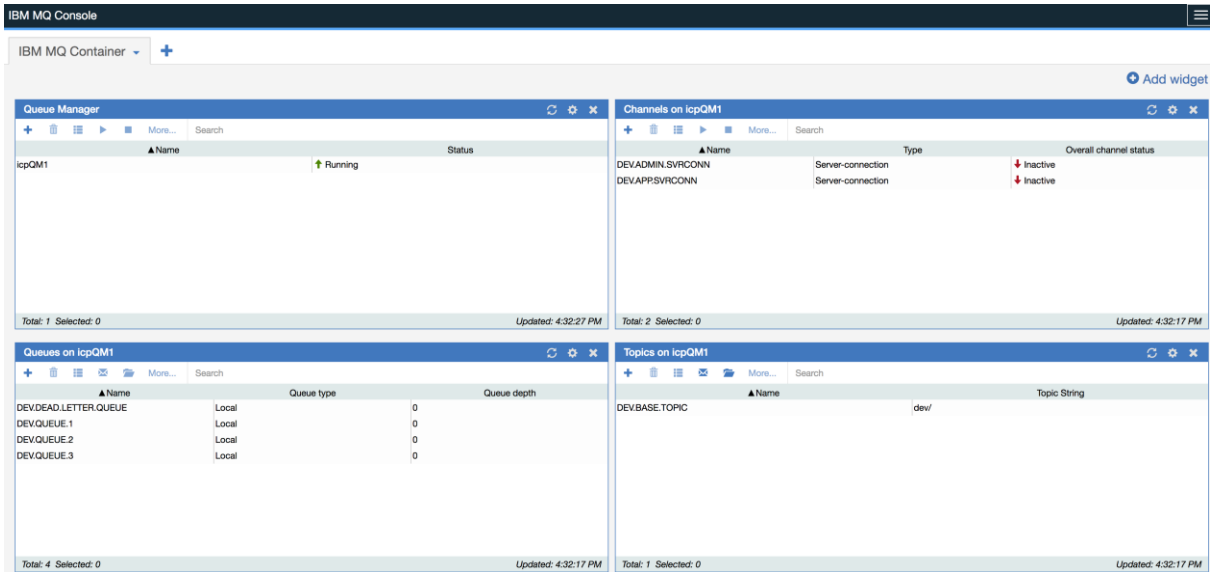
Connect the IBM MQ Web Console

Hover over the Node port links and the IP address will appear bottom left.

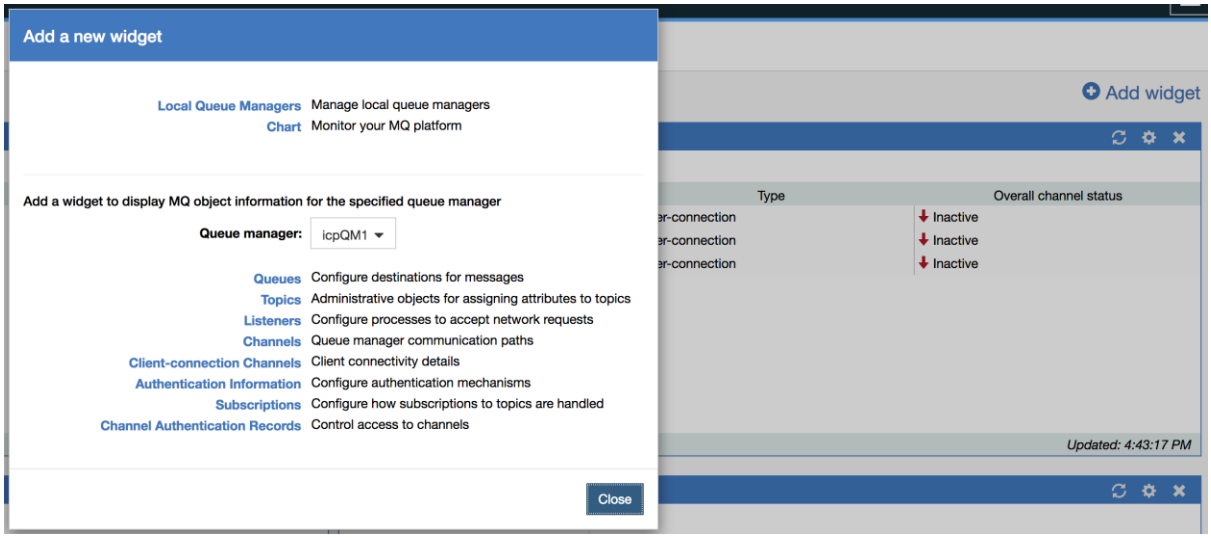
Click on qmgr-server 30464/TCP link and a new browser tab will open (you have to add https://)

Default username is **admin** and the password as the one set at configuration time (passw0rd)





Add additional Qmgr information panes (add widget) – listeners is a good one



Running command line commands in the MQ Container

Bash into a container

Issue a kubectl get pods command

```
Daves-MacBook-Pro-2:deploy-ibm-cloud-private-master davearno$ kubectl get pods
NAME                                          READY     STATUS    RESTARTS   AGE
blockchain-ca-2437173632-sfhlw                1/1       Running   0          19h
blockchain-orderer-1019217726-jc9vs           1/1       Running   0          19h
blockchain-org1peer1-3689184489-9jcj8         1/1       Running   0          19h
blockchain-org2peer1-2132069490-bshv0         1/1       Running   0          19h
composer-playground-943199952-4tb02           1/1       Running   4          19h
composer-rest-server-1507354567-m993w         1/1       Running   0          19h
db2server1-ibm-db2oltp-dev-2982159975-hxq2s   1/1       Running   0          2d
```

ibm-msb-pipeline-ibm-microservicebuilder-pipeline-146566793qkl5 1/1 Running 0 16h
ibm-msbf-zipkin-3680589052-cgfjw 1/1 Running 0 17h
icpmsbdemo-deployment-3190653047-r9c8r 1/1 Running 0 15h
javacloudtrader4db2-3324903285-9v2lm 1/1 Running 1 1d
mq-da-icpqm1-ibm-mqadvan-0 1/1 Running 0 9m

Issue kubectl exec -it mq-da-icpqm1-ibm-mqadvan-0 bash

Using RUNMQSC to configure the queue manager

Daves-MacBook-Pro-2:deploy-ibm-cloud-private-master davearno$ kubectl exec -it mq-da-icpqm1-ibm-mqadvan-0 bash
(mq:9.0.4.0)root@mq-da-icpqm1-ibm-mqadvan-0:/# runmqsc icpQM1
5724-H72 (C) Copyright IBM Corp. 1994, 2017.
Starting MQSC for queue manager icpQM1.

dis ql(*)
1 : dis ql(*)
AMQ8409I: Display Queue details.
QUEUE(DEV.DEAD.LETTER.QUEUE) TYPE(QLOCAL)

Connecting the MQ Explorer

Having bashed into your container - create a user with a password that is in the MQ group I used davearno

Got the instructions from here
https://github.com/ibm-messaging/mq-docker

Add a new user name in the mqm group

```
useradd davearno -G mqm && \
    echo davearno:passw0rd | chpasswd
```

(mq:9.0.3.0)root@mq-rel-1-ibm-mqadvanced-0:/# useradd davearno -G mqm && \

Use RUNMQSC to set up channel authentication

(mq:9.0.3.0)root@mq-rel-1-ibm-mqadvanced-0:/# runmqsc icpQM1
5724-H72 (C) Copyright IBM Corp. 1994, 2017.
Starting MQSC for queue manager icpQM1.

DEFINE CHANNEL(PASSWORD.SVRCONN) CHLTYPE(SVRCONN) REPLACE
1 : DEFINE CHANNEL(PASSWORD.SVRCONN) CHLTYPE(SVRCONN) REPLACE
AMQ8014: IBM MQ channel created.
SET CHLAUTH(PASSWORD.SVRCONN) TYPE(BLOCKUSER) USERLIST('nobody')
DESCR('Allow privileged users on this channel')

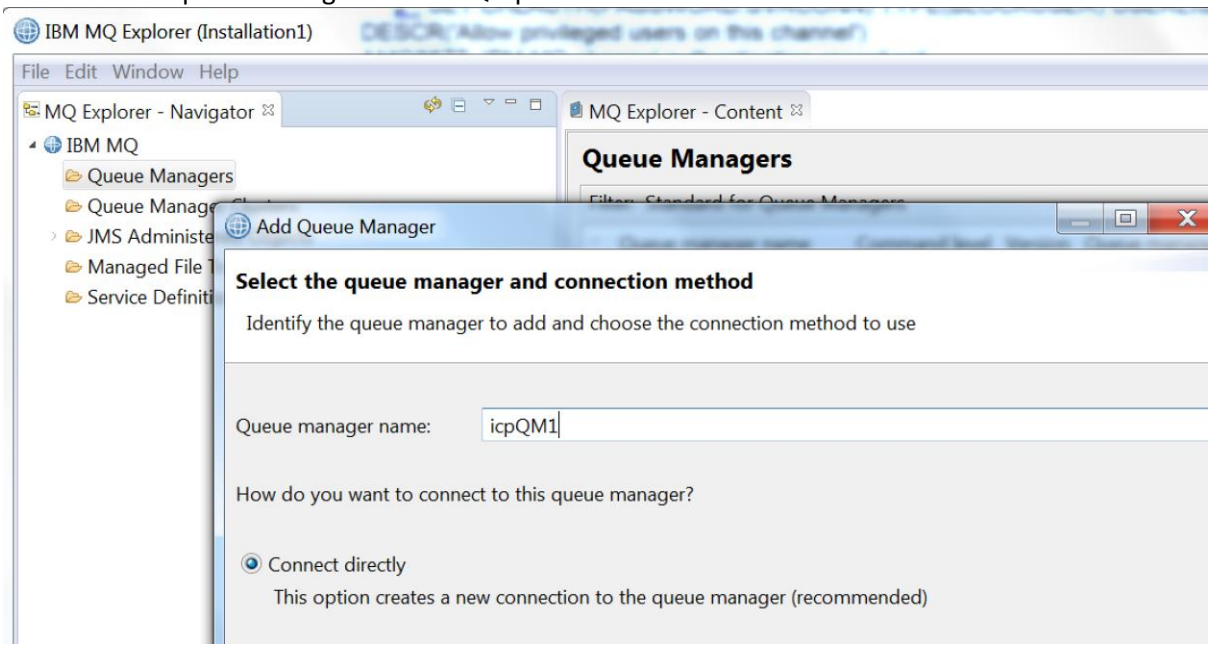
2 : SET CHLAUTH(PASSWORD.SVRCONN) TYPE(BLOCKUSER) USERLIST('nobody') DESCR('Allow privileged users on this channel')
AMQ8877: IBM MQ channel authentication record set.
SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) DESCR('BackStop rule')
3 : SET CHLAUTH('*') TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(NOACCESS) DESCR('BackStop rule')
AMQ8883: Channel authentication record already exists.
SET CHLAUTH(PASSWORD.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) CHCKCLNT(REQUIRED)
4 : SET CHLAUTH(PASSWORD.SVRCONN) TYPE(ADDRESSMAP) ADDRESS('*') USERSRC(CHANNEL) CHCKCLNT(REQUIRED)
AMQ8877: IBM MQ channel authentication record set.
ALTER AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPWOS) AUTHTYPE(IDPWOS) ADOPTCTX(YES)
5 : ALTER AUTHINFO(SYSTEM.DEFAULT.AUTHINFO.IDPWOS) AUTHTYPE(IDPWOS) ADOPTCTX(YES)
AMQ8567: IBM MQ authentication information changed.
REFRESH SECURITY TYPE(CONNAUTH)
6 : REFRESH SECURITY TYPE(CONNAUTH)

end

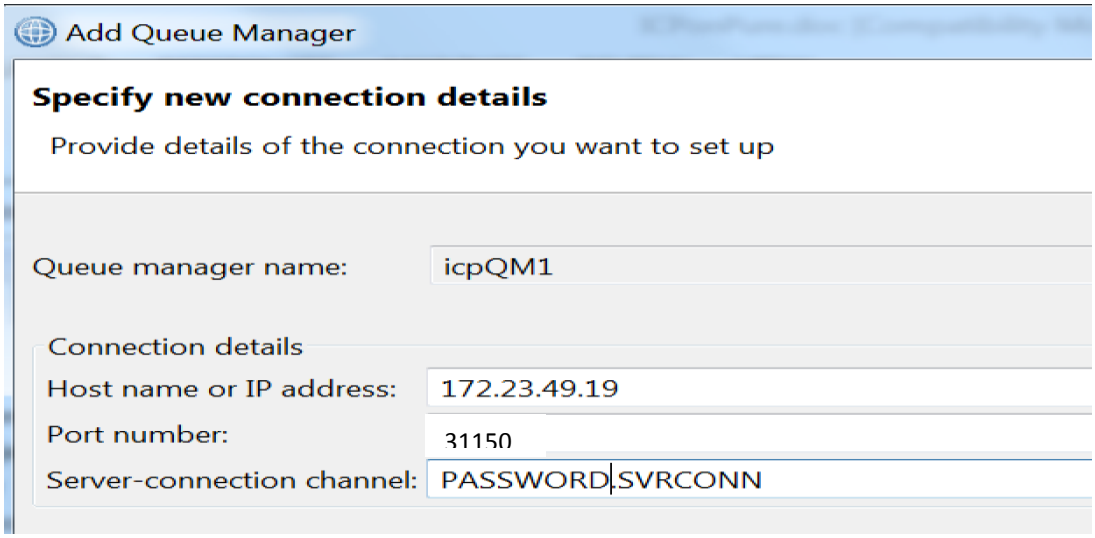
Connect from MQ Explorer

From your MQ explorer - add remote queue manager connect using your PASSWORD.SVRCONN and enable user identification with a password

Add a remote queue manager in the MQExplorer

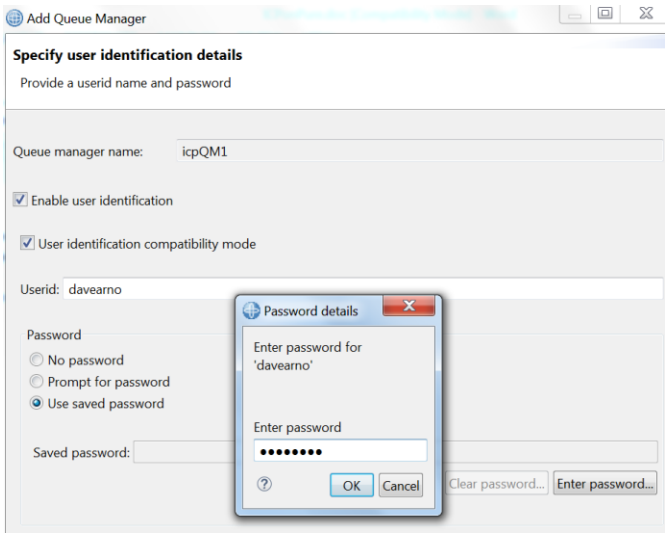


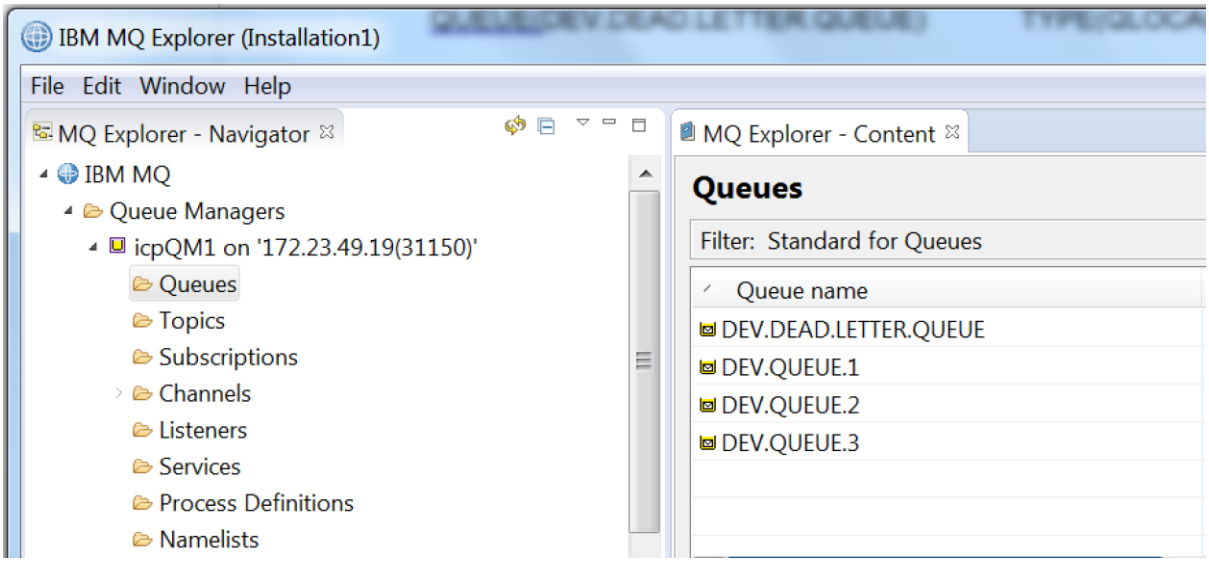
Next specify the host name (port is 31150 not 1414) and the PASSWORD.SVRCONN



Next leverage the username/password you set up using the following commands:

```
useradd davearno -G mqm && \
    echo davearno:passw0rd | chpasswd
```





Some points of note

The container is running in the following loop. If you endmqm your icpQM1 the container will stop and be restarted in it's initial state i.e. your userid to mqm group and icpQM1 new SVRCONN channel will be gone.

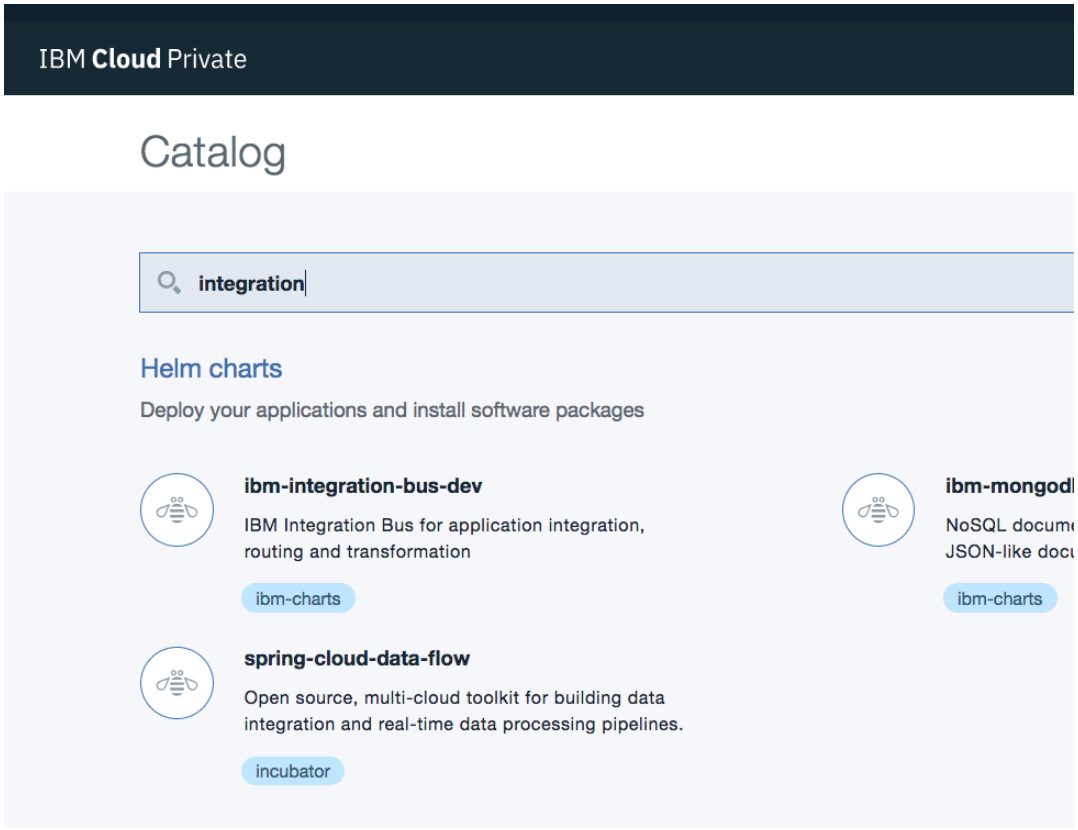
```
# Loop until "dspmq" says the queue manager is not running any more
  until [ "`state`" != "RUNNING" ]; do
    sleep 5
  done
```

You can however create a second queue manager once "bashed in" and start and stop that as much as you like.

IBM Integration Bus on ICP via the ICP Console

Installing IIB by Helm Chart

Go to the catalog and filter on integration

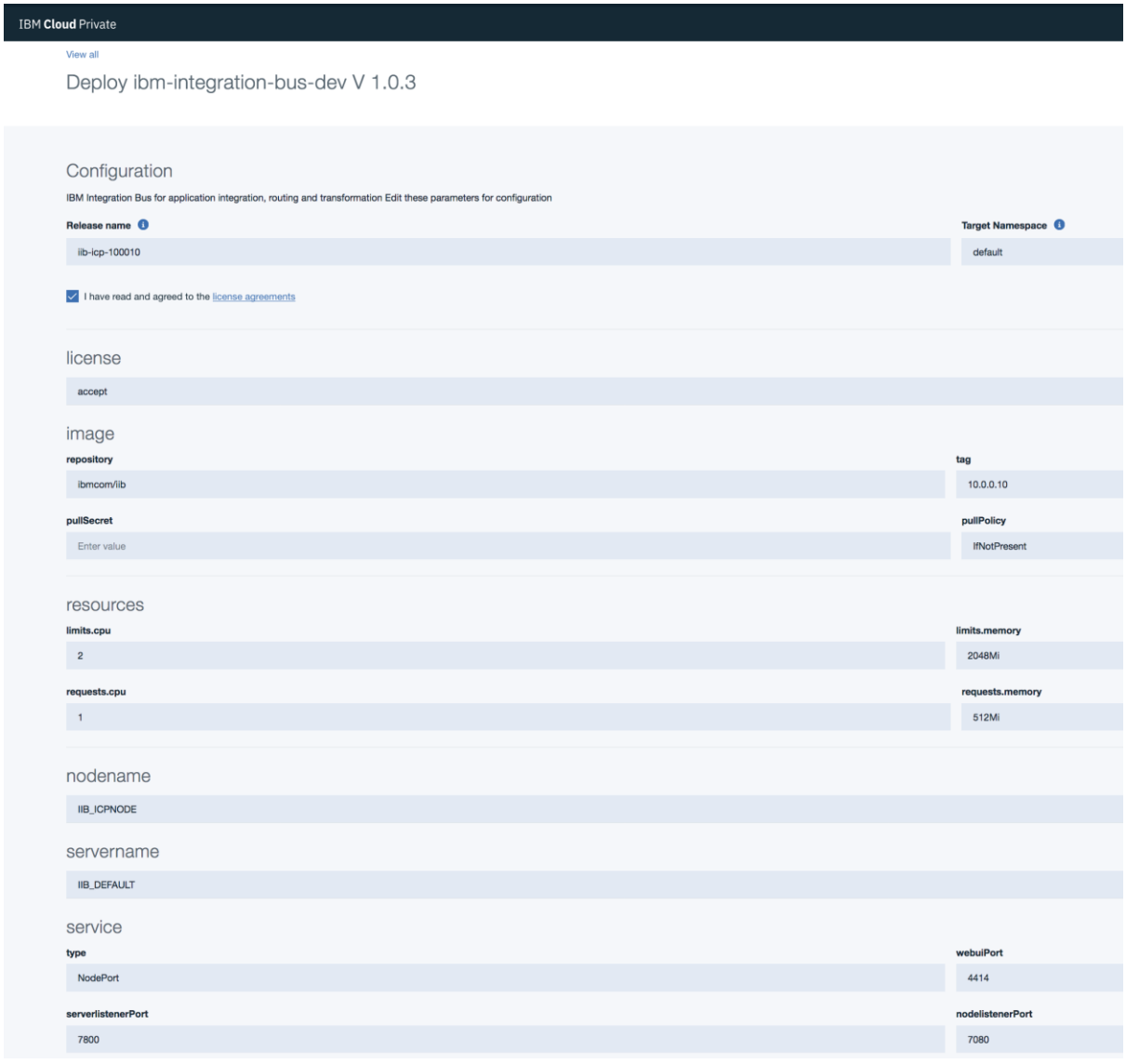


Click on ibm-integration-bus-dev

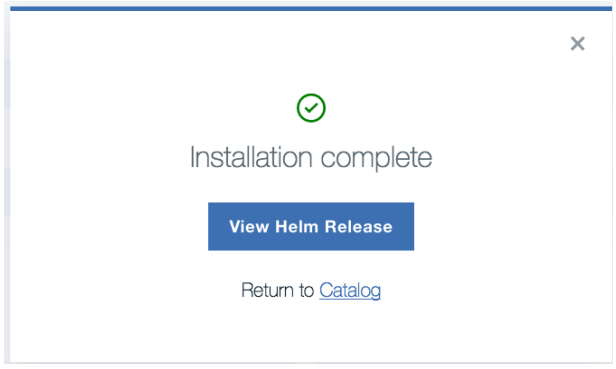
Configure the Helm Chart template

Fill in the template configuration values

Be careful to use lower case characters, avoid numerics up front and only dashes no dots.

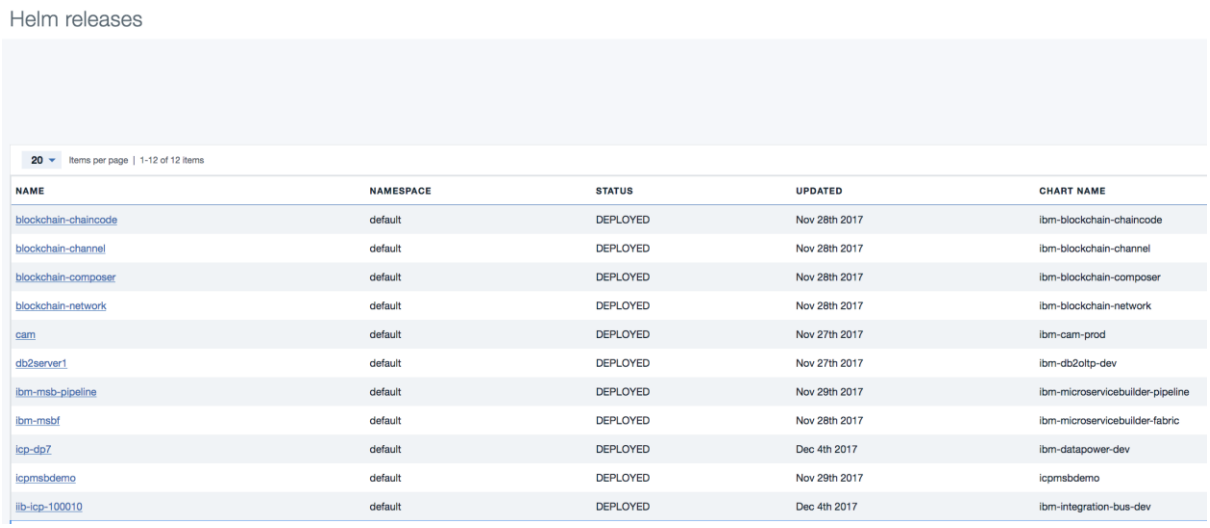


Click Install



Check the IIB Helm Release

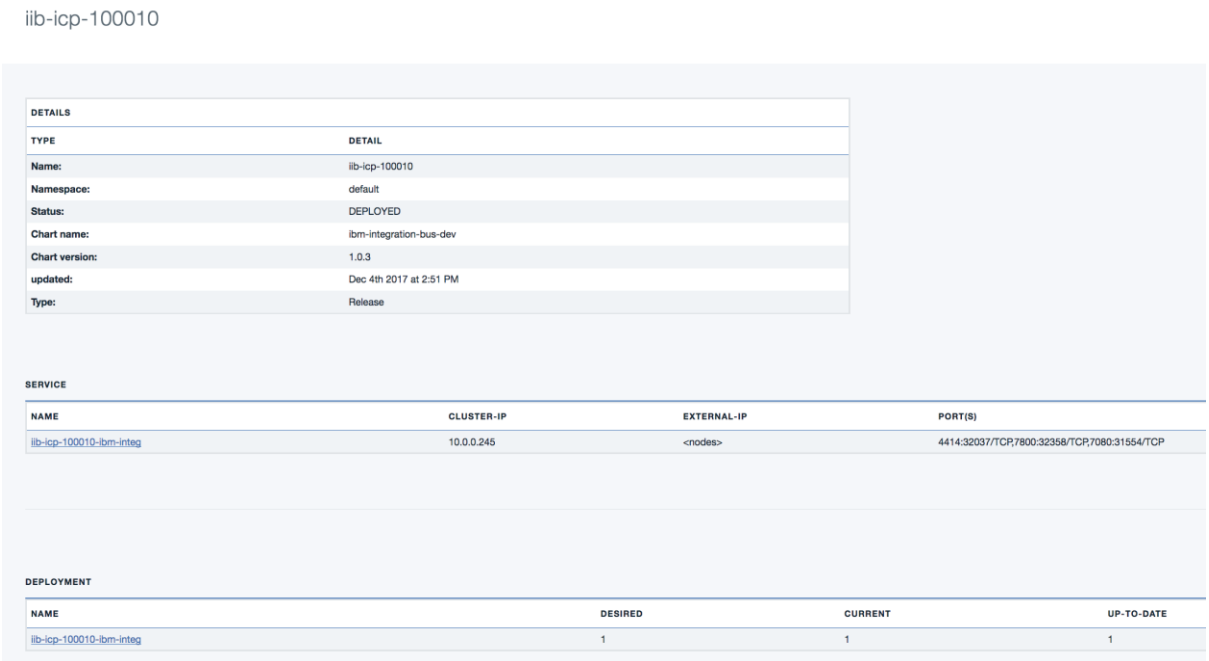
Check the Workloads->Helm releases



NAME	NAMESPACE	STATUS	UPDATED	CHART NAME
blockchain-chaincode	default	DEPLOYED	Nov 28th 2017	ibm-blockchain-chaincode
blockchain-channel	default	DEPLOYED	Nov 28th 2017	ibm-blockchain-channel
blockchain-composer	default	DEPLOYED	Nov 28th 2017	ibm-blockchain-composer
blockchain-network	default	DEPLOYED	Nov 28th 2017	ibm-blockchain-network
cam	default	DEPLOYED	Nov 27th 2017	ibm-cam-prod
db2server1	default	DEPLOYED	Nov 27th 2017	ibm-db2oltp-dev
ibm-msb-pipeline	default	DEPLOYED	Nov 29th 2017	ibm-microservicebuilder-pipeline
ibm-msbf	default	DEPLOYED	Nov 28th 2017	ibm-microservicebuilder-fabric
icp-dp7	default	DEPLOYED	Dec 4th 2017	ibm-datapower-dev
icpmsbdemo	default	DEPLOYED	Nov 29th 2017	icpmsbdemo
iib-icp-100010	default	DEPLOYED	Dec 4th 2017	ibm-integration-bus-dev

Capture the IIB IP address and Ports

Select your iib release



Click on service

Services / iib-icp-100010-ibm-integ /

iib-icp-100010-ibm-integ

Overview

Service details

TYPE	DETAIL
Name	iib-icp-100010-ibm-integ
Namespace	default
Creation time	Dec 4th 2017 at 2:51 PM
Type	NodePort
Labels	app=iib-icp-100010-ibm-integ,chart=ibm-integration-bus-dev-1.0.3,heritage=Tiller,release=iib-icp-100010
Selector	app=iib-icp-100010-ibm-integ
IP	10.0.0.245
Port	webui 4414/TCP; serverlistener 7800/TCP; nodelistener 7080/TCP
Node port	webui 32037/TCP serverlistener 32358/TCP nodelistener 31554/TCP
Session affinity	None

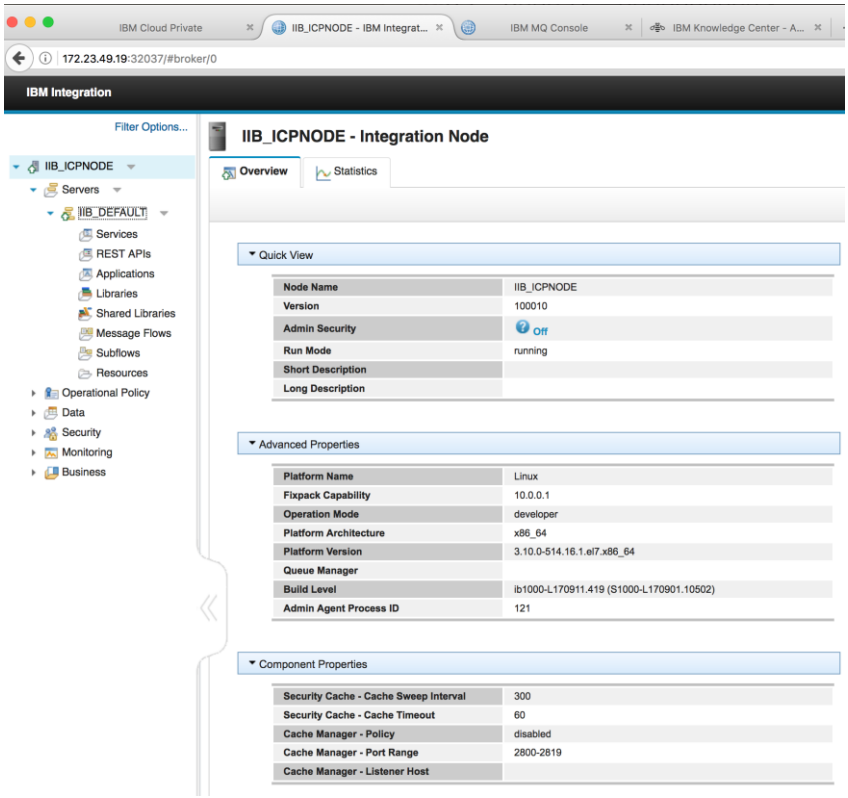
Observe the Node ports

webui 4414/TCP; serverlistener 7800/TCP; nodelistener 7080/TCP

webui 32037/TCP
serverlistener 32358/TCP
nodelistener 31554/TCP

Connect the Web UI

Click on the webUI link or copy the IP address and port to a browser. It may take a couple of minutes for ICP to deploy and run up IIB.



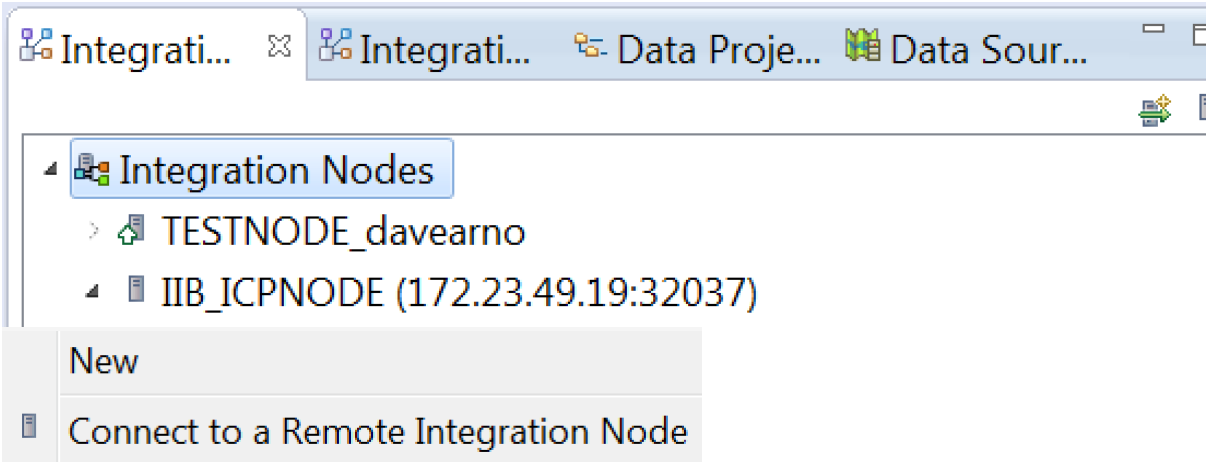
Connect the IIB Toolkit to the runtime node

webui 4414/TCP; serverlistener 7800/TCP; nodelistener 7080/TCP

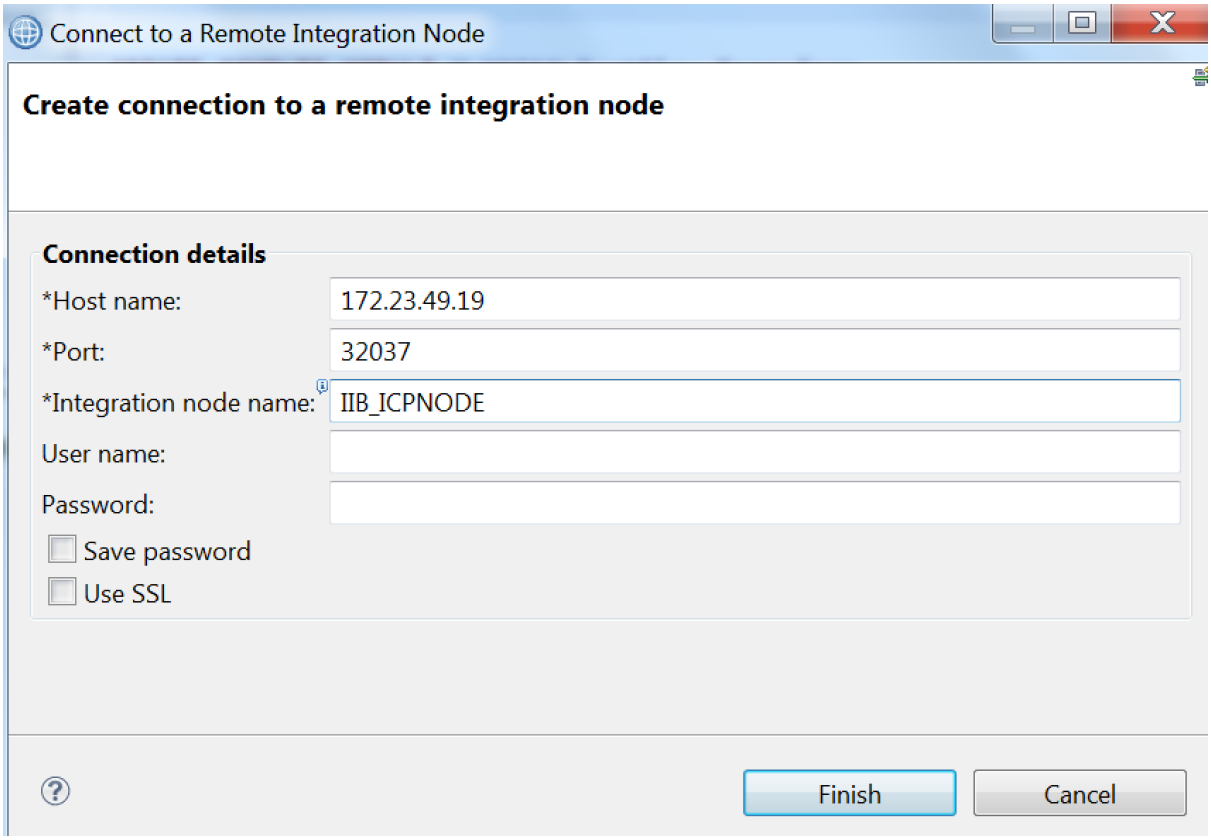
webui 32037/TCP
serverlistener 32358/TCP
nodelistener 31554/TCP

The IIB Toolkit will use the webUI ip address and port

Ctrl-select Integration Nodes and select Connect to a Remote Integration Node

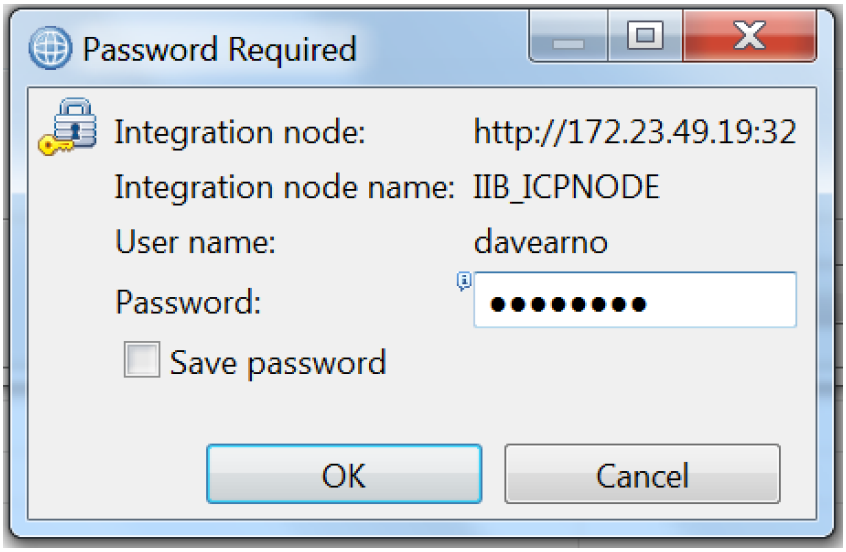


Fill in the details



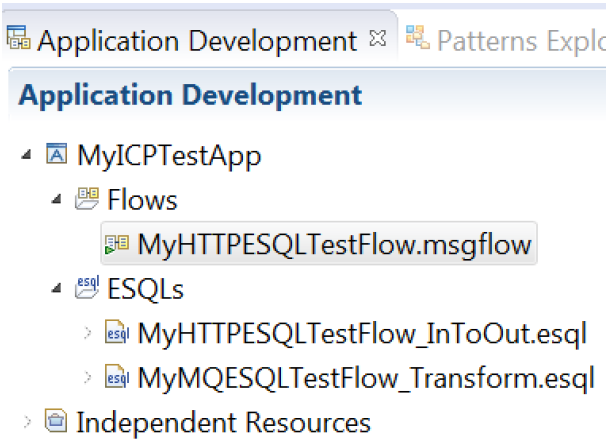
A userid/password dialog may well be opened.

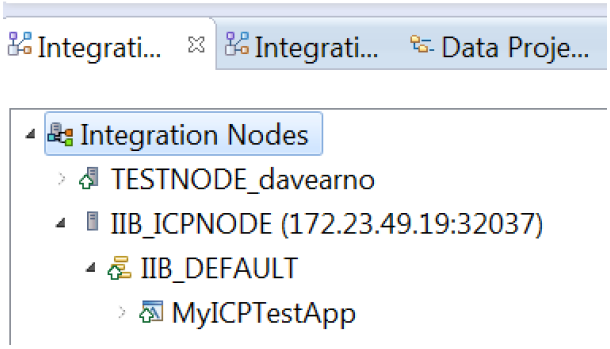
I used password: passw0rd. Not sure where davearno is being picked up from. It is a username I created in the MQ ICP container but that shouldn't be relevent! Davearno is the username of my Windows image that the toolkit it running in – but passw0rd is not the password.

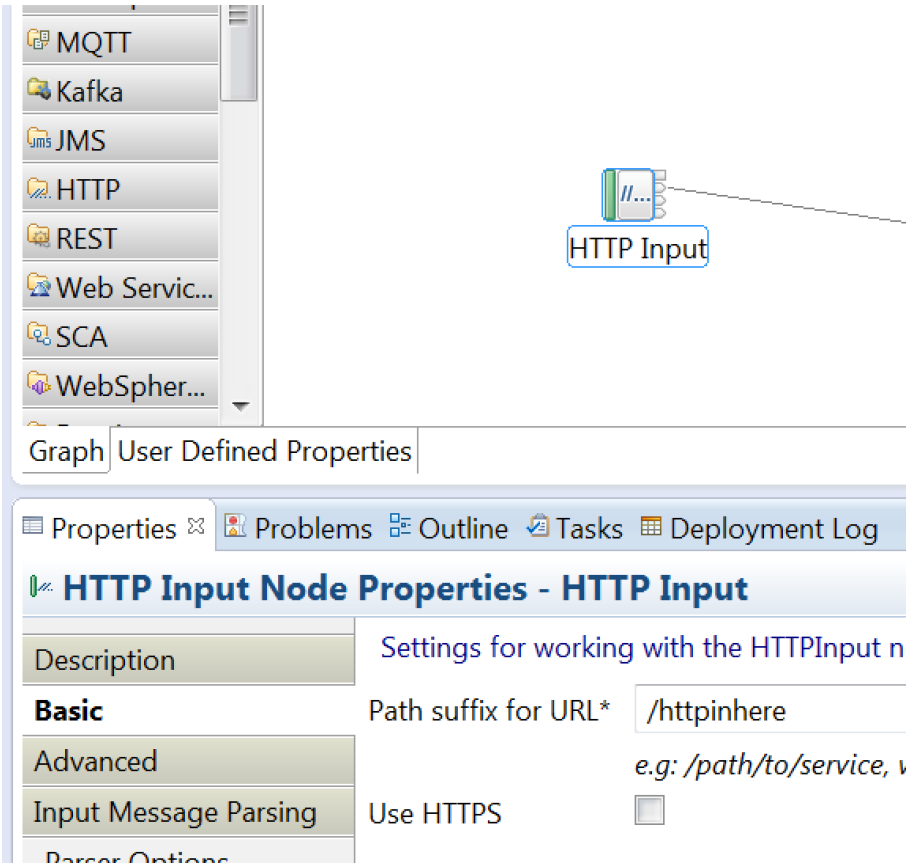


Deploying a BAR file

Assuming you have a BAR file ready go. Deploy a BAR file to IIB_DEFAULT integration server







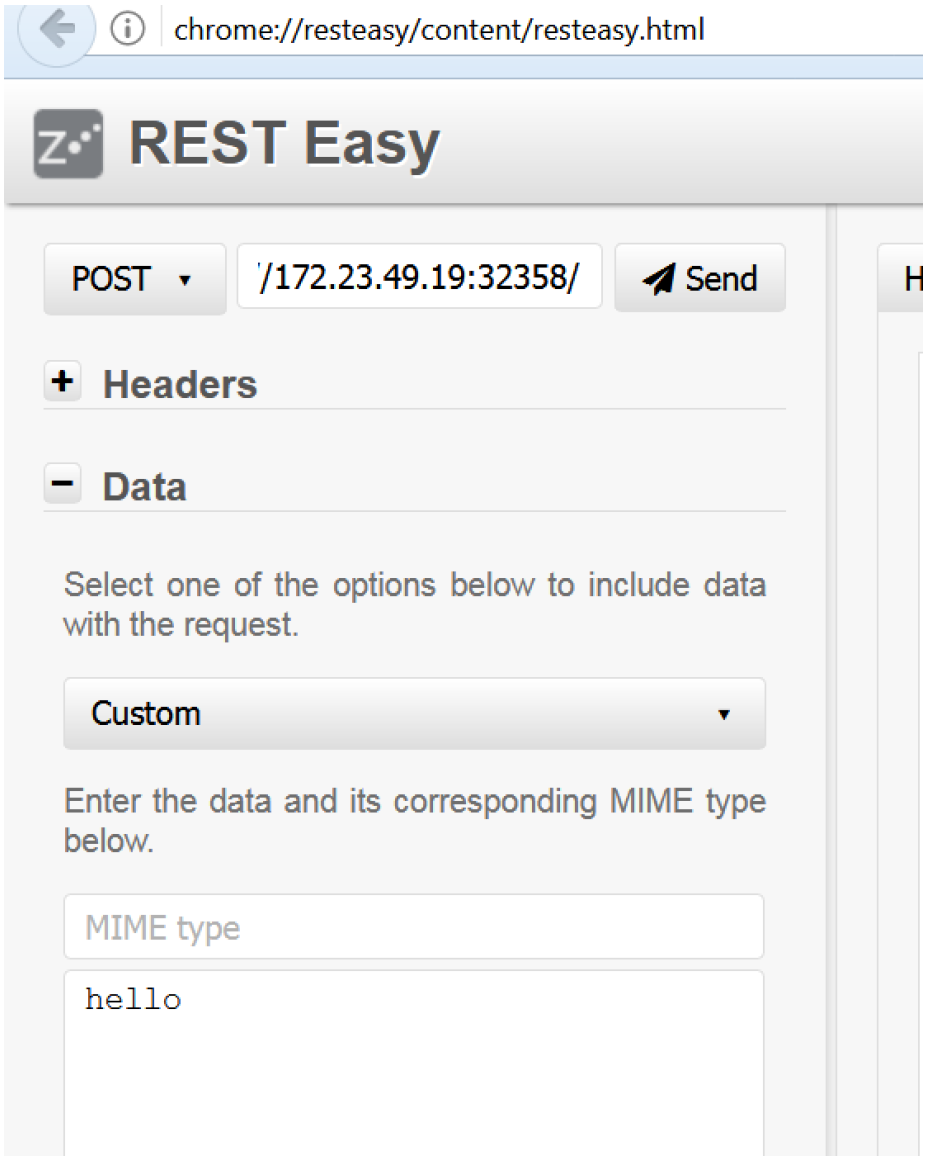
Testing an IIB message flow

webui 4414/TCP; serverlistener 7800/TCP; nodelistener 7080/TCP

webui 32037/TCP
serverlistener 32358/TCP
nodelistener 31554/TCP

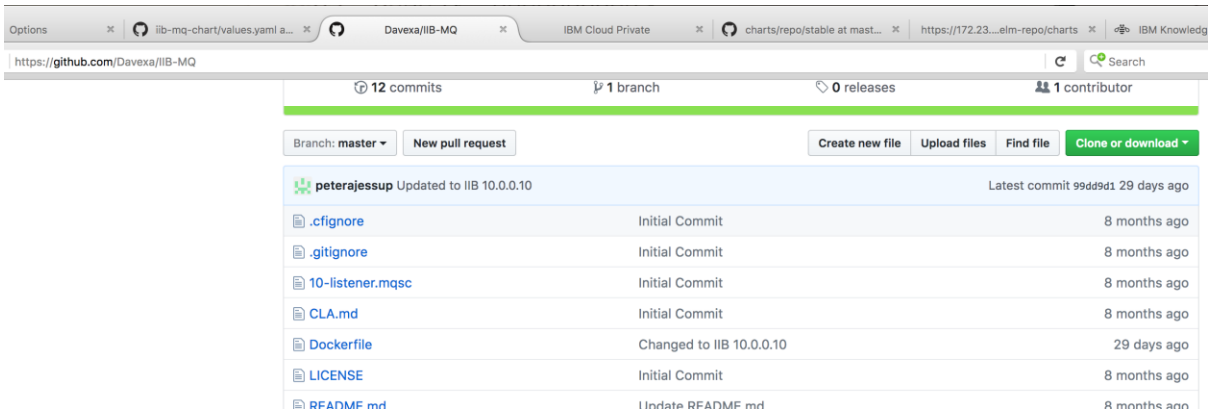
Test using a tool like Postman or Resteasy

http://172.23.50.111:32358/httpinhere do a POST and use the serverlistener port


chrome://resteasy/content/resteasy.html
REST Easy
POST
/172.23.49.19:32358/
Send
Headers
Data
Select one of the options below to include data with the request.
Custom
Enter the data and its corresponding MIME type below.
MIME type
hello

IIB & MQ Container on ICP from Github repository by hand

MQ and IIB Docker repository on github



Git Clone the IIB-MQ repository to a local repository

In this example I am ssh into a VM image on the Pure App that has all the icp/docker/helm clients installed. If you have all these tools you will not need to do this.

Git clone http://github.com.Davexa/IIB-MQ.git

```
...-cloud-private-master — (mq:9.0.4.0)root@mq-da-icpqm1-ibm-mqadvan-0: / — -bash   ~/Downloads/deploy-ibm-cloud-private-master — ssh root@172.23.52.143
Last login: Wed Nov 29 14:49:00 on ttys001
Daves-MacBook-Pro-2:deploy-ibm-cloud-private-master davearno$ ssh root@172.23.52.143
The authenticity of host '172.23.52.143 (172.23.52.143)' can't be established.
ECDSA key fingerprint is SHA256:UclkfeYujg+YwpVnGhh/NoFF3A1qDkZdk9lhyeBfgY4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.23.52.143' (ECDSA) to the list of known hosts.
root@172.23.52.143's password:
Last login: Mon Dec 11 13:35:56 2017 from 172.23.1.171
[-bash-4.2# ls
anaconda-ks.cfg  dockerImages  git  ibm-cloud-auto-mgr-x86_64-2.1.0.tar.gz  ldap  push.sh
[-bash-4.2# cd git/
[-bash-4.2# ls
cfc-crawler  container-service-getting-started-wt  repos
[-bash-4.2# git clone https://github.com/Davexa/IIB-MQ.git
Cloning into 'IIB-MQ'...
remote: Counting objects: 60, done.
remote: Compressing objects: 100% (34/34), done.
remote: Total 60 (delta 24), reused 60 (delta 24), pack-reused 0
Unpacking objects: 100% (60/60), done.
[-bash-4.2# ls
cfc-crawler  container-service-getting-started-wt  IIB-MQ  repos
[-bash-4.2# cd IIB-MQ/
[-bash-4.2# ls
10-listener.mqsc  CLA.md      iib-license-check.sh  kernel_settings.sh  mq-dev-config         mq.sh          README.md          switch.json
admin.json        Dockerfile  iib_manage.sh         LICENSE             mq-dev-config.sh      odbc.ini       setup-mqm-web.sh
agentx.json       iib_env.sh  iib-mq.yml            login.defs          mq-license-check.sh   odbcinst.ini   setup-var-mqm.sh
[-bash-4.2# vi Dockerfile
[-bash-4.2# docker build -t icpcluster.icp:8500/default/iib-mq:latest .
```

Build the IIB-MQ image

Docker build –t icpcluster.icp:8500/default/ii-mq:latest .

```
-bash-4.2# docker build -t icpcluster.icp:8500/default/iib-mq:latest .
Sending build context to Docker daemon 1.136 MB
Step 1/30 : FROM ubuntu:14.04
14.04: Pulling from library/ubuntu
01a4f8387457: Pull complete
c887940e680c: Pull complete
5432573ac160: Pull complete
027ee9a9665e: Pull complete
5611db80430d: Pull complete
Digest: sha256:7b3b72e6a388c24c0cc3e0d1aade3364c52f03e54bd5ab440f8fd93c69203733
Status: Downloaded newer image for ubuntu:14.04
 ---> d6ed29ffda6b
Step 2/30 : MAINTAINER Sam Rogers srogers@uk.ibm.com
 ---> Running in d914938c8e3d
 ---> 5719840be833
Removing intermediate container d914938c8e3d
Step 3/30 : RUN export DEBIAN_FRONTEND=noninteractive  && apt-get update &&     apt-get install -y --no-install-recommends
     coreutils     curl     debianutils     findutils     gawk     grep     libc-bin     lsb-release     mount     passwd
d     tar       util-linux
 ---> Running in ea016ca1eb6a
Ign http://archive.ubuntu.com trusty InRelease
Get:1 http://security.ubuntu.com trusty-security InRelease [65.9 kB]
Get:2 http://archive.ubuntu.com trusty-updates InRelease [65.9 kB]
Get:3 http://archive.ubuntu.com trusty-backports InRelease [65.9 kB]
Get:4 http://security.ubuntu.com trusty-security/universe Sources [80.7 kB]
```

```
Step 6/30 : ARG MQ_URL=http://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/messaging/mqadv/mqadv_dev903_linux_x86-64.tar.gz
 ---> Running in 7228d6aed3c6
 ---> 029ea64c09c2
Removing intermediate container 7228d6aed3c6
Step 7/30 : ARG MQ_PACKAGES="MQSeriesRuntime-*.rpm MQSeriesServer-*.rpm MQSeriesMsg*.rpm MQSeriesJava*.rpm MQSeriesJRE*.rpm MQSeriesGSKit*.rpm MQSe
riesWeb*.rpm"
 ---> Running in ab85ccaa87a0
 ---> ca79cd6dd083
Removing intermediate container ab85ccaa87a0
Step 8/30 : RUN mkdir -p /tmp/mq        && cd /tmp/mq           && curl -LO $MQ_URL     && tar -zxvf ./*.tar.gz         && groupadd --gid 1000 mqm
       && useradd --create-home --home-dir /home/mqm --uid 1000 --gid mqm mqm          && usermod -G mqm root  && cd /tmp/mq/MQServer           &&
./mqlicense.sh -text_only -accept       && rpm -ivh --force-debian $MQ_PACKAGES         && /opt/mqm/bin/setmqinst -p /opt/mqm -i        && rm -rf /
tmp/mq  && rm -rf /var/lib/apt/lists/*  && rm -rf /var/mqm      && sed -i 's/PASS_MAX_DAYS\t99999/PASS_MAX_DAYS\t90/' /etc/login.defs           &&
sed -i 's/PASS_MIN_DAYS\t0/PASS_MIN_DAYS\t1/' /etc/login.defs   && sed -i 's/password\t\[success=1 default=ignore\]\tpam_unix\.so obscure sha512/pa
ssword\t[success=1 default=ignore]\tpam_unix.so obscure sha512 minlen=8/' /etc/pam.d/common-password
 ---> Running in fbed010d10b0
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  684M  100  684M    0     0  2592k      0  0:04:30  0:04:30 --:--:-- 2556k
MQServer/
MQServer/MQSeriesXRService-9.0.3-0.x86_64.rpm
MQServer/README$S/
```

```
Step 14/30 : RUN mkdir /opt/ibm &&     curl http://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/integration/10.0.0.10-IIB-LINUX64-DEVELOP
ER.tar.gz     | tar zx --exclude iib-10.0.0.10/tools --directory /opt/ibm &&     /opt/ibm/iib-10.0.0.10/iib make registry global accept license sil
ently
 ---> Running in 3f88c94a015c
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
 98 1211M   98 1193M    0     0  2620k      0  0:07:53  0:07:46  0:00:07 2632k^C

License accepted
Group 'mqbrkrs' will be created
 ---> 34240887929b
Removing intermediate container 3d648c420875
Step 15/30 : COPY kernel_settings.sh /tmp/
 ---> 06653ccb96e0
Removing intermediate container 48f4fa474a7d
Step 16/30 : RUN echo "IIB_10:" > /etc/debian_chroot  &&     touch /var/log/syslog &&     chown syslog:adm /var/log/syslog &&
nel_settings.sh;sync &&     /tmp/kernel_settings.sh
 ---> Running in 6023ab1ea9b3
 ---> c3b8c0841816
Removing intermediate container 6023ab1ea9b3
Step 17/30 : RUN useradd --create-home --home-dir /home/iibuser -G mqbrkrs,sudo,mqm iibuser &&     sed -e 's/^%sudo     .*/%sudo
D:ALL/g' -i /etc/sudoers

 ---> 1dac954c24ab
Removing intermediate container f90d1b868247
Step 30/30 : ENTRYPOINT iib_manage.sh
 ---> Running in cb69dda0159d
 ---> f949560c55bc
Removing intermediate container cb69dda0159d
Successfully built f949560c55bc
-bash-4.2#
```

Copy the IIB-MQ image to the ICP repository

Ensure the terminal can ping the correct cluster via hostname

Change the hosts file to ensure icpcluster.icp is pointing to the correct Pure App IP address in this example 172.23.50.112

```
-bash-4.2# vi /etc/hosts
-bash-4.2# ping icpcluster.icp
PING icpcluster.icp (172.23.49.17) 56(84) bytes of data.
64 bytes from icpcluster.icp (172.23.49.17): icmp_seq=1 ttl=63 time=0.181 ms
64 bytes from icpcluster.icp (172.23.49.17): icmp_seq=2 ttl=63 time=0.162 ms
^C
--- icpcluster.icp ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.162/0.171/0.181/0.016 ms
-bash-4.2# docker login icpcluster.icp:8500
Username (admin): admin
Password:
Login Succeeded
```

Docker login

Log in with admin/admin using command

Docker login icpcluster.icp:8500

Check the contents of the repository

```
-bash-4.2# docker images
REPOSITORY                                      TAG                 IMAGE ID            CREATED             SIZE
icpcluster.icp:8500/default/iib-mq              latest              f949560c55bc        4 minutes ago       2.22 GB
icpcluster.icp:8500/default/watson-talk         latest              18d3b4a1f59f        5 days ago          746 MB
icpcluster.icp:8500/default/watson              latest              fcb436d4b877        5 days ago          746 MB
node                                            6                   2fcc51d4e73d        6 days ago          662 MB
hitomitak/agentless-crawler                     latest              96e469d3dfaf        6 days ago          712 MB
icpcluster.icp:8500/default/agentless-crawler   latest              96e469d3dfaf        6 days ago          712 MB
```

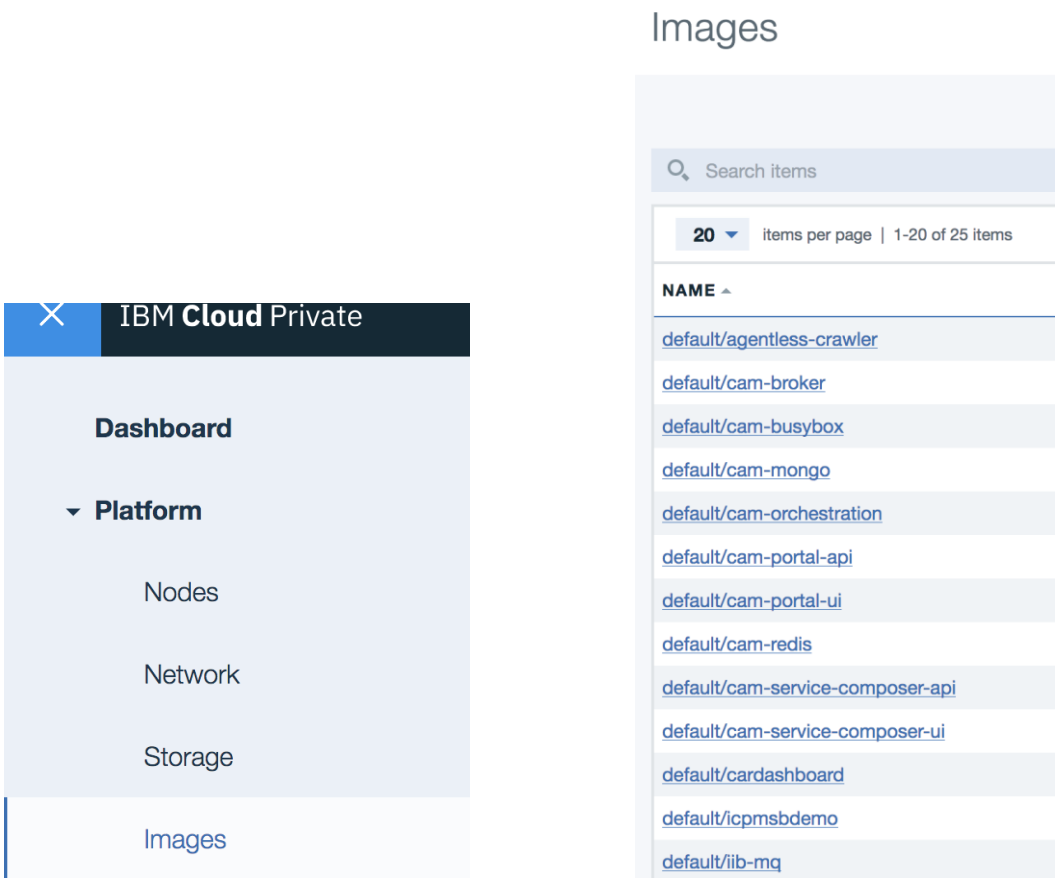
Docker push the image to ICP

```
-bash-4.2# docker push icpcluster.icp:8500/default/iib-mq:latest
The push refers to a repository [icpcluster.icp:8500/default/iib-mq]
fd298fd47be7: Pushed
00387f1a1382: Pushed
b5a36b864b9c: Pushed
580eba8bd60b: Pushed

c088f4b849d4: Pushed
c08b59ef4a3d: Pushed
latest: digest: sha256:55398b2aba08f71bc8dc499323fc3e700566b17f03b9bc14a2aa86e5a5f51b0d size: 5735
```

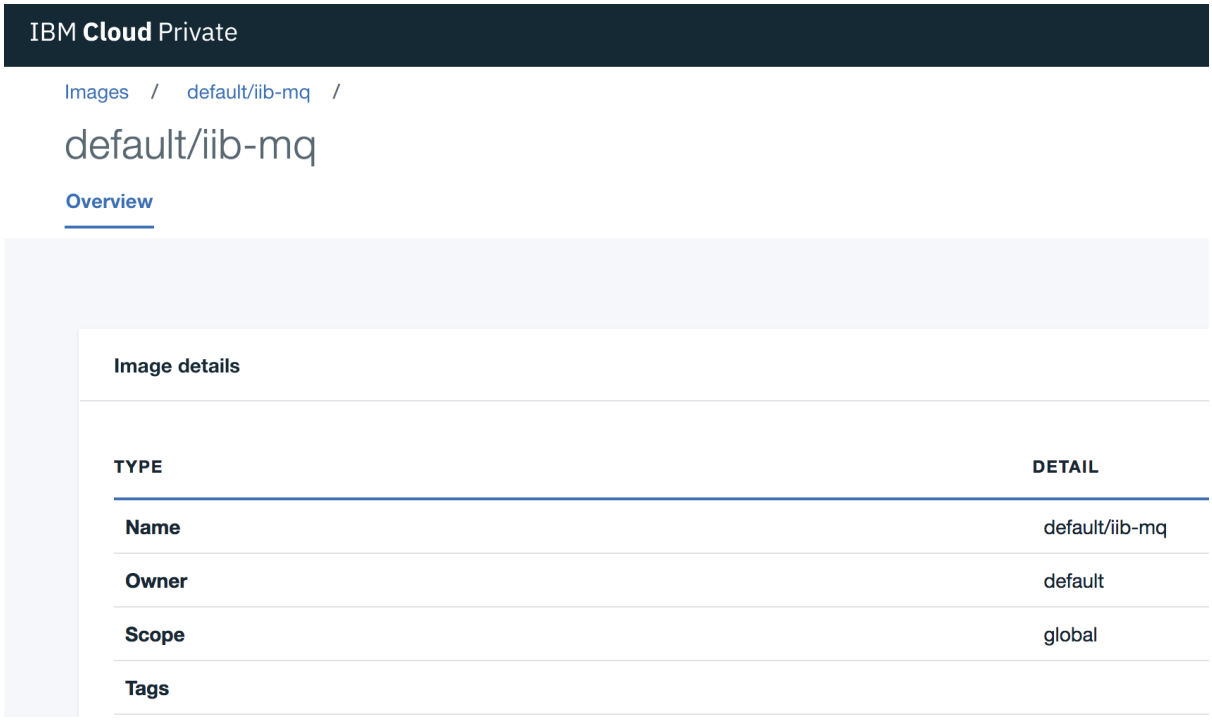
Check it was successful

```
-bash-4.2# docker images
REPOSITORY                                      TAG                 IMAGE ID            CREATED             SIZE
icpcluster.icp:8500/default/iib-mq              latest              f949560c55bc        12 minutes ago      2.22 GB
icpcluster.icp:8500/default/watson-talk         latest              18d3b4a1f59f        5 days ago          746 MB
```

Check the IIB-MQ image via the ICP Console

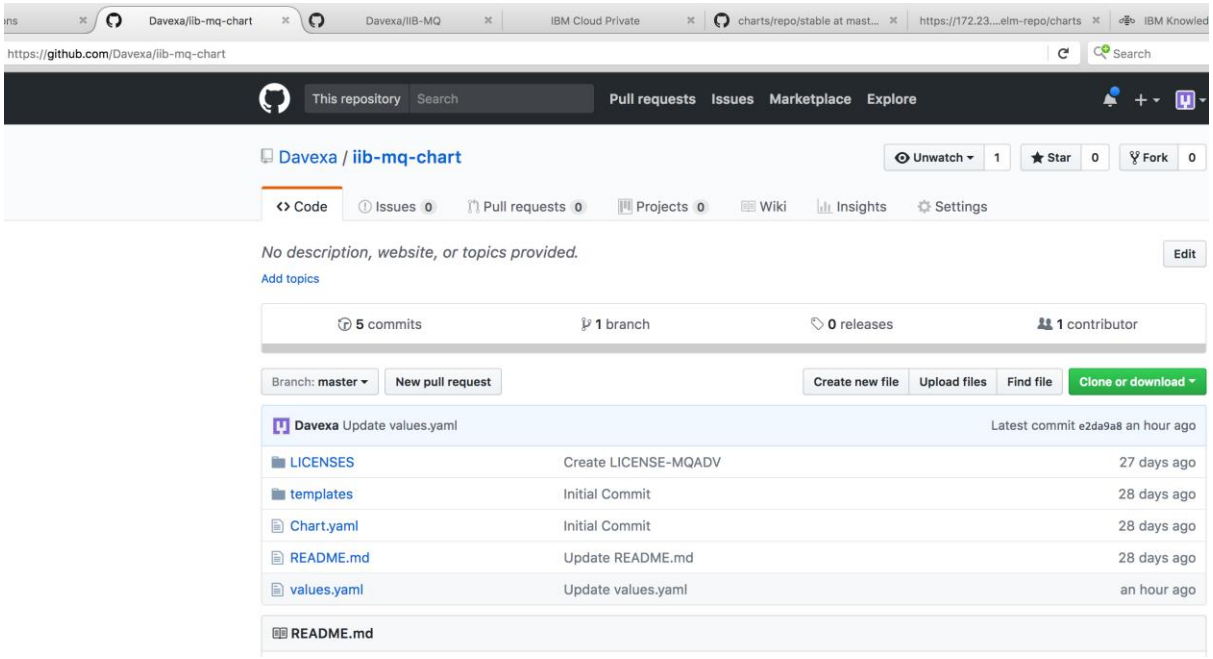


Click on default/iib-mq



Adding a new (custom) IIB and MQ Helm chart to ICP

MQ and IIB Helm Chart repository on github

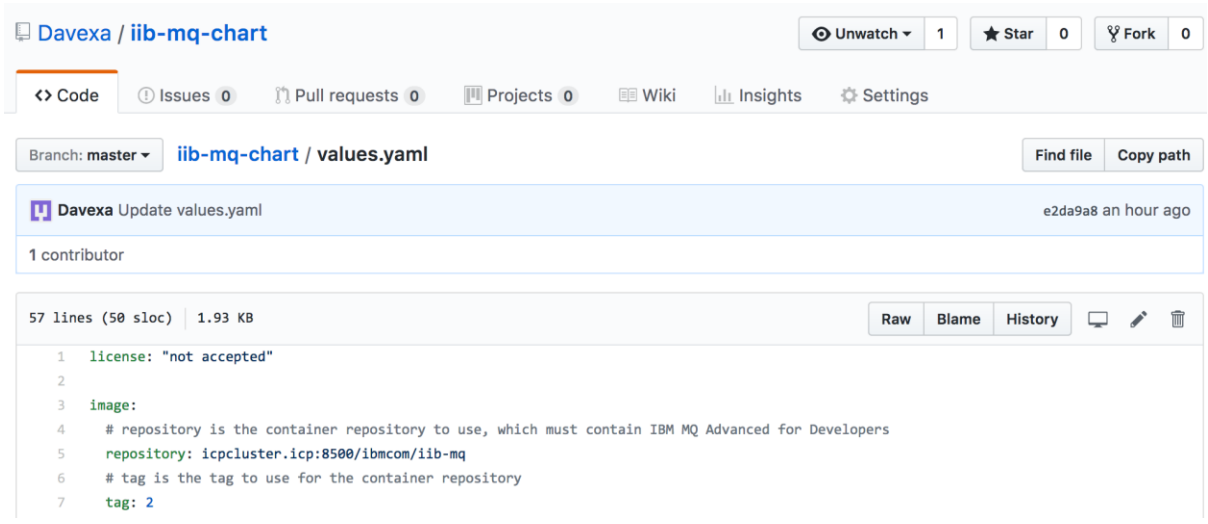


Edit the values.yaml file

At this point you have a choice to make.

1) Edit and commit the values.yaml here on github
2) Edit the values.yaml after the git clone to local system leaving the github copy “vanilla”

In this example we’ll perform the edit on Github and then commit before the clone.



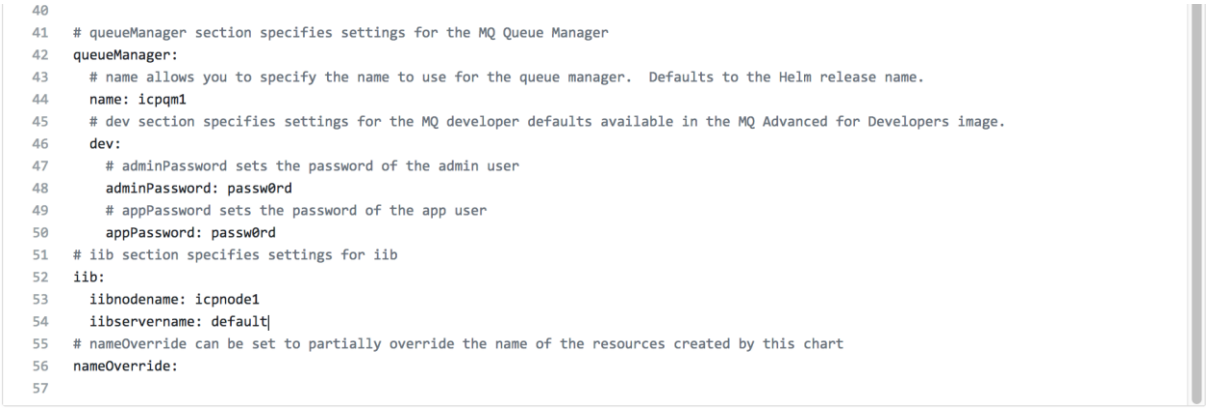
Edit the values.yaml file and ensure the respository and tag are correct.

The tag will need to be **latest**

The repository will need to resolve to the icp instance and check that the namespace is correct **default** in this case.

Set the license to “accept”

Page down and fill in the IIB and MQ specific values. Remember the spaces after the colons.

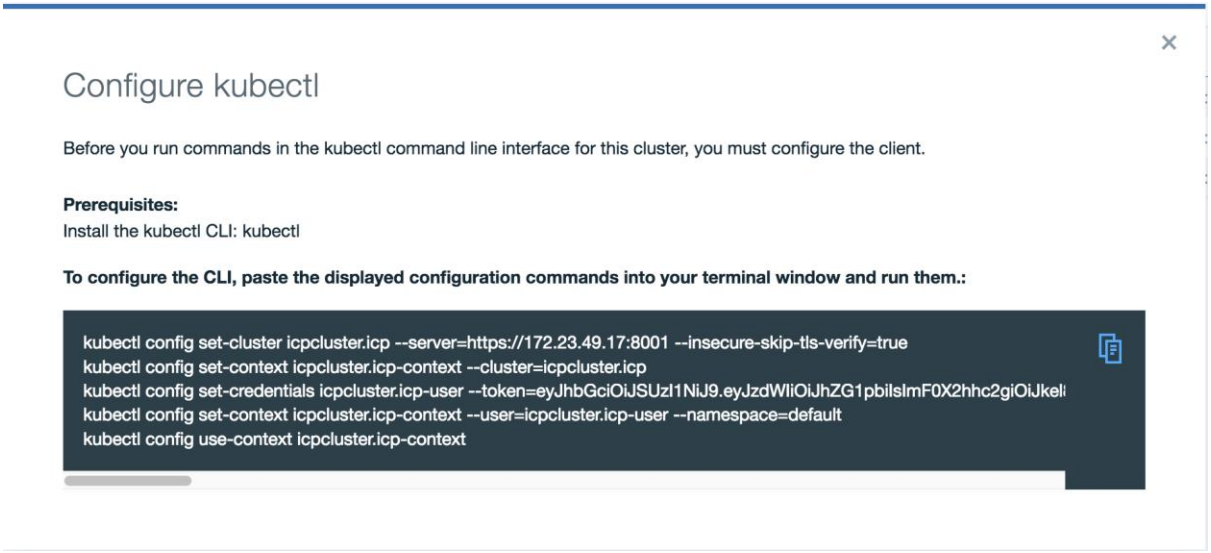


Git Clone the Helm Chart repository to local machine (not ICP in this example)

In this case the local machine is the VM image on the Pure App where all the command line tools are installed. If you have this set up on your machine you will not need to be ssh into the Pure.

```
[-bash-4.2# ls
10-listener.mqsc  CLA.md      iib-license-check.sh  kernel_settings.sh  mq-dev-config        mq.sh          README.md         switch.json
admin.json        Dockerfile  iib_manage.sh         LICENSE             mq-dev-config.sh     odbc.ini       setup-mqm-web.sh
agentx.json       iib_env.sh  iib-mq.yml            login.defs          mq-license-check.sh  odbcinst.ini   setup-var-mqm.sh
[-bash-4.2# cd ..
[-bash-4.2# ls
 cfc-crawler  container-service-getting-started-wt  IIB-MQ  repos
[-bash-4.2# git clone https://github.com/Davexa/iib-mq-chart.git
Cloning into 'iib-mq-chart'...
remote: Counting objects: 28, done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 28 (delta 10), reused 20 (delta 6), pack-reused 0
Unpacking objects: 100% (28/28), done.
[-bash-4.2# ls
 cfc-crawler  container-service-getting-started-wt  IIB-MQ  iib-mq-chart  repos
```

Configure the Kubernetes client to connect to ICP



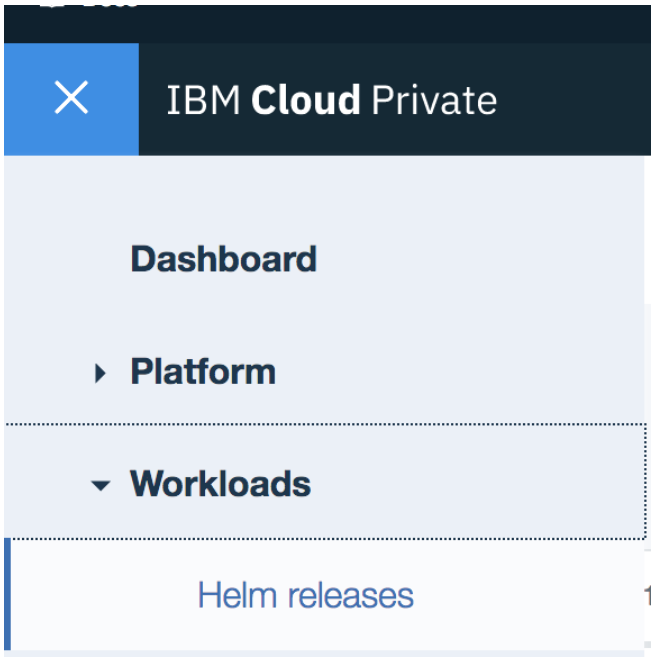

```
-bash-4.2# kubectl config set-cluster icpcluster.icp --server=https://172.23.49.17:8001 --insecure-skip-tls-verify=true
Cluster "icpcluster.icp" set.
-bash-4.2# kubectl config set-context icpcluster.icp-context --cluster=icpcluster.icp
Context "icpcluster.icp-context" modified.
-bash-4.2# kubectl config set-credentials icpcluster.icp-user --token=eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJhZG1pbiIsImF0X2hhc2giOiJkel8yVU9od0xfZnBrNGkx
ZEJhR2ZnIiwiaXNzIjoiaHR0cHM6Ly9pY3BjbHVzdGVyLmljcDo5NDQzL29pZGMvZW5kcG9pbnQvT1AiLCJhdWQiOiJkMzFkZDgxMjVmNjg3ZjIxMzE1Y2ZjYTEyYTY5ZjI4MiIsImV4cCI6MTU
xMzA4ODE0NSwiaWF0IjoxNTEzMDQ0OTQ1fQ.Nfd5IndDdszng0vF8qMYRGUSzrKaWNuf_YDI3Hl9nd5oPAwoijPnm9bxjfE6YQlGDdutEb0vwLDnHX1zQU2j4eeoAM1IDOx4bA5A-yM5mMc6mkT
d8nngO0Sj6p5D3yr_xm5OD3GyUvNAWBMG6hb9sN08o2rUlaCpJowm9J7nEDelsdkt-Yk1cDMMgiJqSxwV6Jp_Q8Z1hMznuk_Td5IbzkEFBNdzVzfllrYdMDgTvfms_zz7jR081j6ycghIxj2lHo
P-TLT7oN0PdQifkQK-b1aL8tXmdGz0BIkzdCvDCztL3fX-7w78mAiGImt444kca9NWxk6cS-SrcUjnqImnwA
User "icpcluster.icp-user" set.
-bash-4.2# kubectl config set-context icpcluster.icp-context --user=icpcluster.icp-user --namespace=default
Context "icpcluster.icp-context" modified.
-bash-4.2# kubectl config use-context icpcluster.icp-context
Switched to context "icpcluster.icp-context".
```

Install the Helm Chart from the local machine (not ICP UI in this example)

```
-bash-4.2# pwd
/root/git
-bash-4.2# ls
cfc-crawler  container-service-getting-started-wt  IIB-MQ  iib-mq-chart  repos
-bash-4.2# helm install --name daiibmq iib-mq-chart
```

```
NAME:   daiibmq
LAST DEPLOYED: Tue Dec 12 15:39:37 2017
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1beta1/StatefulSet
NAME            DESIRED  CURRENT  AGE
daiibmq-iib-mq  1        1        1s

==> v1/Secret
NAME            TYPE    DATA  AGE
daiibmq-iib-mq  Opaque  1     1s

==> v1/Service
NAME            CLUSTER-IP  EXTERNAL-IP  PORT(S)                                                                AGE
daiibmq-iib-mq  10.0.0.251  <nodes>      1414:30447/TCP,9443:30019/TCP,4414:30116/TCP,7080:31450/TCP,7800:31847/TCP  1s
```

Check the deploy in the ICP console



daiibmq

DETAILS

TYPE	DETAIL
Name:	daiibmq
Namespace:	default
Status:	DEPLOYED
Chart name:	iib-mq
Chart version:	1.1.0
updated:	Dec 12th 2017 at 3:39 PM
Type:	Release

SECRET

NAME	TYPE	DATA
daiibmq-iib-mq	Opaque	1

SERVICE

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)
daiibmq-iib-mq	10.0.0.251	<nodes>	1414:30447/TCP,9443:30019/TCP,4414:30116/TCP,7080:31450/TCP,7800:31847/TCP

Select the Stateful set

StatefulSet / daiibmq-iib-mq / daiibmq-iib-mq-0 /

daiibmq-iib-mq-0

Overview Containers Events Logs

Pod details

TYPE	DETAIL
Name	daiibmq-iib-mq-0
Namespace	default
Start time	Dec 12th 2017 at 3:53 PM
Labels	QM_IDENTIFIER=daiibmq,app=daiibmq-iib-mq,chart=iib-mq-1.1.0,controller-revision-hash=daiibmq-iib-mq-4259841562,heritage=Tiller,release=daiibmq
PodSecurityPolicy applied	default
Node	172.23.49.9
IP	10.1.20.134
Status	Running

Overview Containers Events Logs

NAME	IMAGE	PORT	STATE
qmgr	icpcluster.icp:8500/default/iib-mq:latest	1414/TCP,9443/TCP,4414/TCP,7080/TCP,7800/TCP	Running

Overview Containers Events Logs

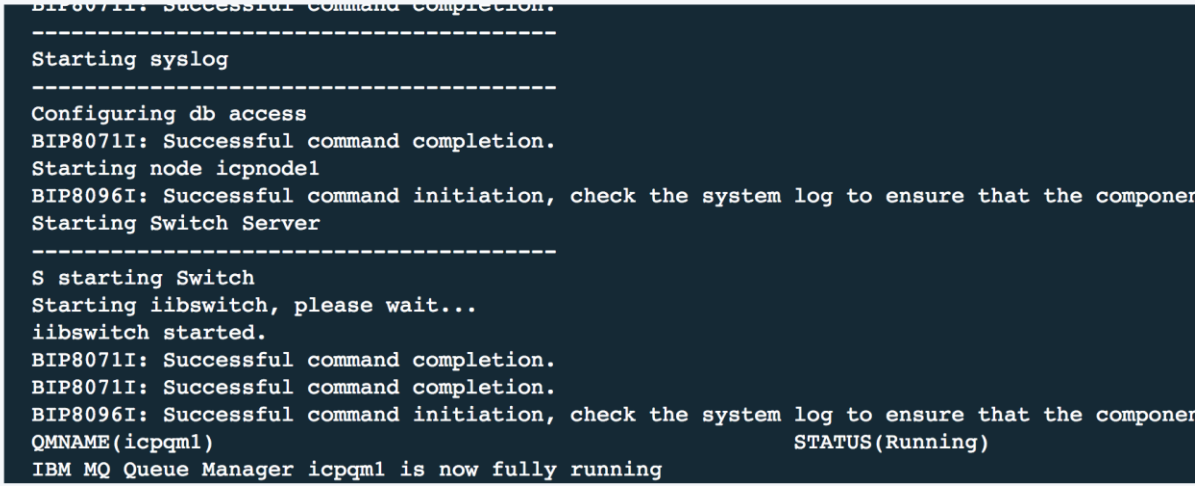
Search items

20 items per page | 1-5 of 5 items

TYPE	SOURCE	COUNT	REASON	MESSAGE
Normal	default-scheduler	1	Scheduled	Successfully assigned daiibmq-iib-mq-0 to 172.23.49.9
Normal	kubelet 172.23.49.9	1	SuccessfulMountVolume	MountVolume.SetUp succeeded for volume "default-token-qrz9z"
Normal	kubelet 172.23.49.9	1	Pulled	Container image "icpcluster.icp:8500/default/iib-mq:latest" already present on machine
Normal	kubelet 172.23.49.9	1	Created	Created container
Normal	kubelet 172.23.49.9	1	Started	Started container

daiibmq-iib-mq-0

Overview Containers Events Logs

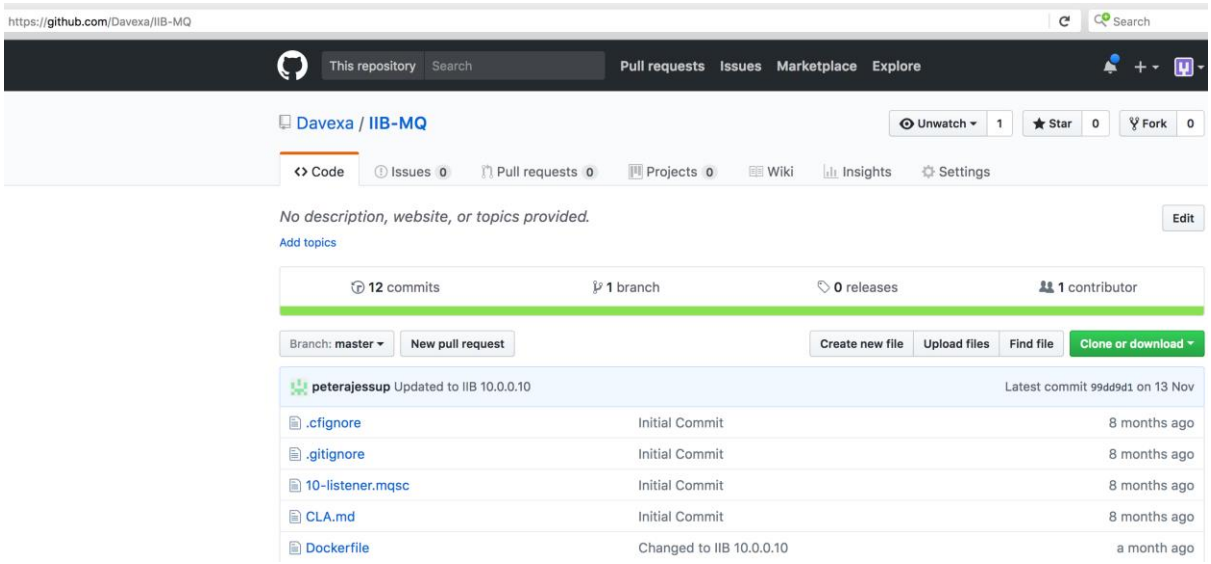


IIB & MQ Container on ICP from Github repository via MSB pipeline

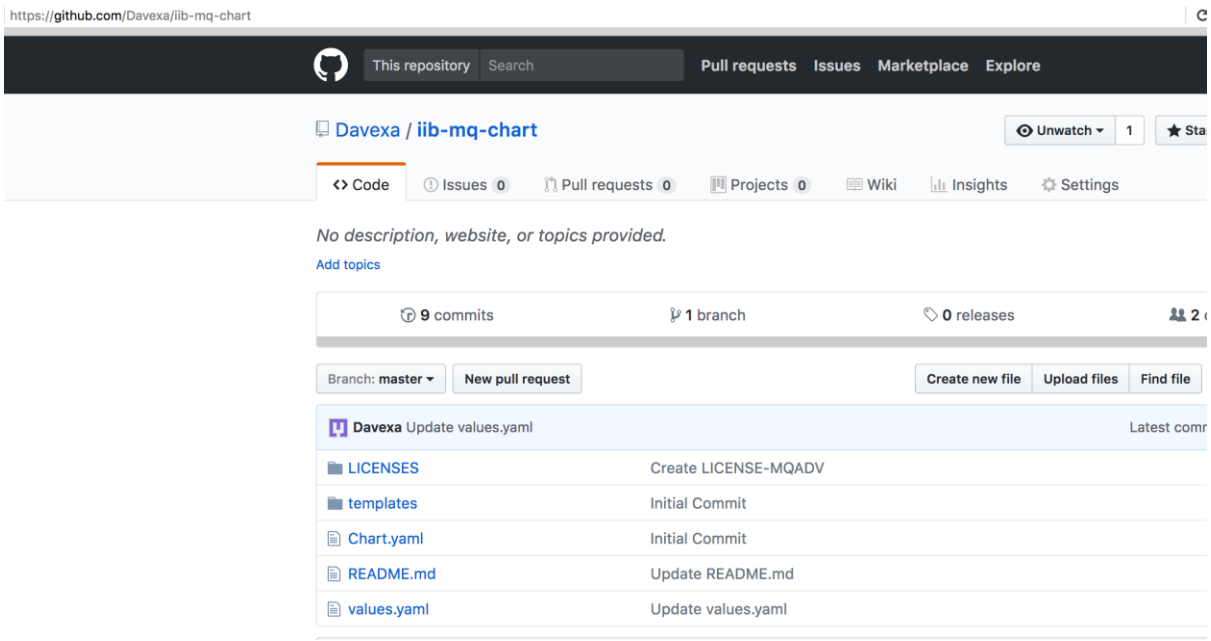
Creating the IIB-MQ Repository in GitHub Organization

Source GitHub Repositories to leverage

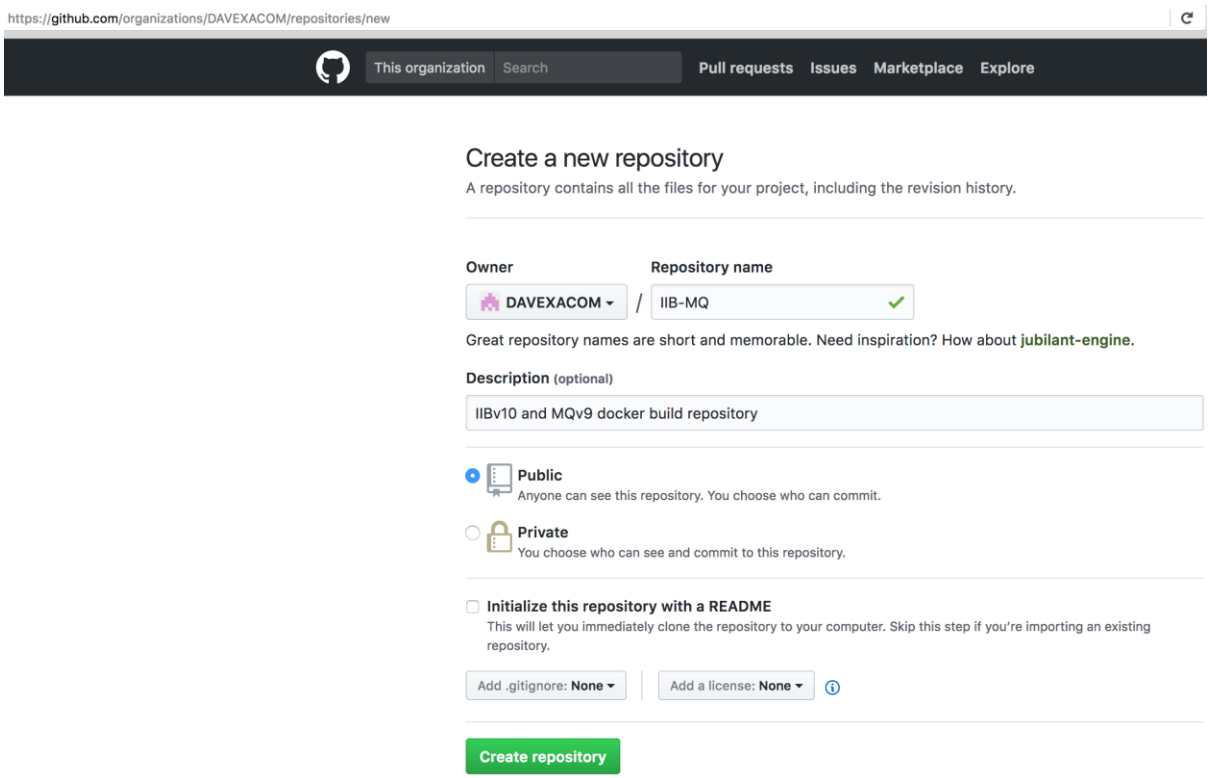
1. IIB an MQ Docker build repository



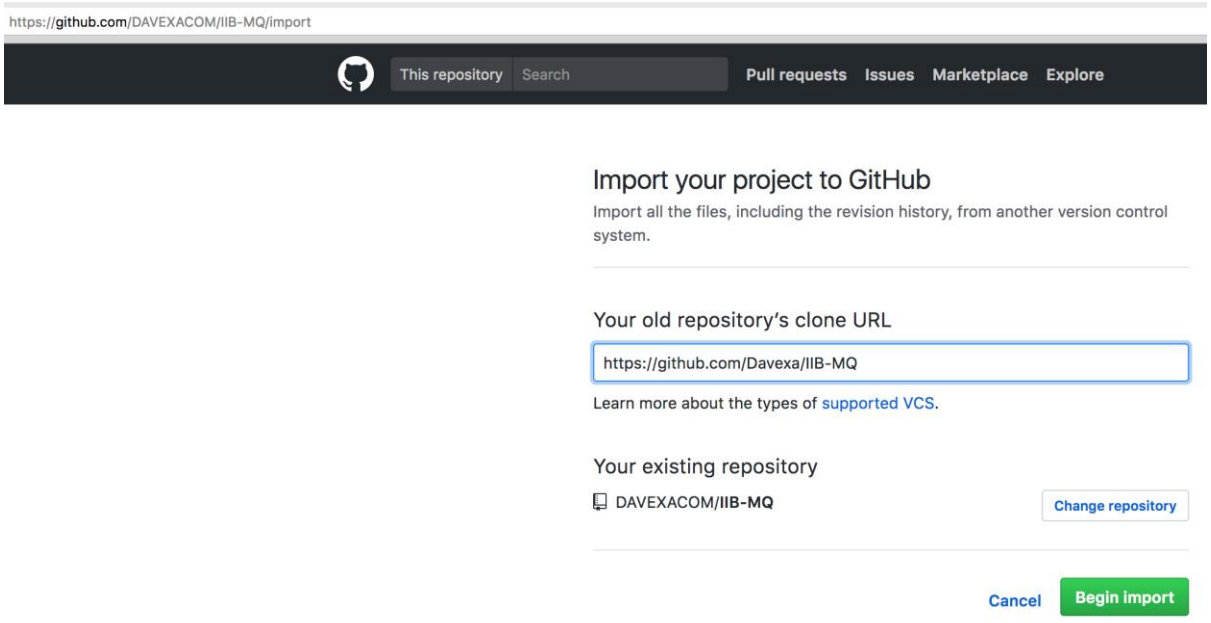
2. IIB and MQ Helm Chart repository

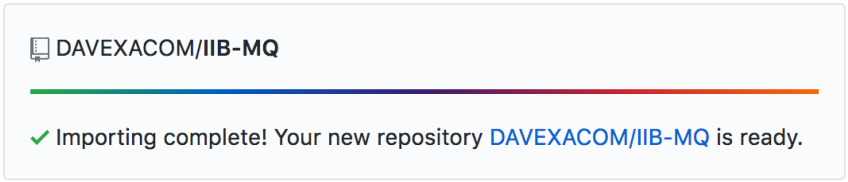


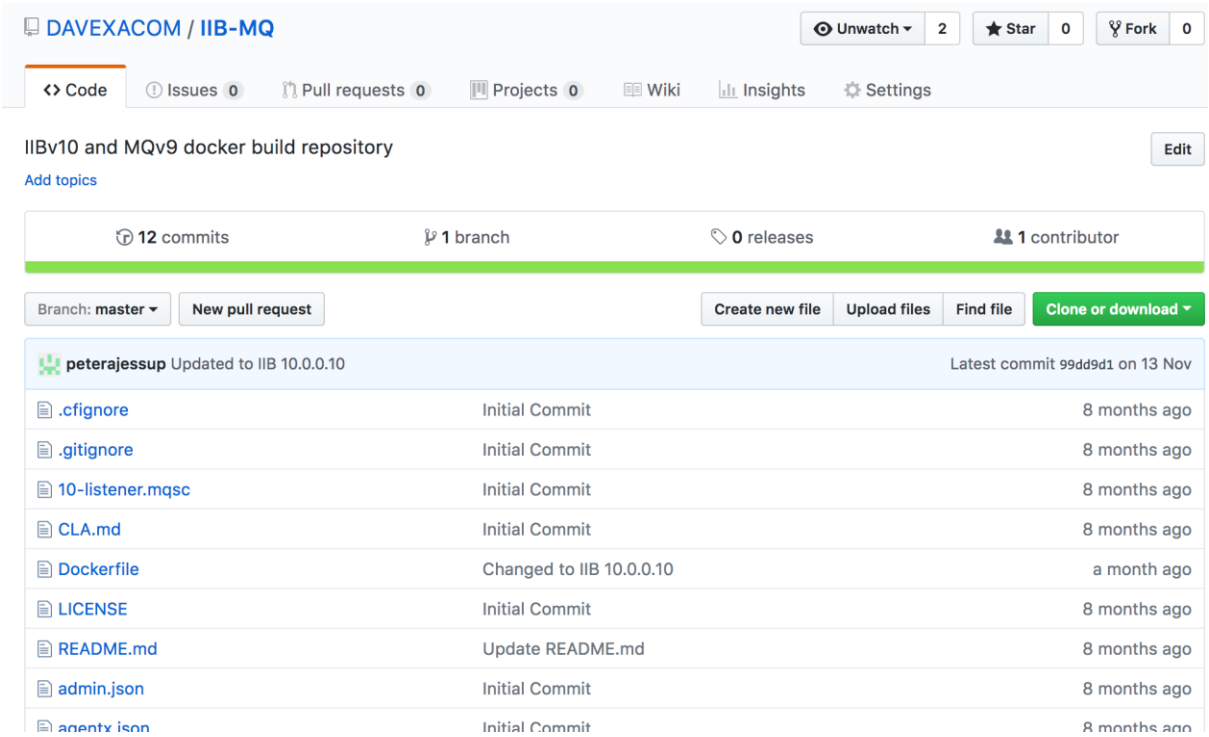
Target GitHub Repository in Github Organization



Import from existing repos

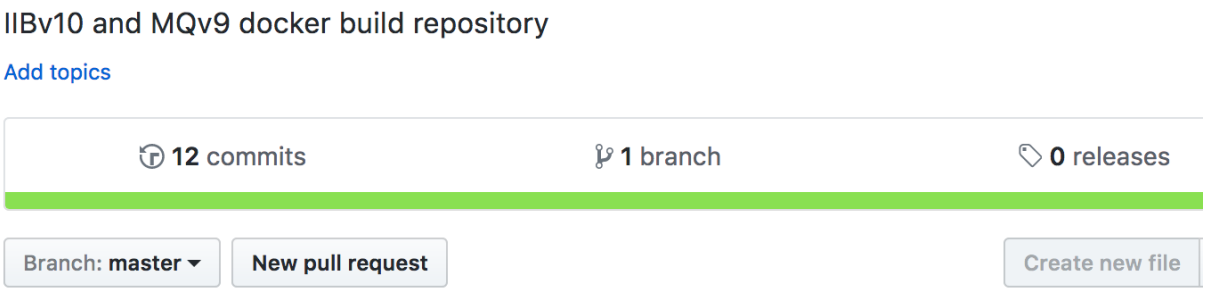






Add a Jenkins file to the repository

Use Create new file to create the Jenkinsfile

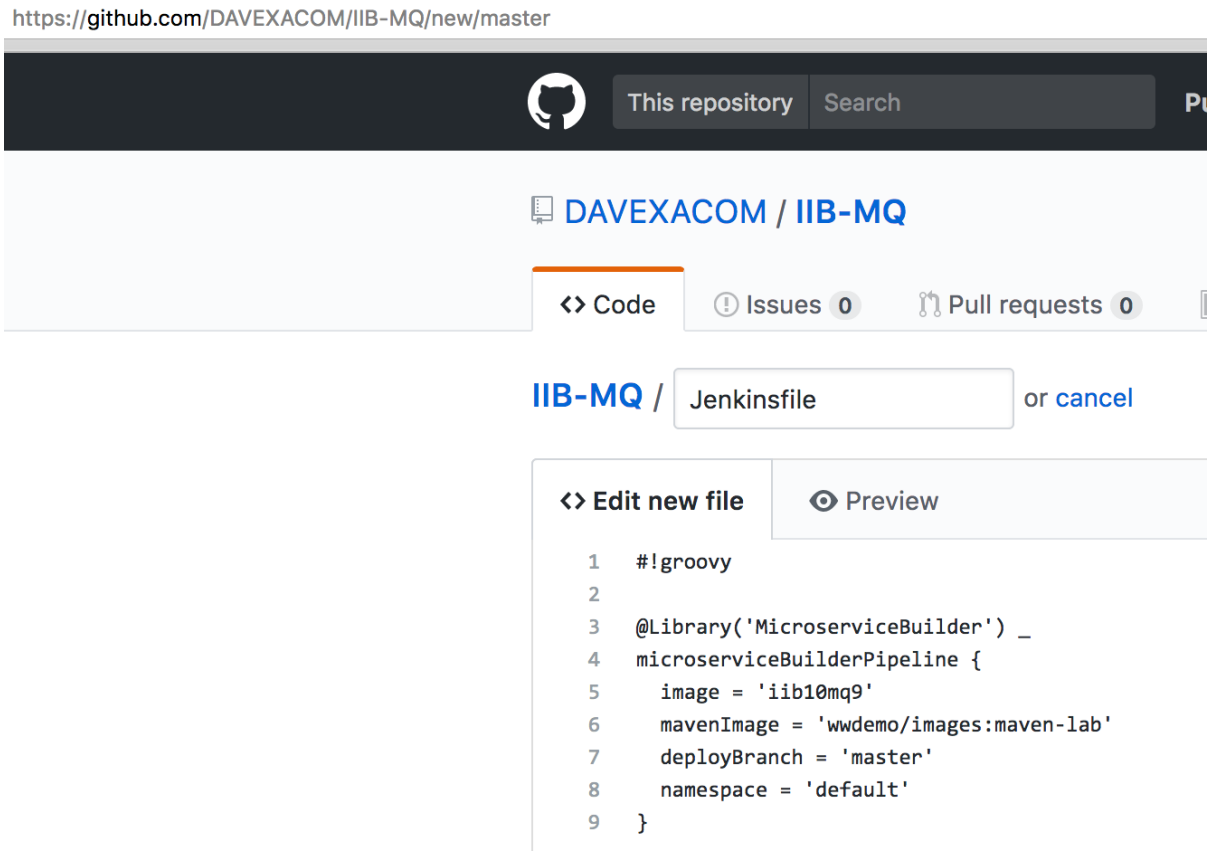


Name the file Jenkinsfile

Cut and paste the following or similar – you might want to change the image name if you prefer

#!groovy

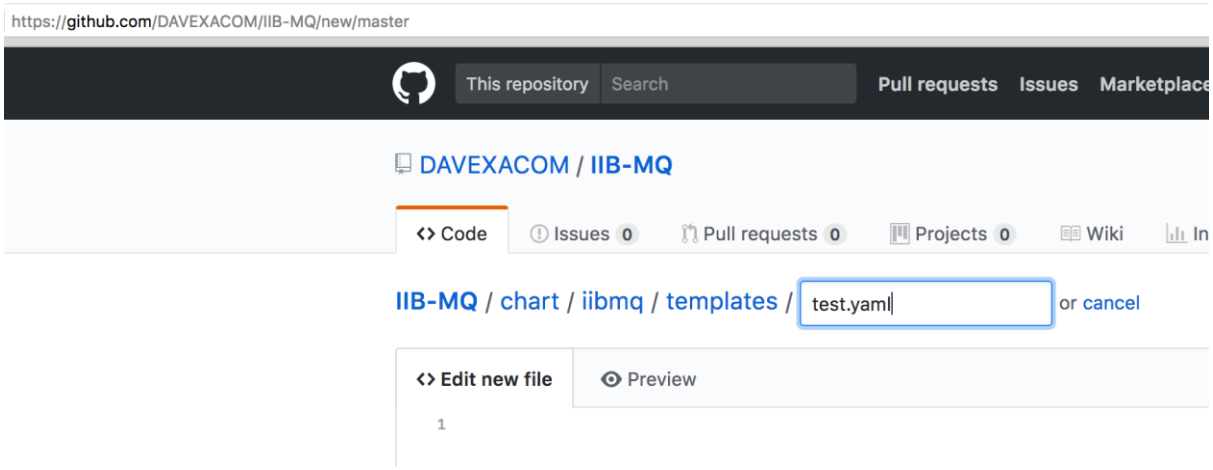
```
@Library('MicroserviceBuilder') _
microserviceBuilderPipeline {
  image = 'iib10mq9'
  mavenImage = 'wwdemo/images:maven-lab'
  deployBranch = 'master'
  namespace = 'default'
}
```

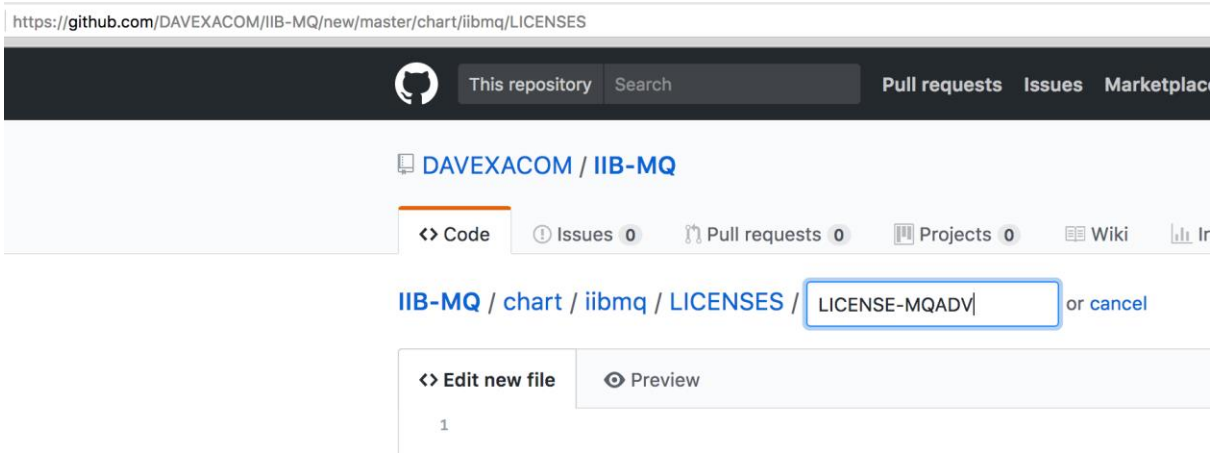


Page down and hit commit

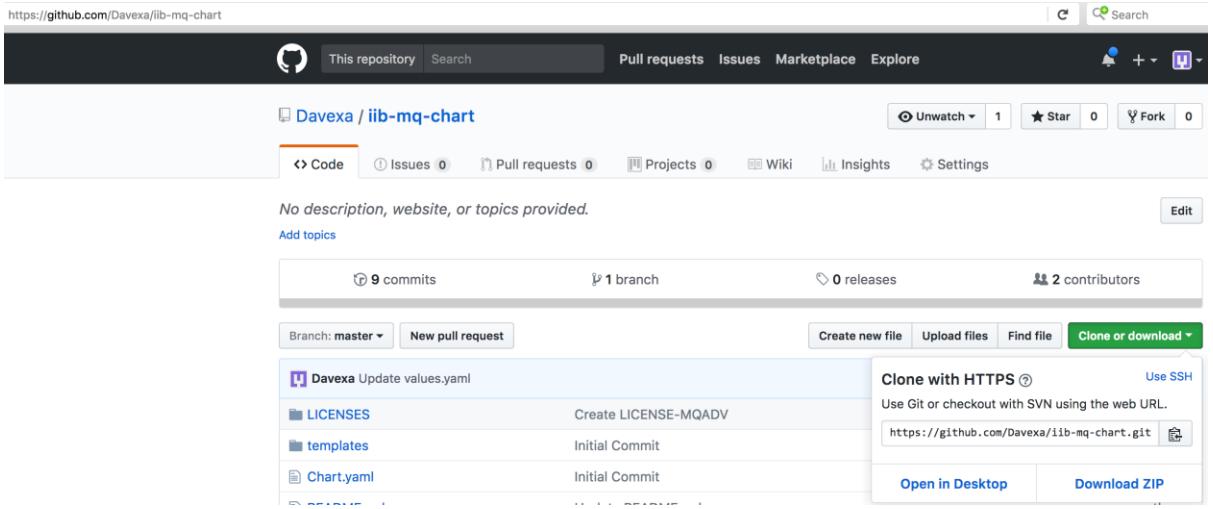
Add the IIB and MQ Helm chart files to the repository

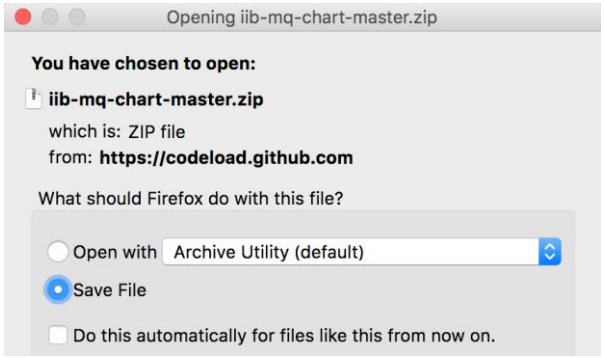
Create the directory structure chart/iibmq/templates and chart/iibmq/LICENSE

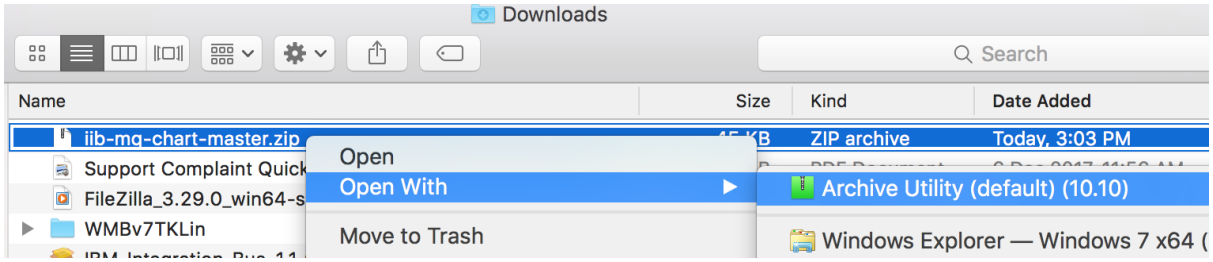


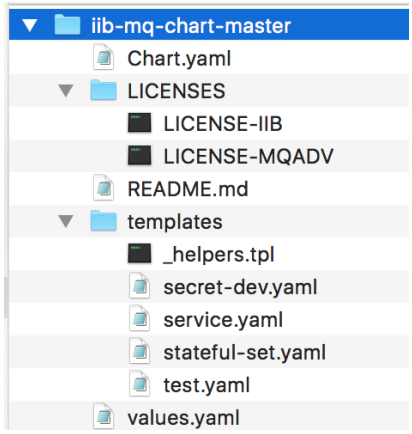


Download from the iib-mq-chart repository to the local file system

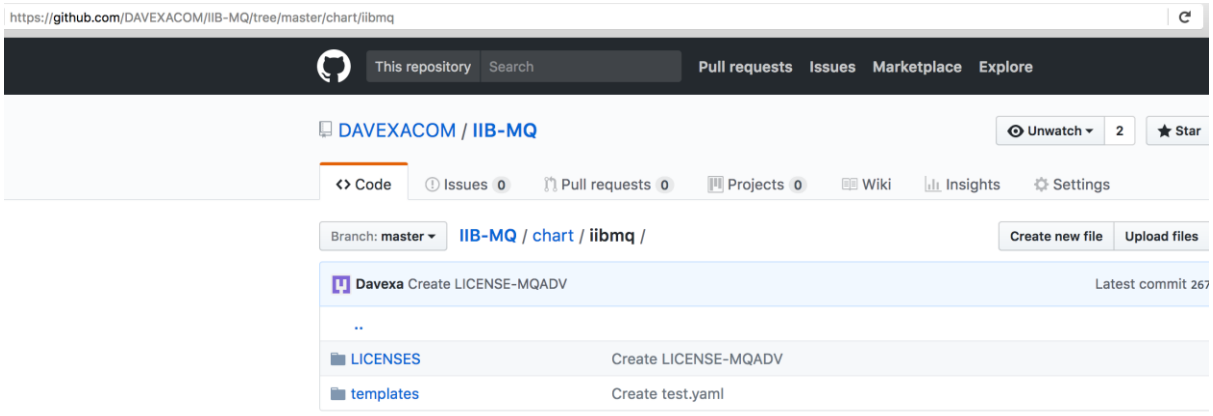


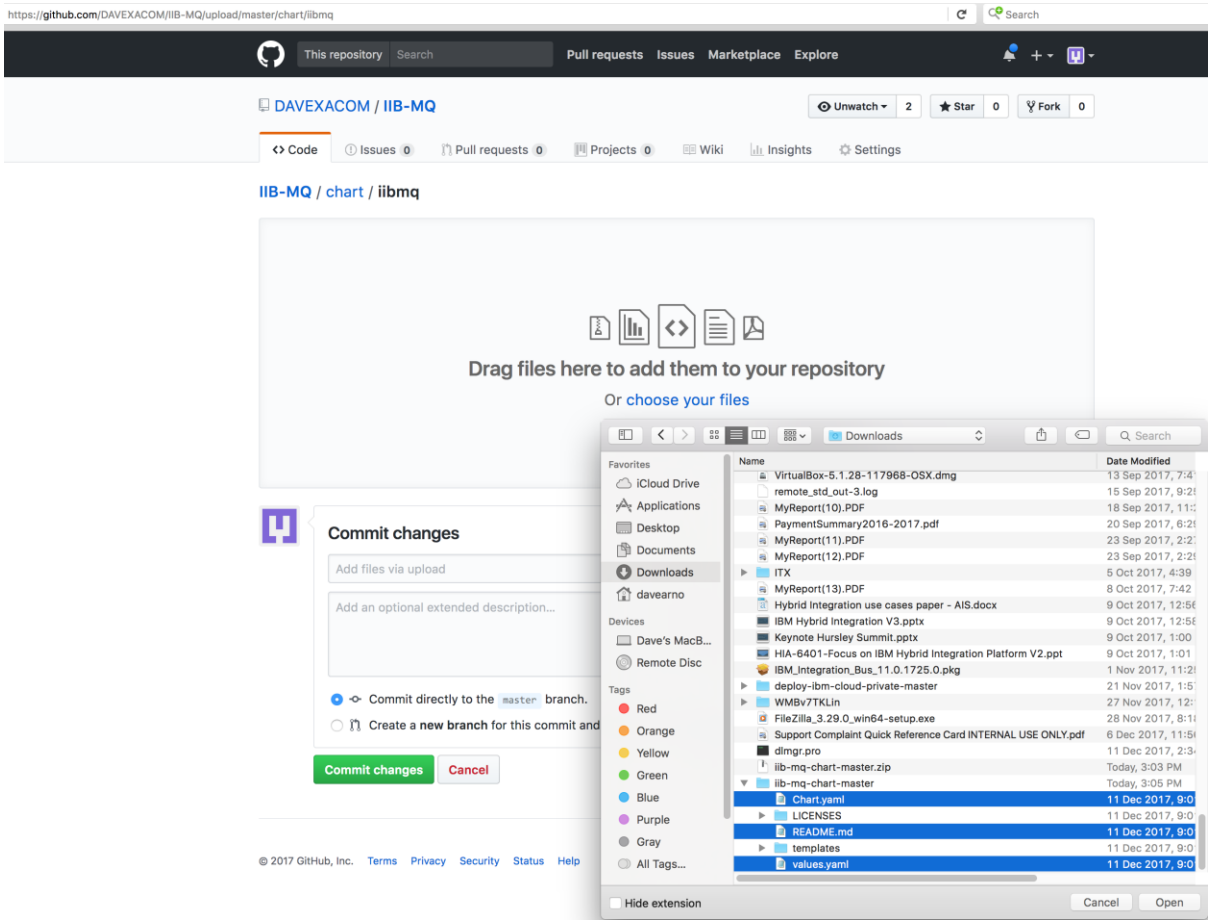




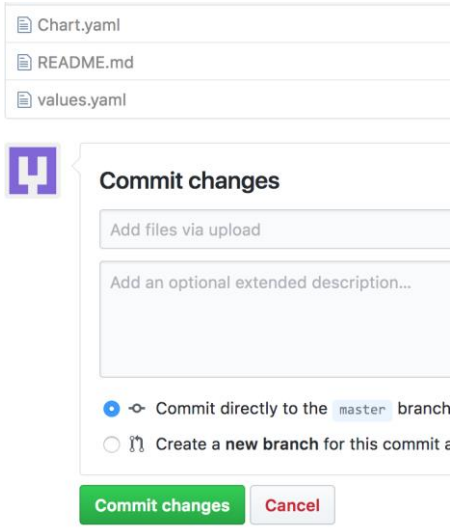


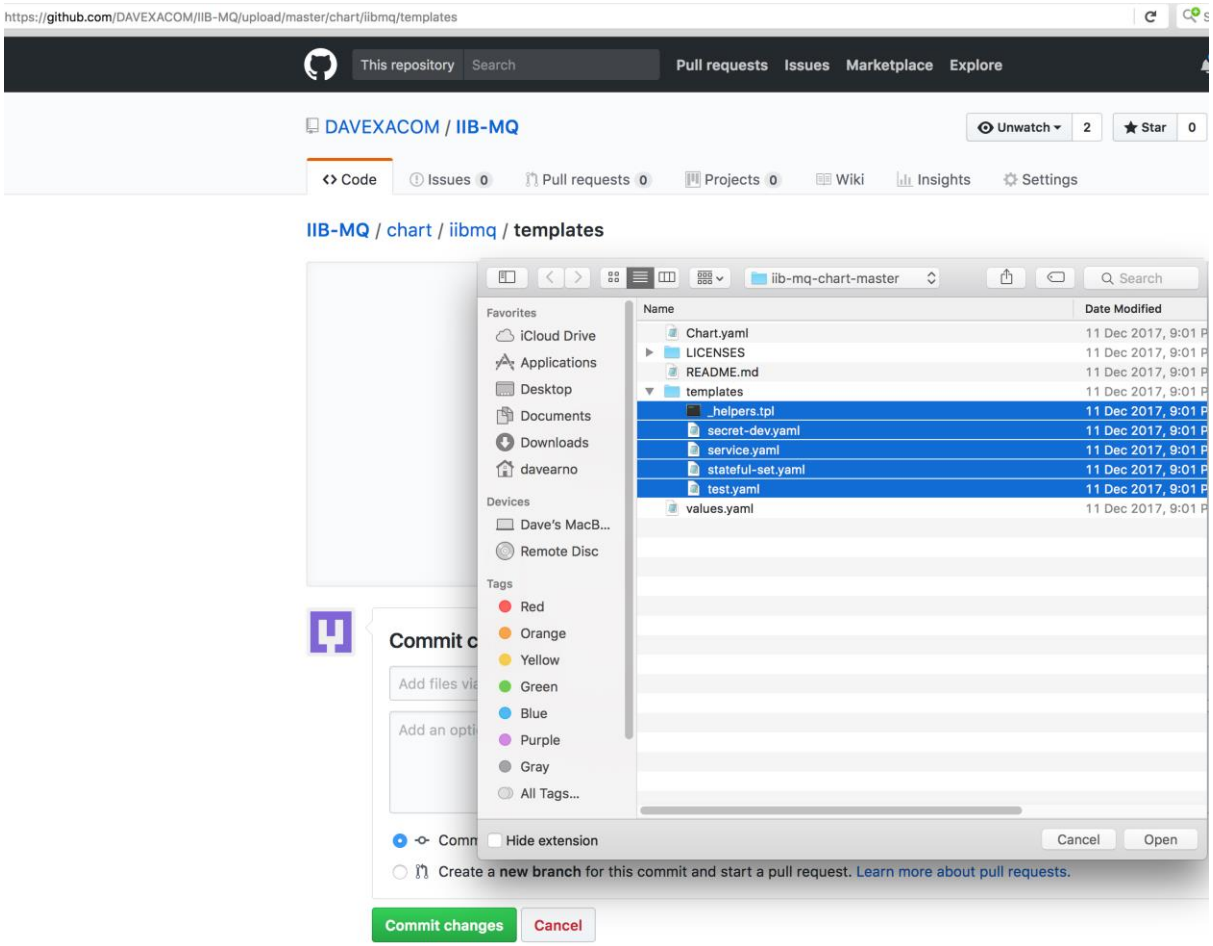
Import to from the iib-mq-chart repository to the local file system

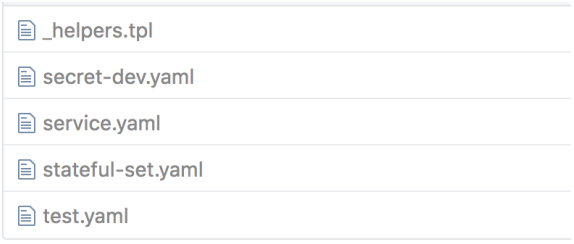


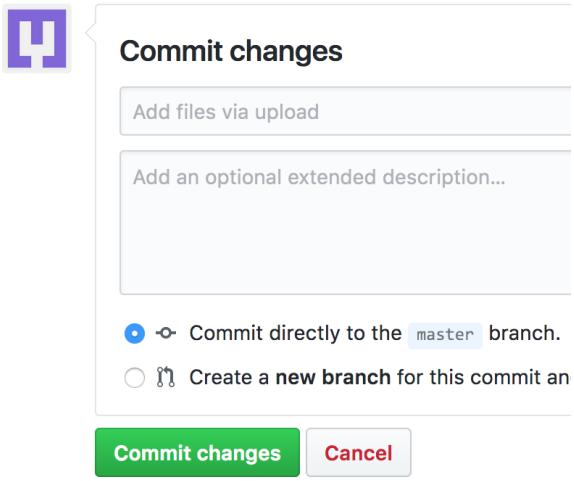


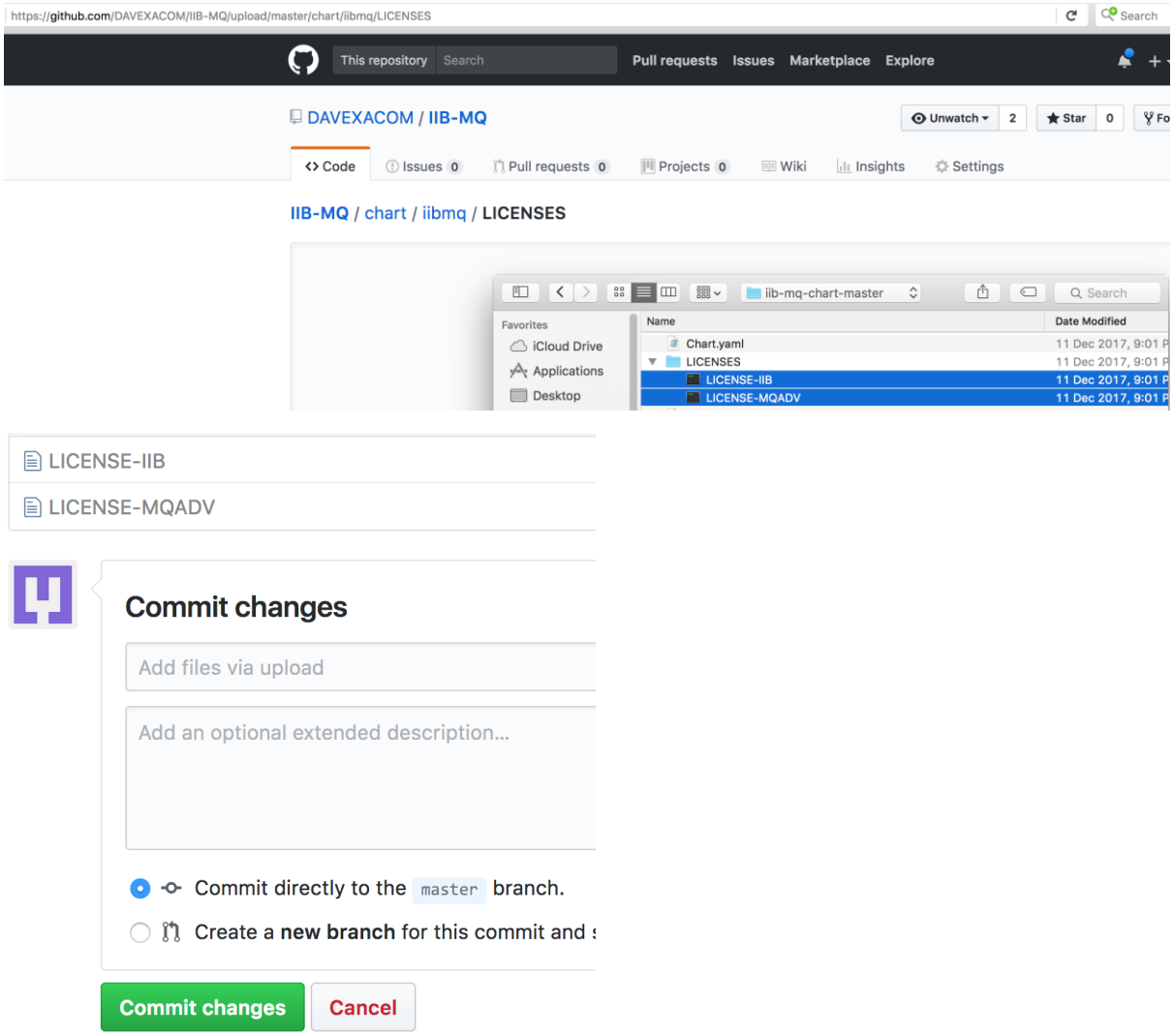
Commit changes




https://github.com/DAVEXACOM/IIB-MQ/upload/master/chart/iibmq/templates
This repository
Search
Pull requests
Issues
Marketplace
Explore
DAVEXACOM / IIB-MQ
Unwatch
2
Star
0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
IIB-MQ / chart / iibmq / templates
iib-mq-chart-master
Search
Favorites
iCloud Drive
Applications
Desktop
Documents
Downloads
davearno
Devices
Dave's MacB...
Remote Disc
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
Name
Date Modified
Chart.yaml
LICENSES
README.md
templates
_helpers.tpl
secret-dev.yaml
service.yaml
stateful-set.yaml
test.yaml
values.yaml
11 Dec 2017, 9:01 P
Hide extension
Cancel
Open
Create a new branch for this commit and start a pull request. Learn more about pull requests.
Commit changes
Cancel


_helpers.tpl
secret-dev.yaml
service.yaml
stateful-set.yaml
test.yaml


Commit changes
Add files via upload
Add an optional extended description...
Commit directly to the master branch.
Create a new branch for this commit an
Commit changes
Cancel



Update the Jenkinsfile to reference the helm charts files in the chart folder

Before :

https://github.com/DAVEXACOM/IIB-MQ/blob/master/Jenkinsfile

This repository Search Pull requests

DAVEXACOM / IIB-MQ

Code Issues 0 Pull requests 0 Projects

Branch: master IIB-MQ / Jenkinsfile

Davexa Create Jenkinsfile

1 contributor

10 lines (8 sloc) 188 Bytes

```groovy
#!groovy

@Library('MicroserviceBuilder') _
microserviceBuilderPipeline {
  image = 'iib10mq9'
  mavenImage = 'wwdemo/images:maven-lab'
  deployBranch = 'master'
  namespace = 'default'
}
```

After :

DAVEXACOM / IIB-MQ

Code Issues 0 Pull requests 0

Branch: master IIB-MQ / Jenkinsfile

Davexa Update Jenkinsfile

1 contributor

11 lines (9 sloc) 218 Bytes

```groovy
#!groovy

@Library('MicroserviceBuilder') _
microserviceBuilderPipeline {
  image = 'iib10mq9'
  mavenImage = 'wwdemo/images:maven-lab'
  chartFolder = 'chart/iibmq'
  deployBranch = 'master'
  namespace = 'default'
}
```

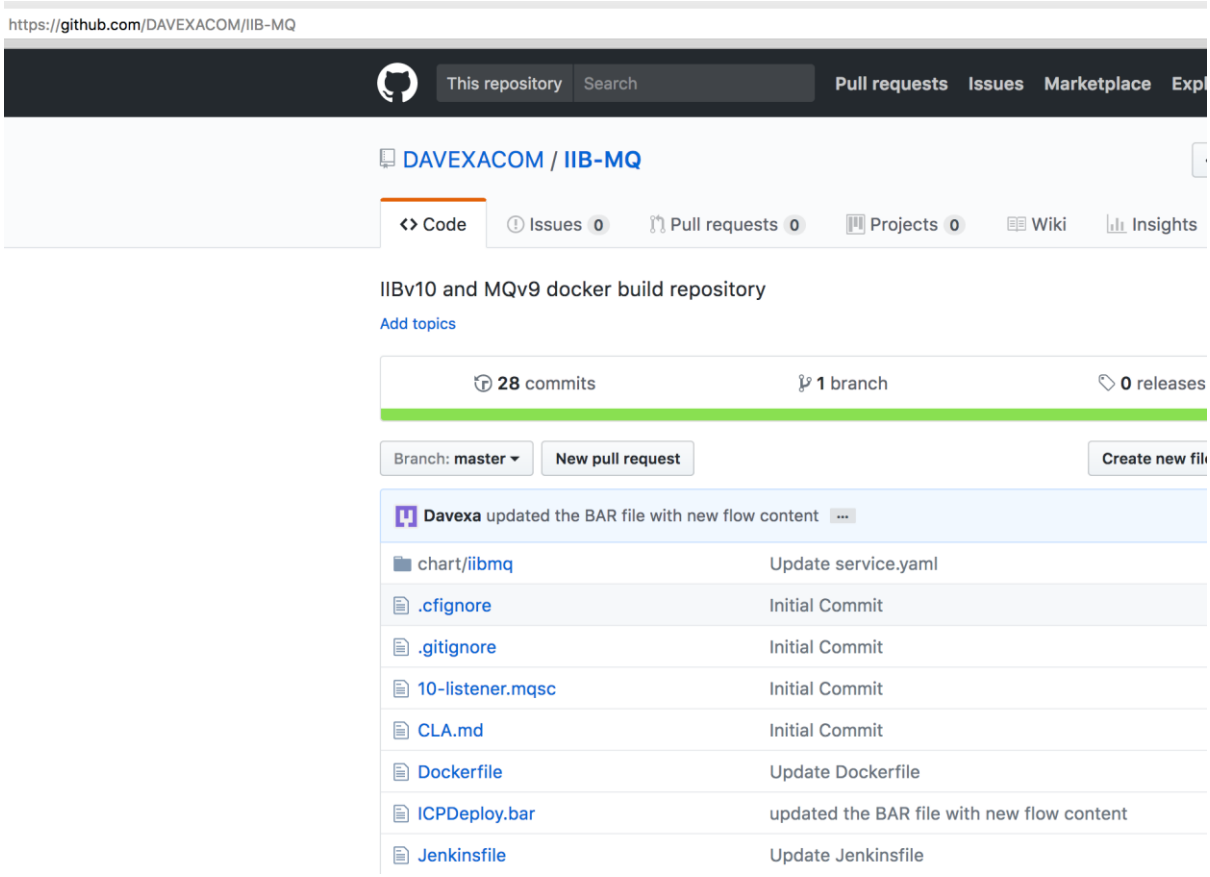
Important Parameters used in this example

As you go through the set up you need to capture these values as you go for use in later steps

Parameter Description	My example value	details
Github organization	DAVEXACOM	Github details
Github repository	IIB-MQ	Github details
Github username	Davexa	Github details
Github Personal access token	23a8537c82a19a358fb957bbfa7fba3d369eeb6b (not the real token)	Github details
Application Name	Icp-msb-iib-demo	Github OAuth registration
Homepage URL	http://172.23.50.111:32050 port dependent on Jenkins instance	Github OAuth registration needs the Jenkins URL
Authorization Call back URL	http://172.23.50.111:32050/securityRealm/finishLogin	Github OAuth registration needs the Jenkins URL
Client ID	e681fbc9e6491de43587	Github OAuth reg returns
mavenImage	'wwdemo/images:maven-lab'	Jenkins File
URL to OAuth Application details	https://github.com/settings/applications/635182	Github details
Github webhook payload URL	{Jenkins IpAddress]:[Jenkins port]/github-webhook/	URL github uses to publish to jenkins. Might be a Secure G/W endpoint rather than direct

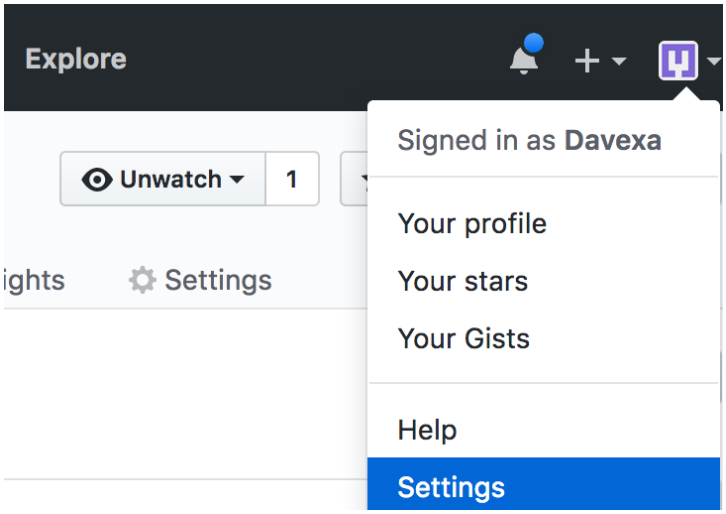
Configuring Github repository

Github repository IIB-MQ in organization DAVEXACOM

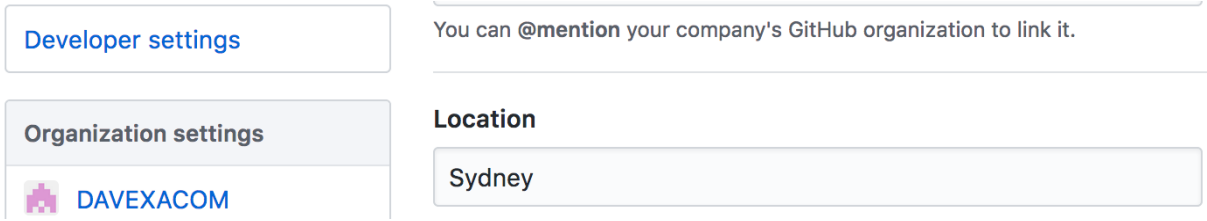


Github Personal Access Token – Create a token

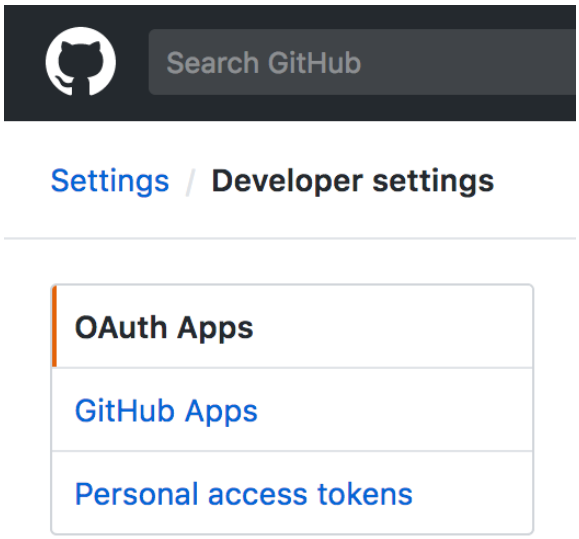
Developer Settings – Personal Access tokens



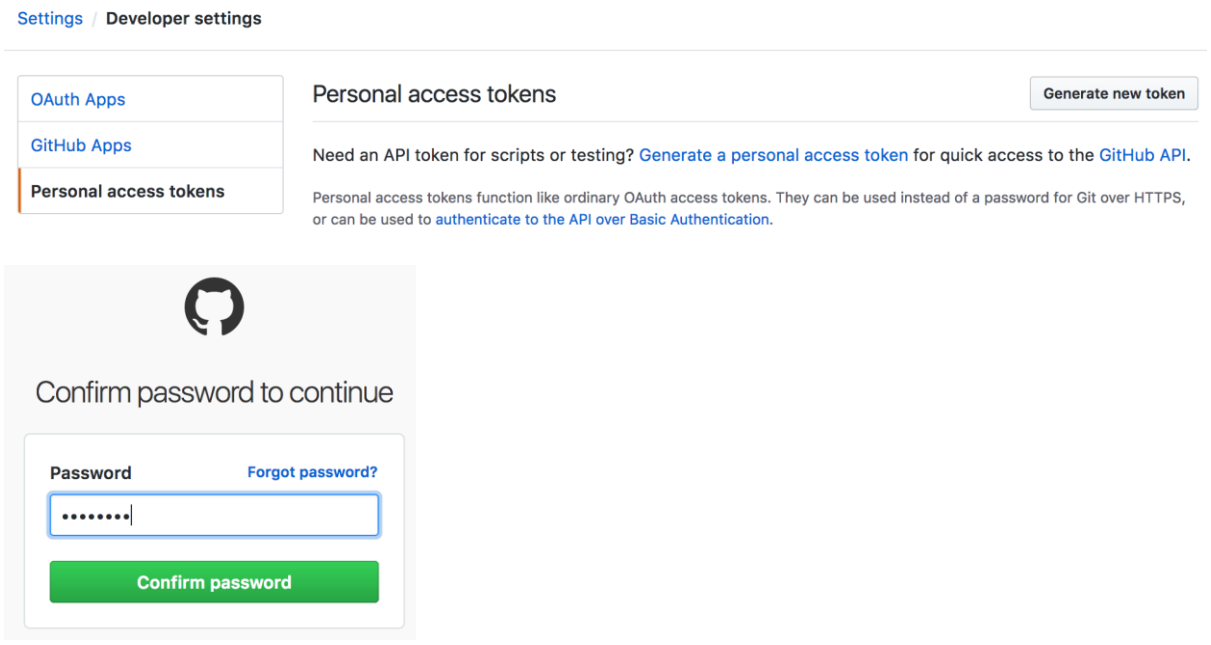
Select Developer settings



Select personal access tokens



Generate new token



New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Token description

Token for remote DevOps tooling

What's this token for?

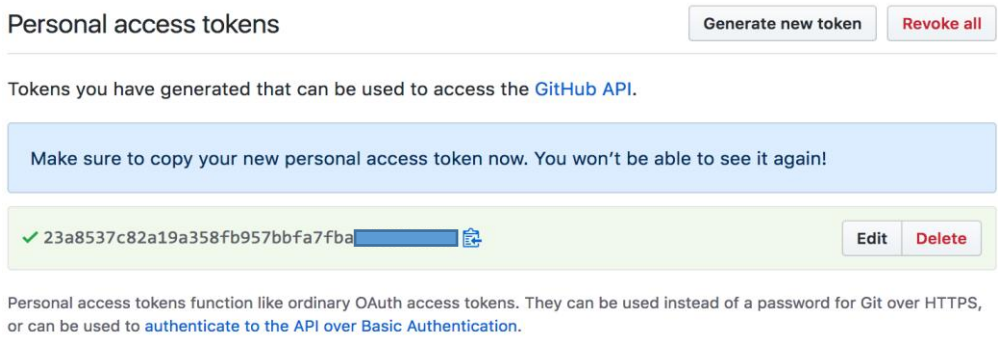
Select scopes

Scopes define the access for personal tokens. Read more about OAuth scopes.

repo	Full control of private repositories
repo:status	Access commit status
repo_deployment	Access deployment status
public_repo	Access public repositories
repo:invite	Access repository invitations
admin:org	Full control of orgs and teams
write:org	Read and write org and team membership
read:org	Read org and team membership
admin:public_key	Full control of user public keys
write:public_key	Write user public keys
read:public_key	Read user public keys
admin:repo_hook	Full control of repository hooks
write:repo_hook	Write repository hooks
read:repo_hook	Read repository hooks
admin:org_hook	Full control of organization hooks
gist	Create gists
notifications	Access notifications
user	Update all user data
read:user	Read all user profile data
user:email	Access user email addresses (read-only)
user:follow	Follow and unfollow users
delete_repo	Delete repositories
admin:gpg_key	Full control of user gpg keys (Developer Preview)
write:gpg_key	Write user gpg keys
read:gpg_key	Read user gpg keys

Generate token Cancel

Hit Generate token

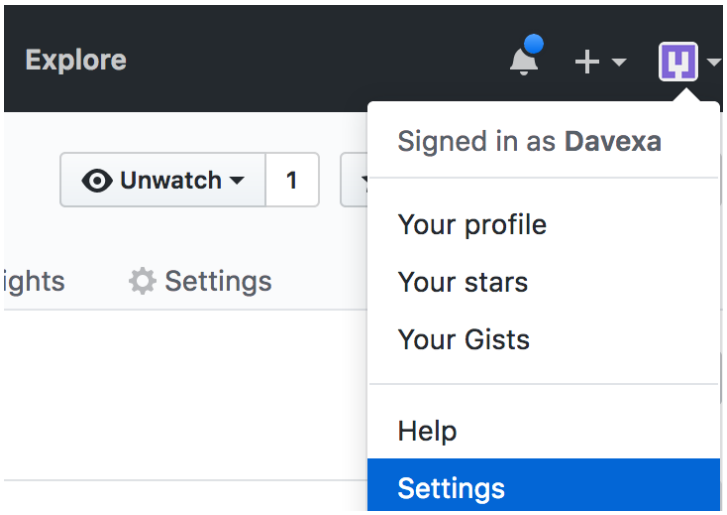


Copy and paste the token for later use.

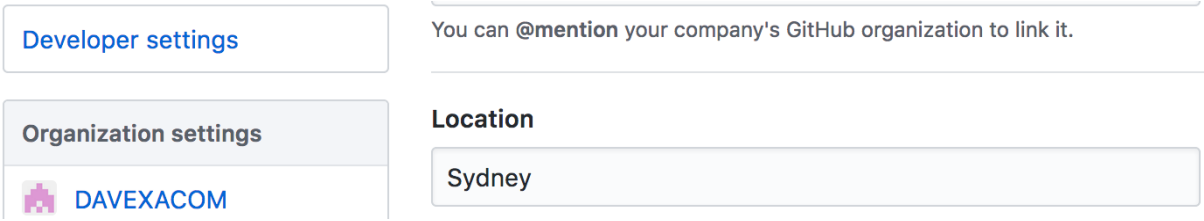
OAuth Apps – Register a new application

Register a new OAuth application

Developer Settings – Set up OAuth

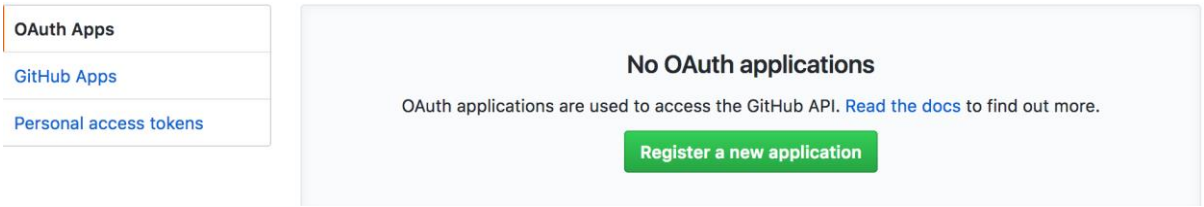


Select Developer settings



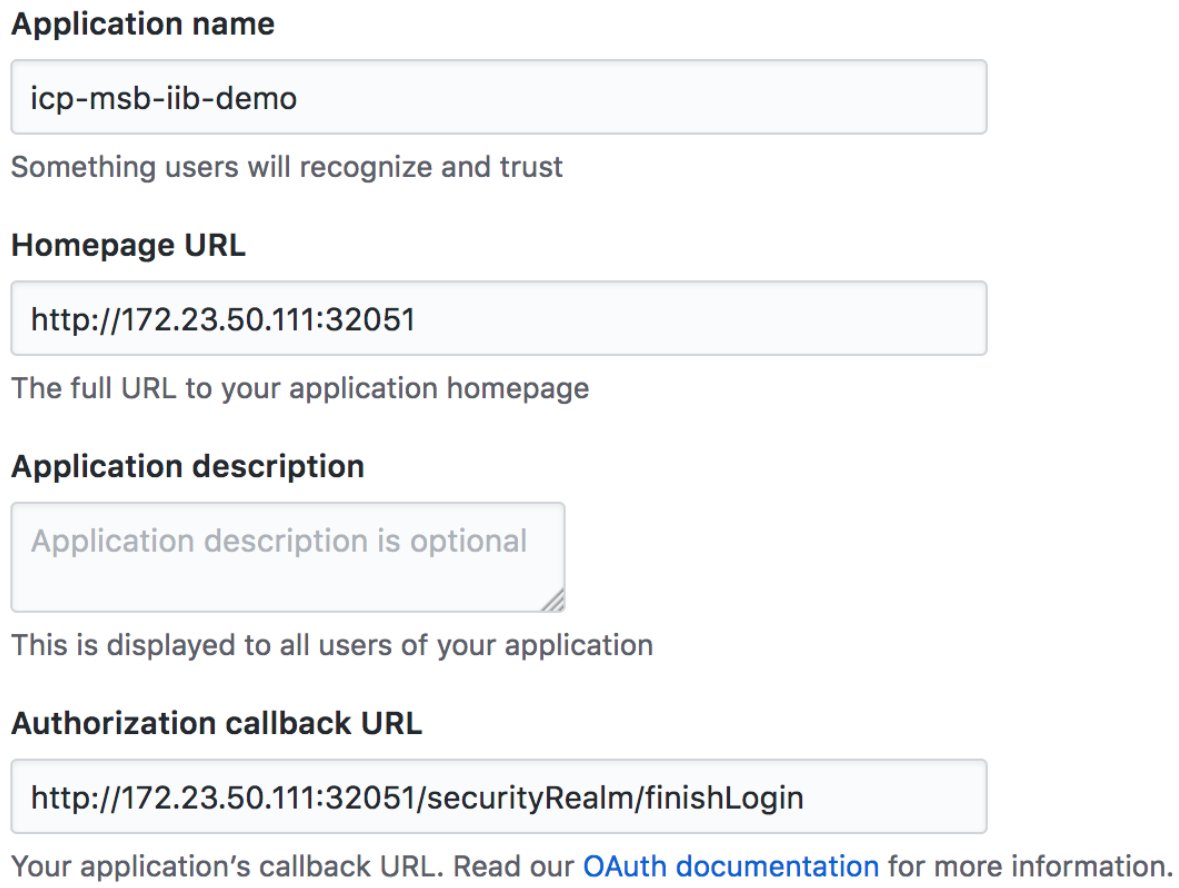
OAuth Apps – Register a new application

Register a new OAuth application

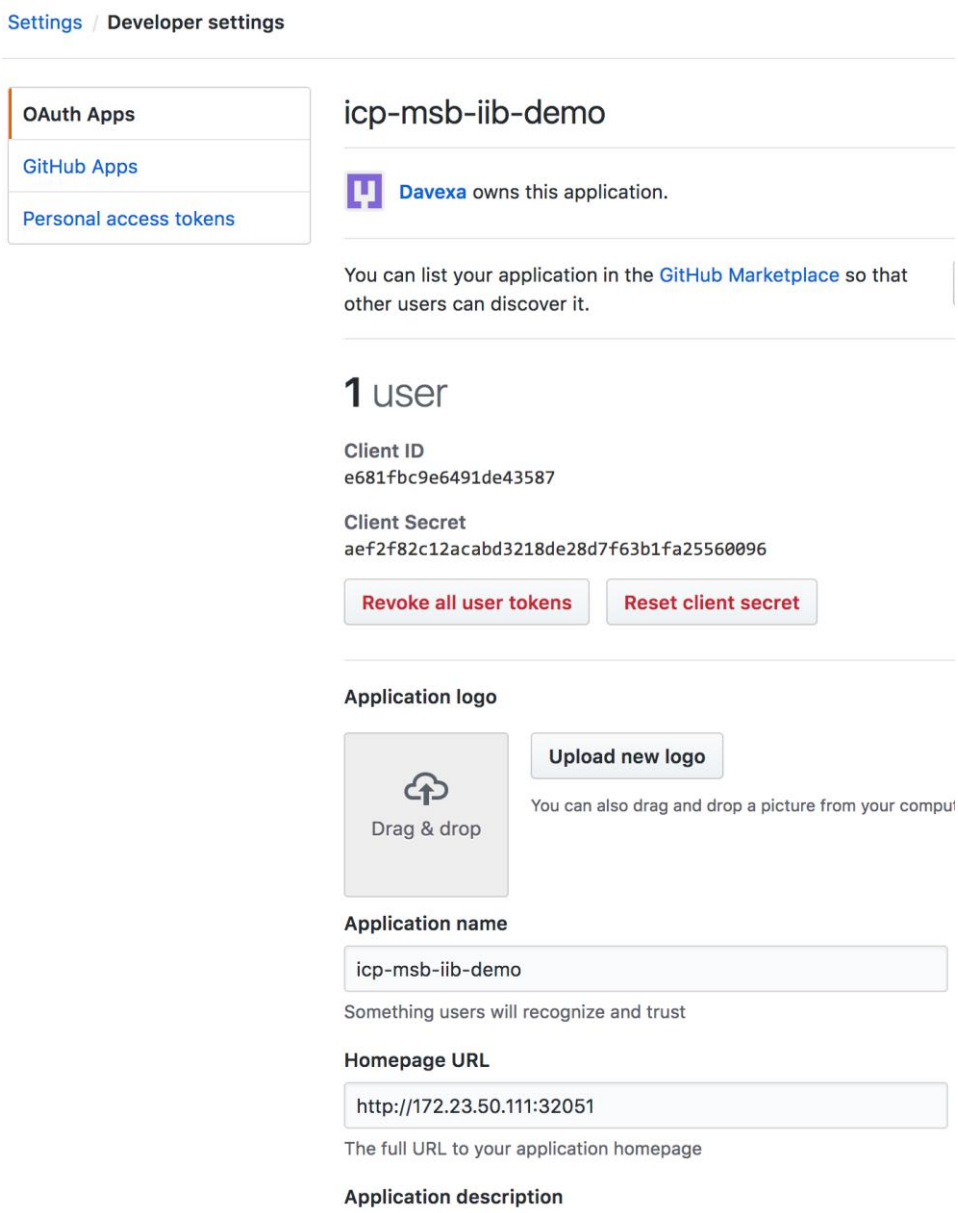


Important! Pick a port number for the pipeline. You will use it when you create the pipeline. (you may not get it right first time as the port might be in use when you create the pipeline. If you have to use a different port at pipeline creation time. You will need to come back here and change this port in Github OAuth application. So do not close this window.

Lets assume 32051 for now!



Click the green register application buttin

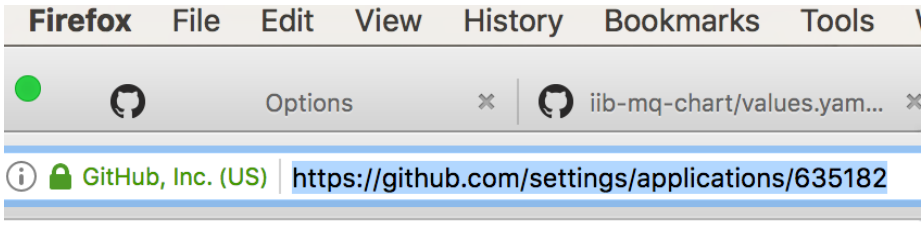


Capture OAuth Client ID and Secret

Copy and Paste the Client ID – you'll need it later.

Copy and Paste the Client Secret – you'll need it later.

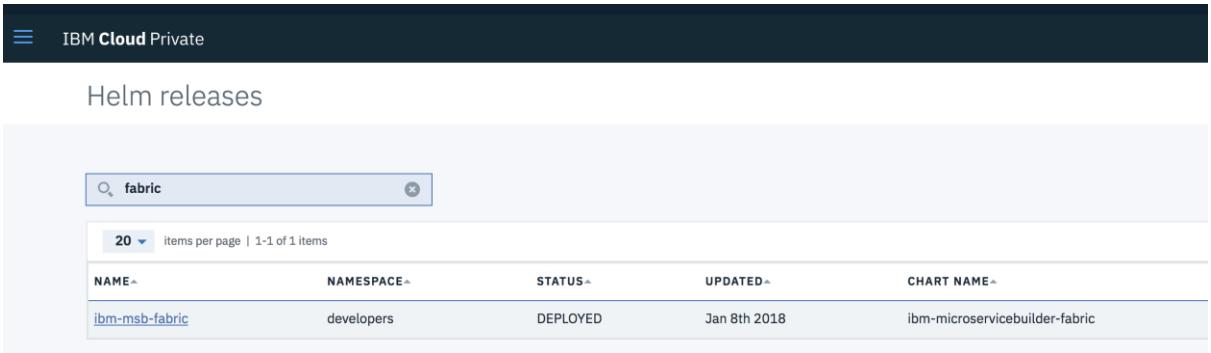
Copy of the Oauth Application Github URL from the Browser URL bar as well.



IBM Cloud Private Console – Check Micro Service Builder Fabric Service

The Micro Service Builder Pipeline pre-reqs that a Micro Service Builder Fabric service has been deployed into the ICP instance in the same namespace that your MSB pipeline (Jenkins) will be deployed.

From the ICP Console select Workloads->Helm Releases and filter on “fabric”



If there is no helm release deploy for ibm-microsservicebuilder-fabric showing as DEPLOYED in the namespace that you will deploy your pipeline talk to you ICP administrator about getting one set up.

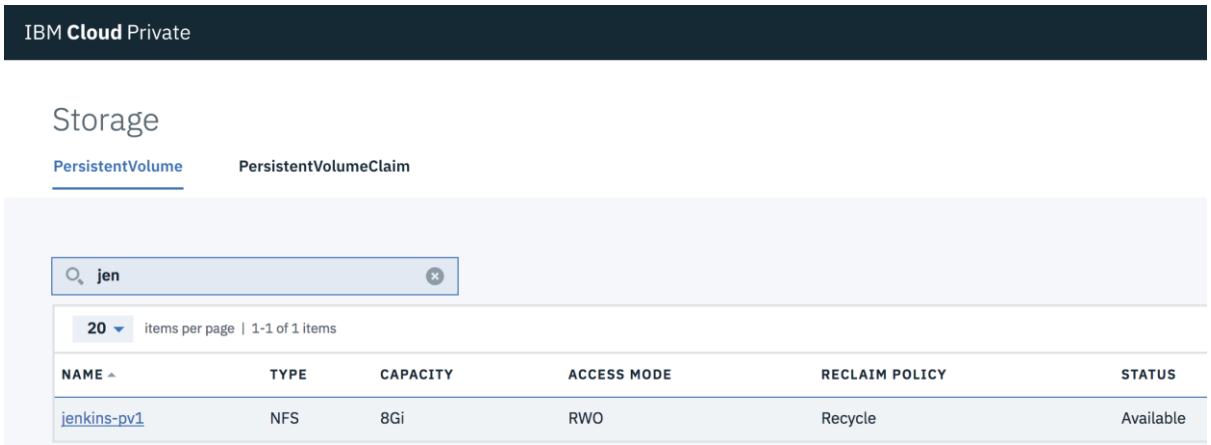
IBM Cloud Private Console – Check Jenkins Persistent Volume

Depending on the nature of your Jenkins set up you may want to use a persistent store. In these examples I don’t use one. However, below shows you have to check if a persistent store for Jenkins has already been set up.

Log into the console

Select Platform->Storage

Filter on jenkins



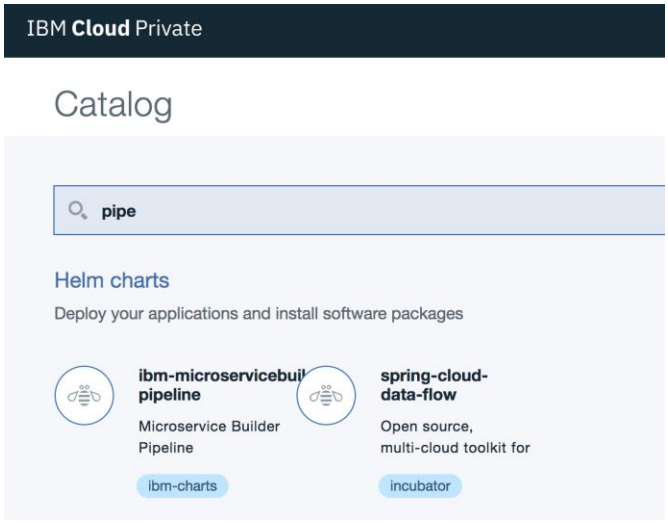
If there is no persistent volume that can be identified for use by jenkins then either create one or ask the ICP administrator.

IBM Cloud Private Console – Deploy MicroServiceBuild Pipeline

Log into the console

Select Catalog

Filter on pipe



Select ibm-microservicebuilder pipeline

Configure the ibm-microservicebuilder pipeline helm chart

Hit the configure button

Set the Github settings

Repos Pattern is .*

OAuth.User and Admins are your Github user name

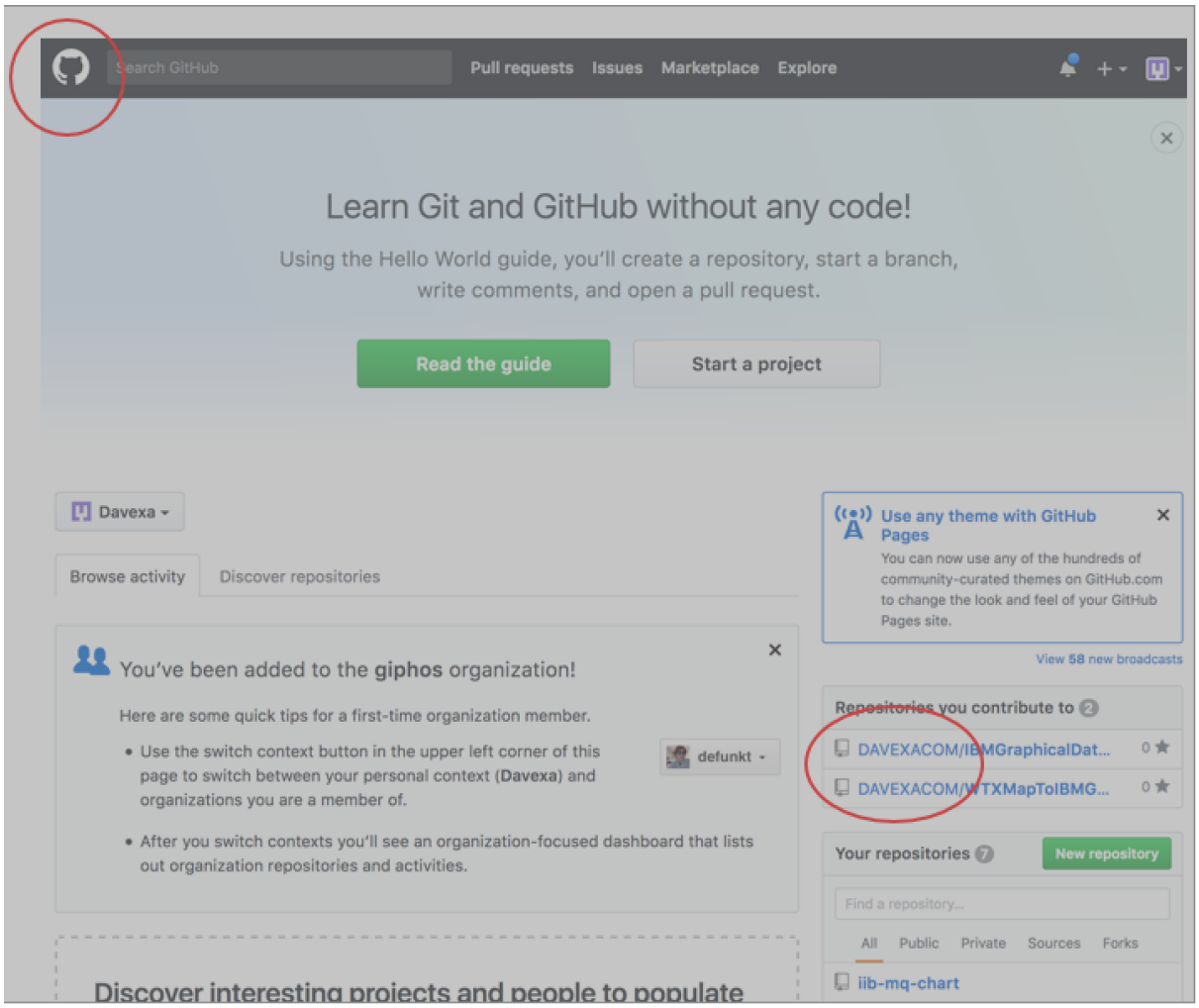
OAuth.Token is the personal access token

App.id is client ID from the OAuth registration in github

App.secret is client secret from the OAuth registration in github

Orgs is the Github organization

Note: To ensure you use the Github organization and not the GitHub user name. From a browser on which you are logged into github click on the silhouette of the cat icon. See below



OAuth Apps

GitHub Apps

Personal access tokens

icp-msb-iib-demo

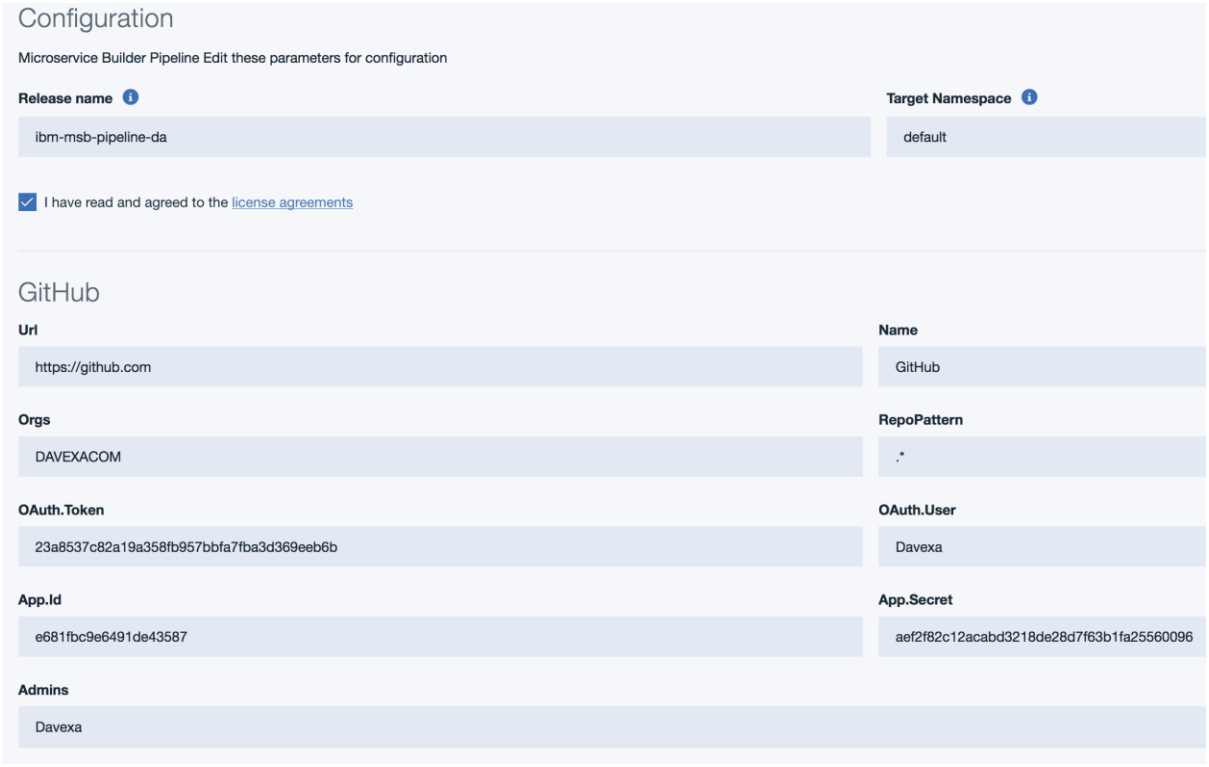
Davexa owns this application.

You can list your application in the GitHub Marketplace so that other users can discover it.

1 user

Client ID
`e681fbc9e6491de43587`

Client Secret
`aef2f82c12acabd3218de28d7f63b1fa25560096`



Pipeline Settings

Registry.URL is icpcluster.icp:8500/default

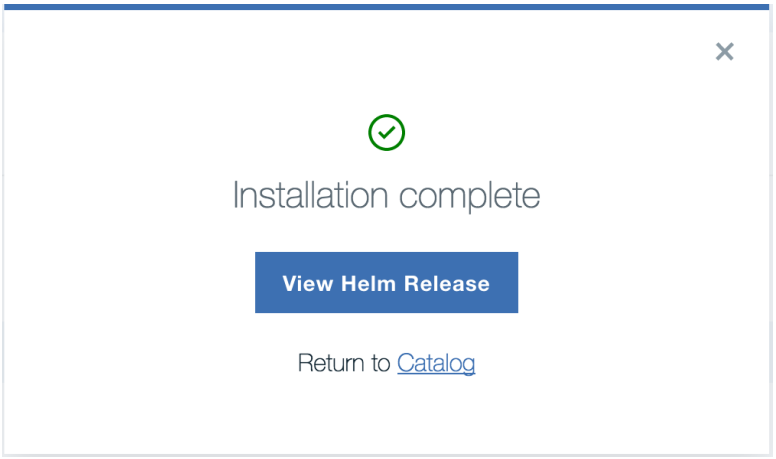
Master Settings

nodePort must be a port that is available, try 32050, 32051 etc

If you pick a port that is in use. You will be informed with a message top of screen when you attempt the install (see below). The Helm release will have failed but the release of the name you provided will have been created. You'll need to open a second browser tab go and delete it via ICP->Workloads->Helm releases before changing the nodePort number and trying the install again.

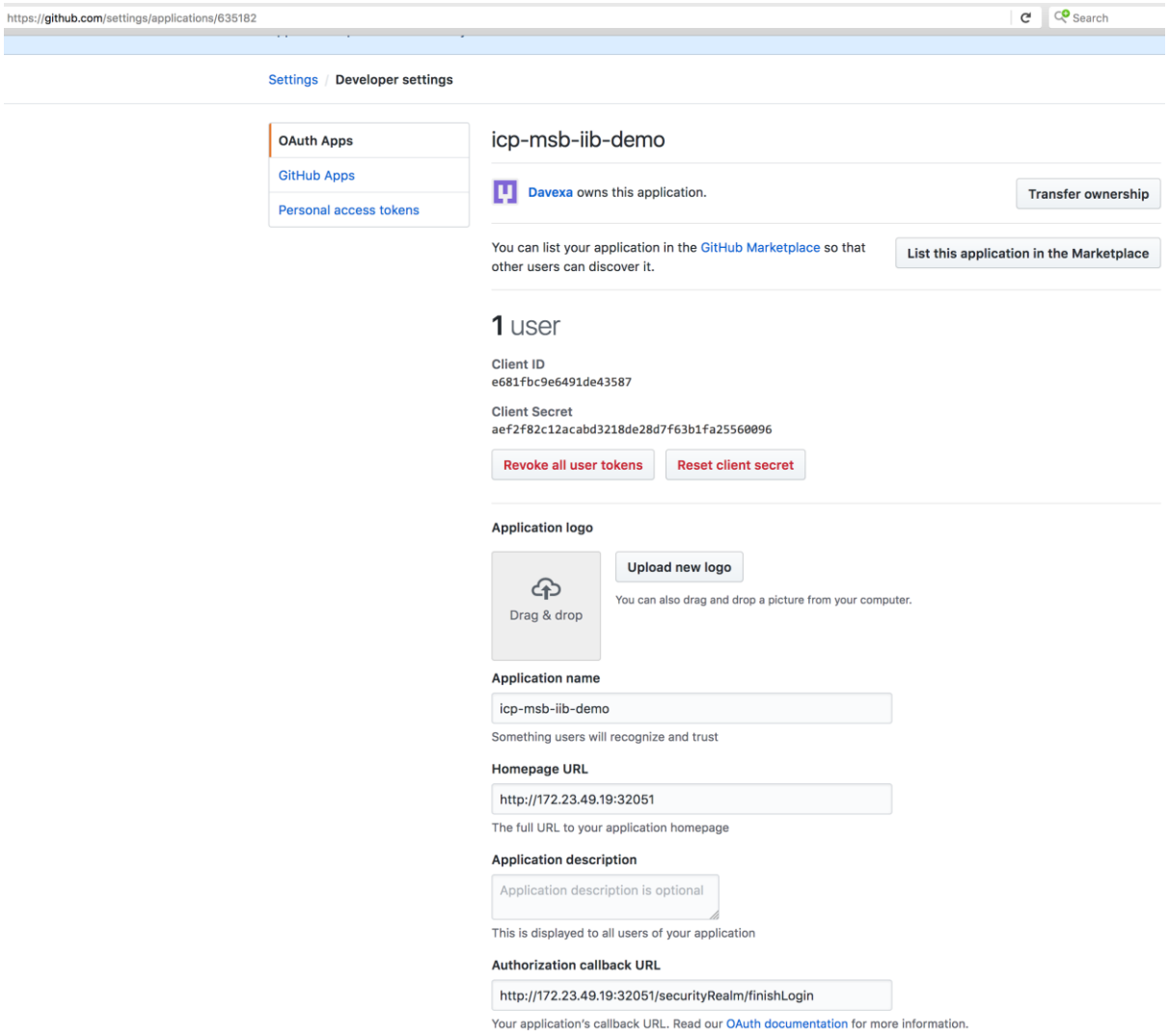
Although we check on a persistent volume for Jenkins we don't have to use it. I haven't used one in this example.

Leave everything else to default and hit install.



Check the Helm release to ensure it worked. If you chose a port that was already in use remember the Helm release will have failed but the release of the name you provided will have been created. You'll need to open a second browser tab go and delete it via ICP->Workloads->Helm releases before changing the nodePort number and trying the install again.

Important! Now go back to GitHub OAuth and ensure you have the correct the port number 32051 (or whatever it turned out to be in this example) as we took a punt on the port being available – see below.



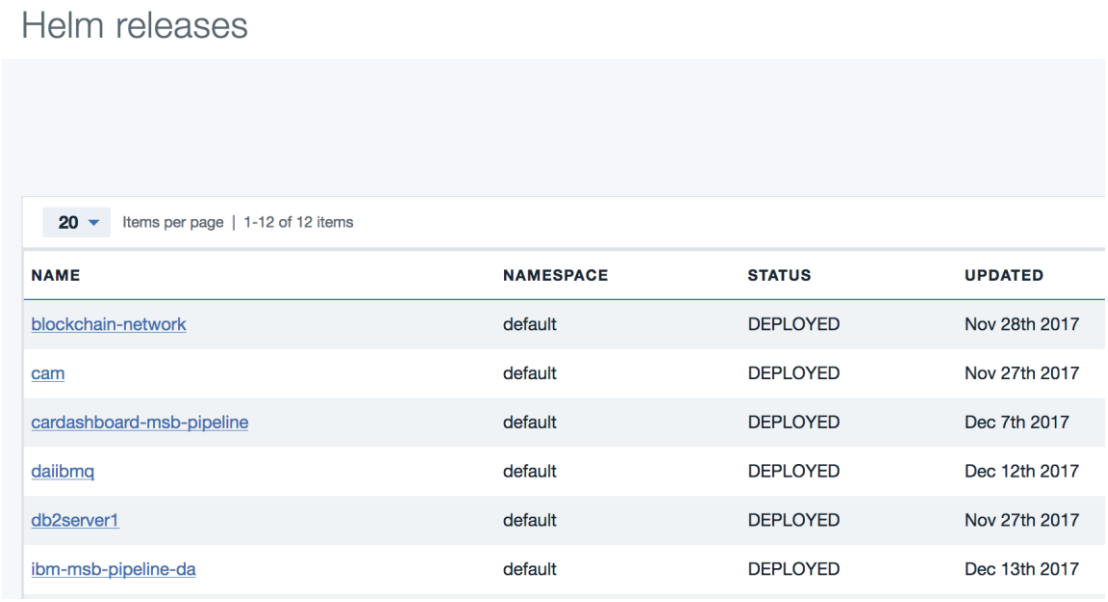
Note: Port number appears in 2 places. Homepage URL and Authorization call back

Then hit the update button.

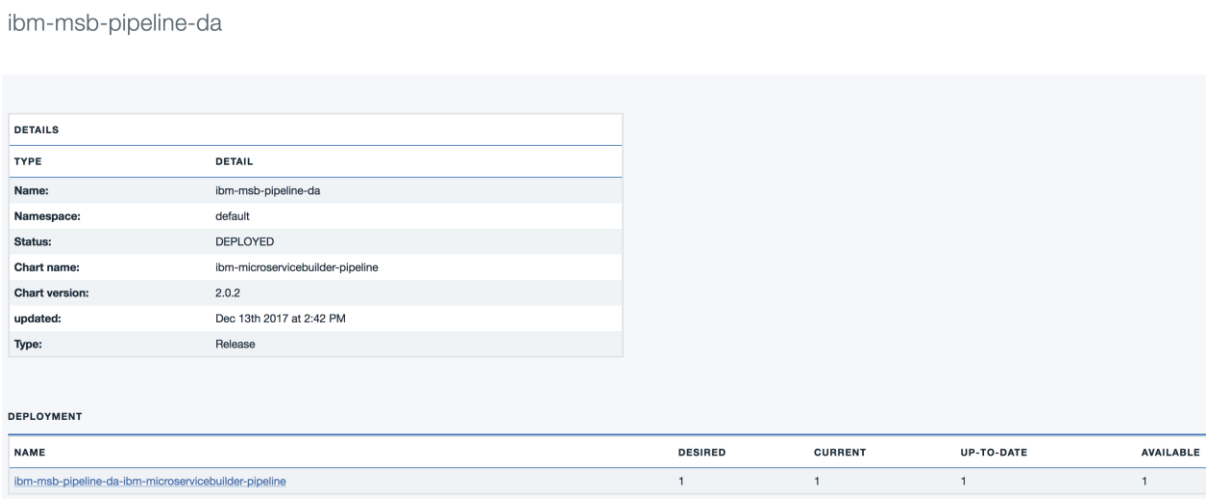
Your Github port will now match the icp msb pipeline parameters.

Click on View Helm Release

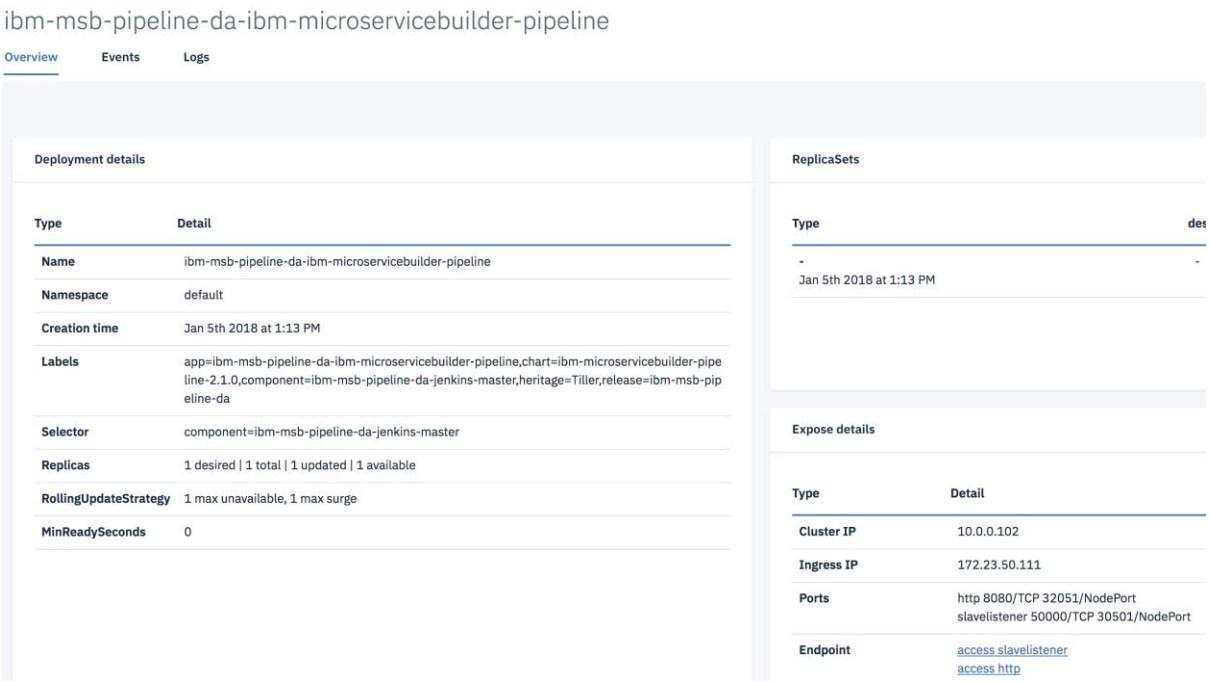
Review the microservice builder pipeline helm release



Select ibm-msb-pipeline-da



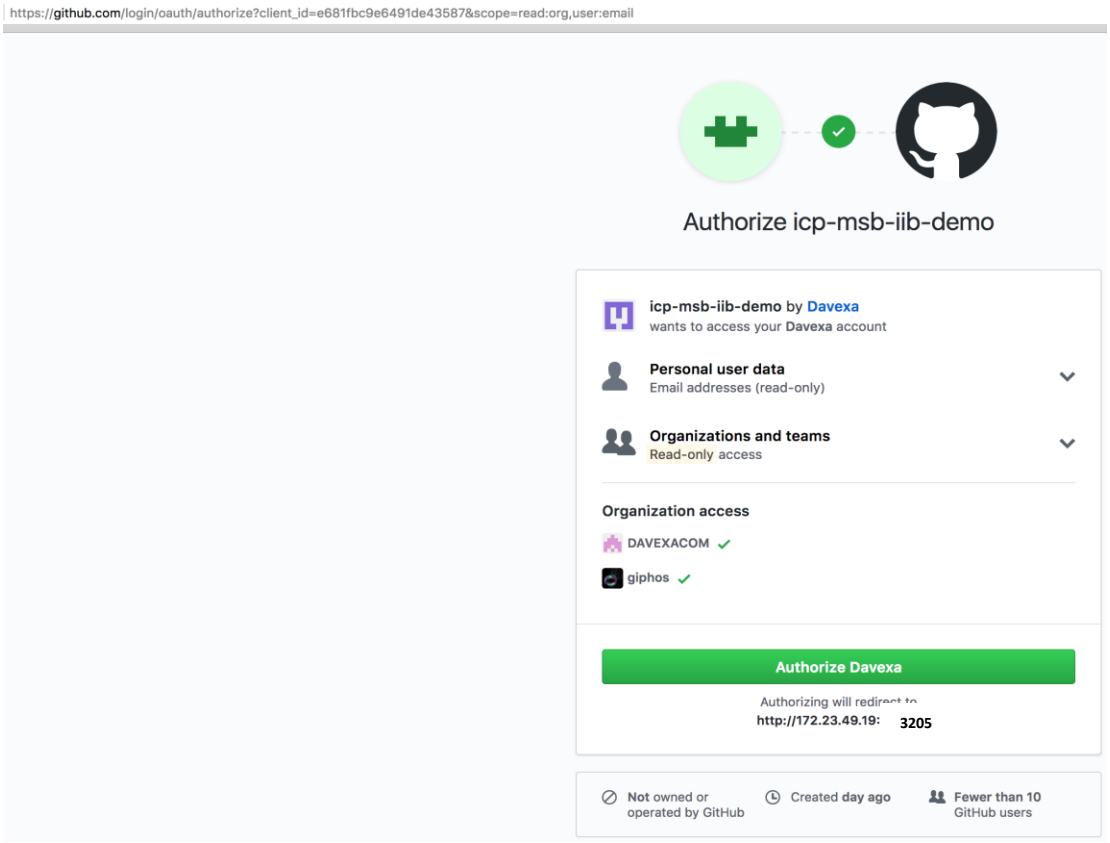
Select the Deployment link



Authorize the ibm-msb-pipeline-da service to access github

Click on access http link

Note: you may not get this Authorize if you have previously Authoried Davexa (in this case) to access the organization repository for OAuth

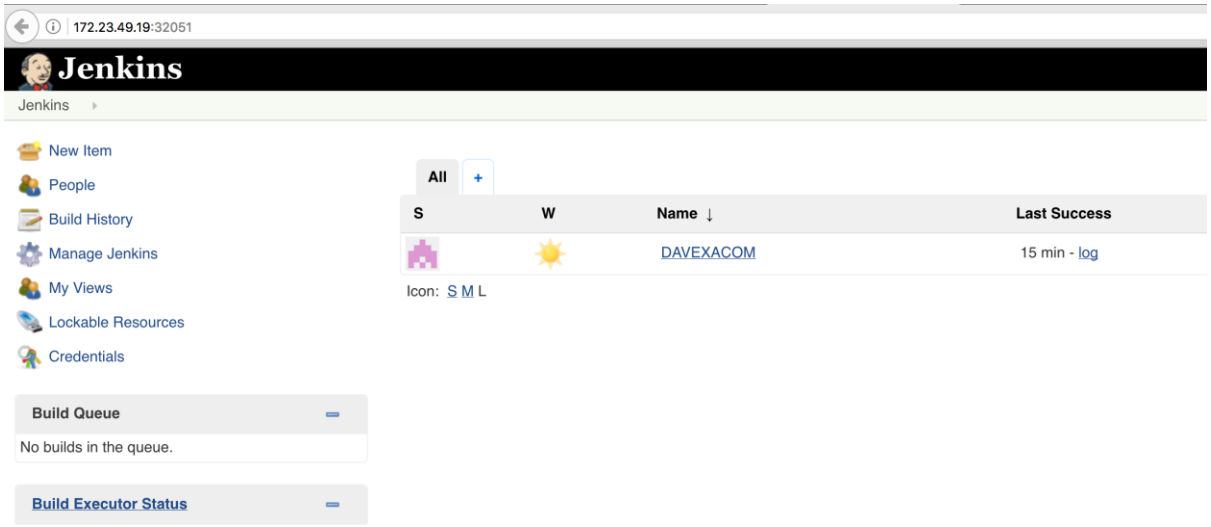


If when you hit the agree Authorize Davexa you get the following

Error

Watch out for the port 32037 mismatch to the 32051! You'll need to go back and esure the Git and ICP MSB pipeline have matched ports as described prevously

If you have you ports correctly matched you should get the Jenkins console.

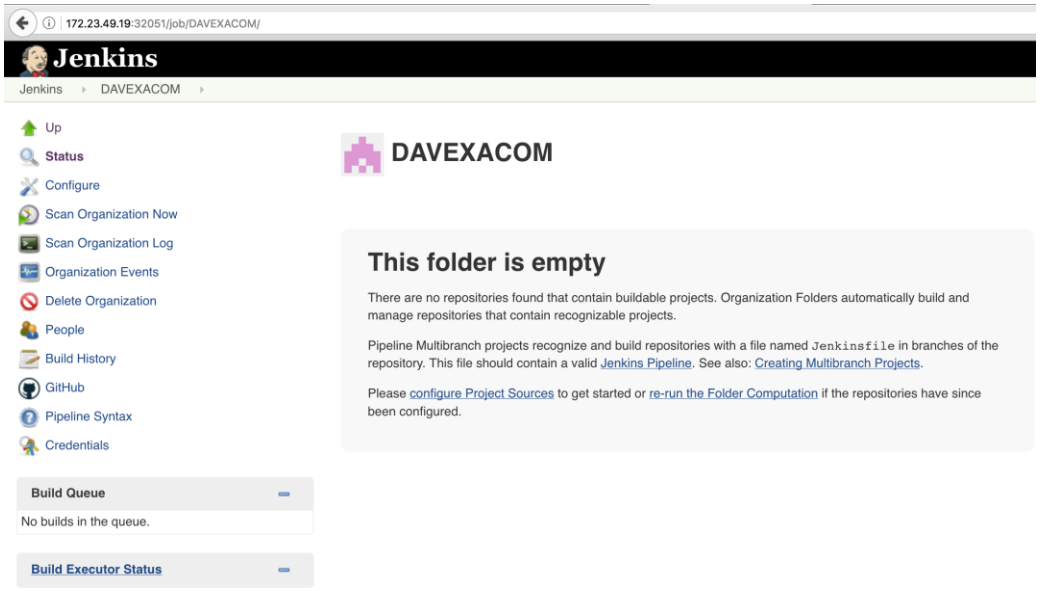


IBM Cloud Private – Jenkins

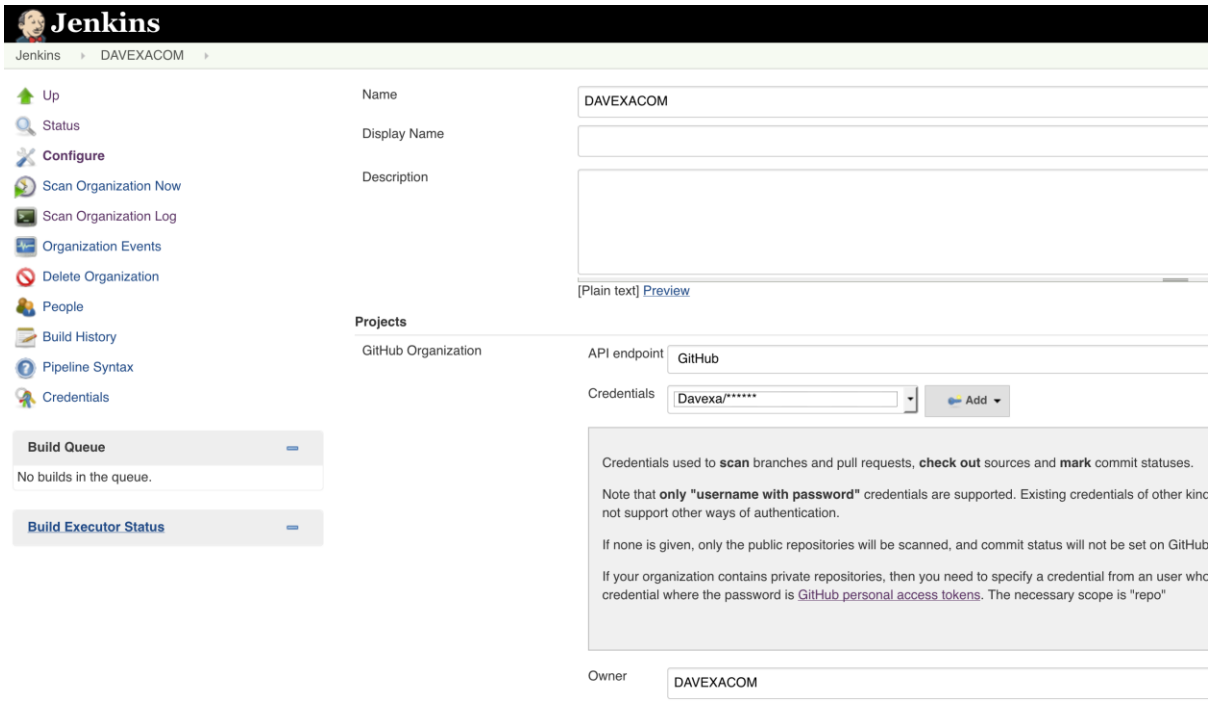
Checking Jenkins connection to Github

Click on the organization name DAVEXACOM in this example

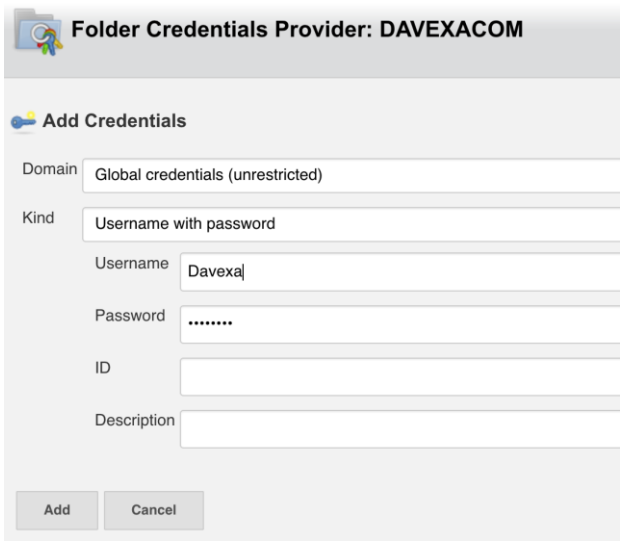
If you get a screen like below it means your repository is not structured as Jenkins expects. i.e. it does not have a Jenkinsfile perhaps. Or you have a credentials problem.



If you get the above you should check the scan organization log. **If it's a credentials problem and you can't work out why… then you can go into Configure.**

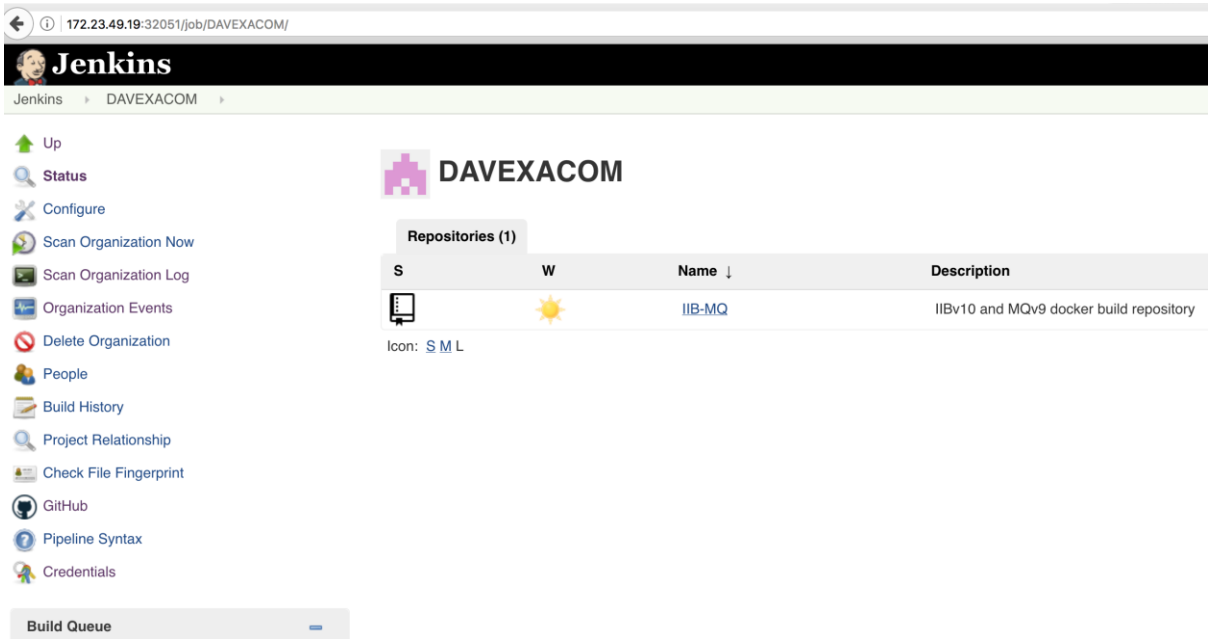


hit Add next to Credentials and select the organization



Add these credentials and make sure they are selected in the Configure screen

If all is correct it will look something like this.



Before Starting the Jenkins build process

You may want to change the values.yaml file in the GitHub repository in order to customize the Queue Manager Name, IIB Node name etc that will be created in the runtime container by the Jenkins build.

Branch: master ▾ **IIB-MQ** / chart / iibmq / **values.yaml**

Davexa Add files via upload

1 contributor

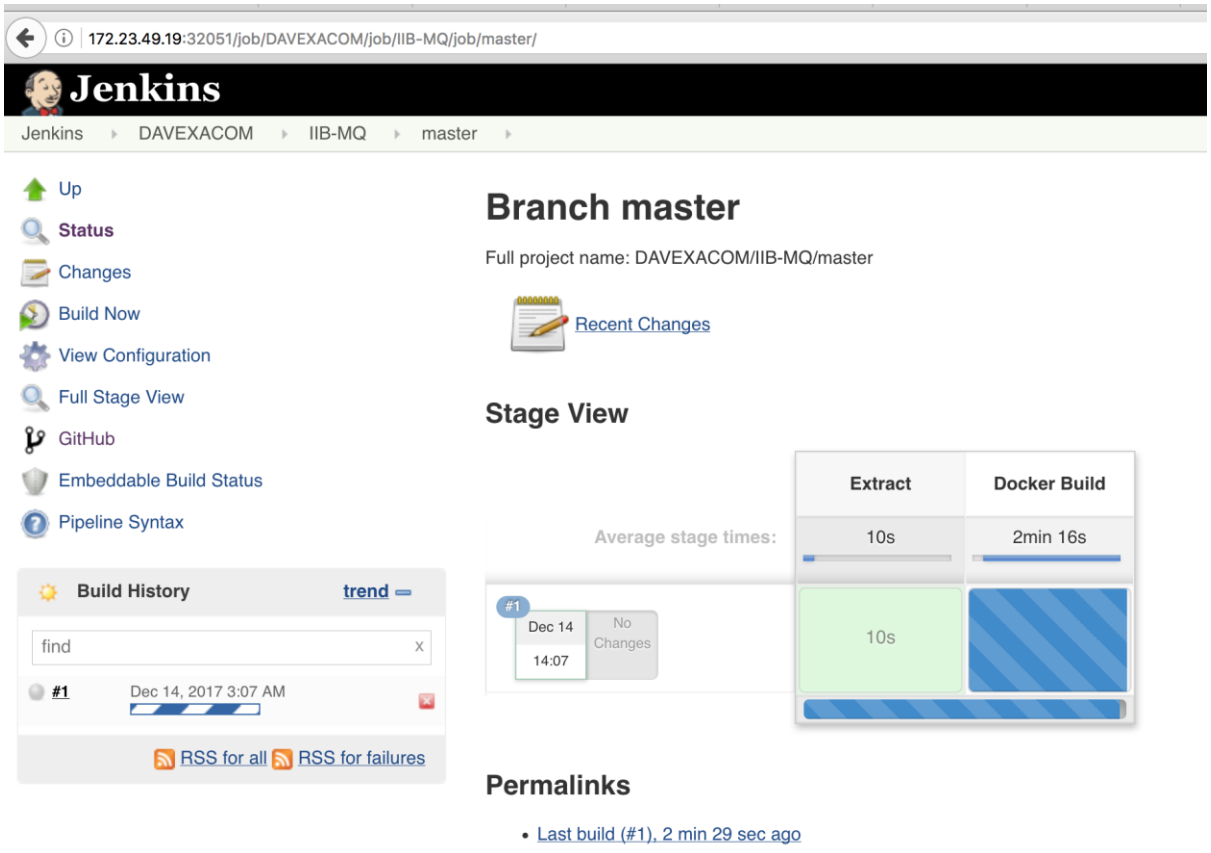
57 lines (50 sloc) | 1.97 KB | Raw | Blame

```
license: "accept"

image:
  # repository is the container repository to use, which must contain IBM MQ Advanced for Developers
  repository: icpcluster.icp:8500/default/iib-mq
  # tag is the tag to use for the container repository
  tag: latest
  # pullSecret is the secret to use when pulling the image from a private registry
  pullSecret:
  # pullPolicy is either IfNotPresent or Always (https://kubernetes.io/docs/concepts/containers/images/)
  pullPolicy: IfNotPresent

# persistence section specifies persistence settings which apply to the whole chart
persistence:
  # enabled is whether to use Persistent Volumes or not
  enabled: false
  # useDynamicProvisioning is whether or not to use Storage Classes to dynamically create Persistent Volumes
  useDynamicProvisioning: true

# dataPVC section specifies settings for the main Persistent Volume Claim, which is used for data in /var/mqm
dataPVC:
  # name sets part of the name for this Persistent Volume Claim
  name: "data"
  ## storageClassName is the name of the Storage Class to use, or an empty string for no Storage Class
  storageClassName: ""
  ## size is the minimum size of the Persistent Volume
  size: 2Gi

service:
  name: iib-mq
  type: NodePort

resources:
  limits:
    cpu: 1024m
    memory: 1024Mi
  requests:
    cpu: 1024m
    memory: 1024Mi

# queueManager section specifies settings for the MQ Queue Manager
queueManager:
  # name allows you to specify the name to use for the queue manager.  Defaults to the Helm release name.
  name: icpqm1
```

I changed the queueManager->name to icpDA1 for example

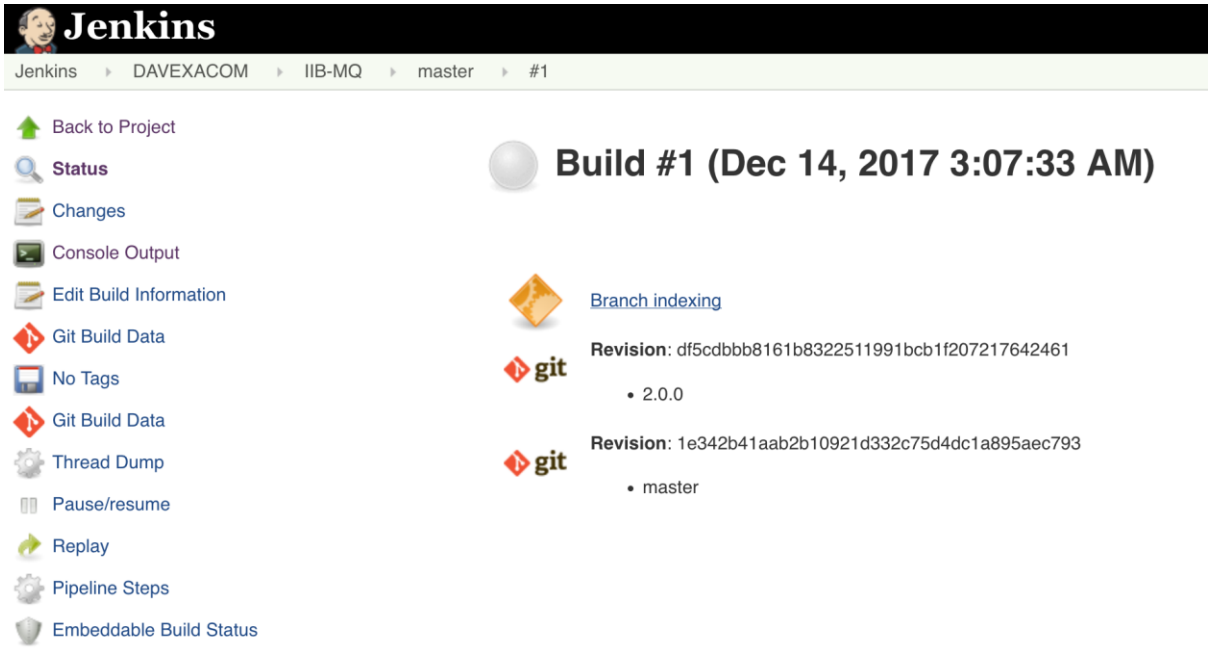
Starting the Jenkins build process

With no Github webhook in place to publish to Jenkins to tell it to start a build we will use the Scan Organization Now button.

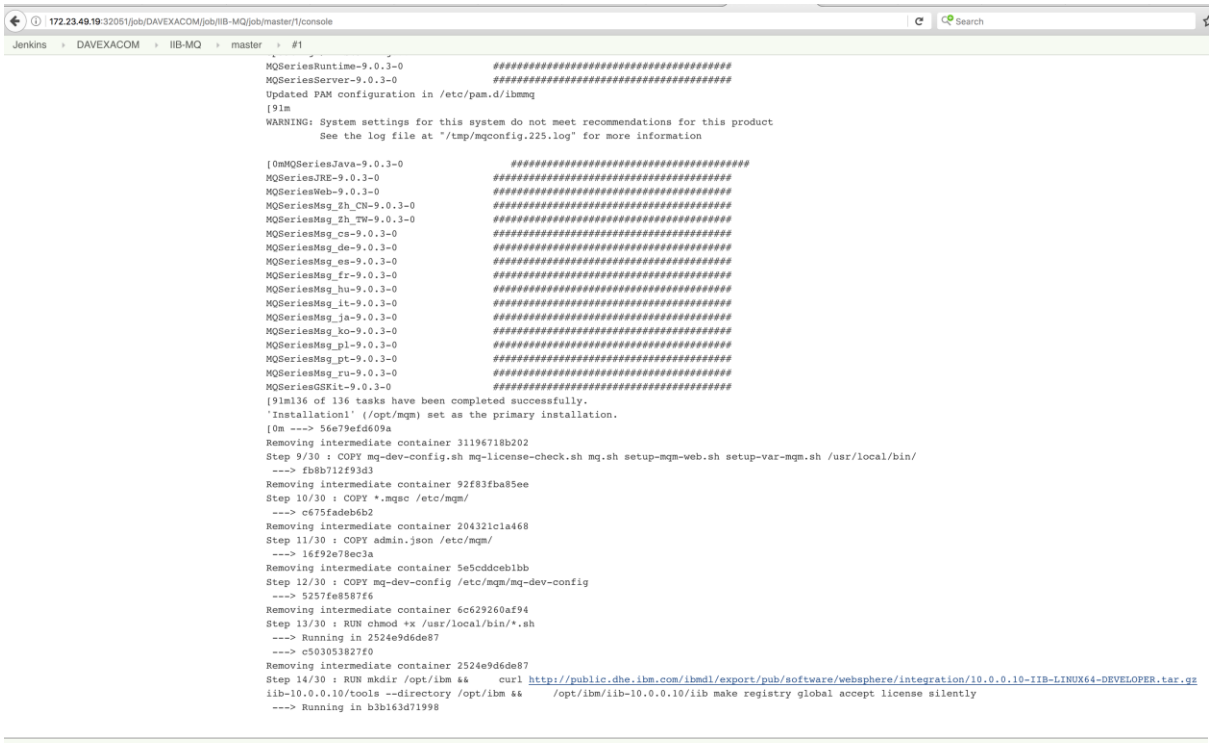


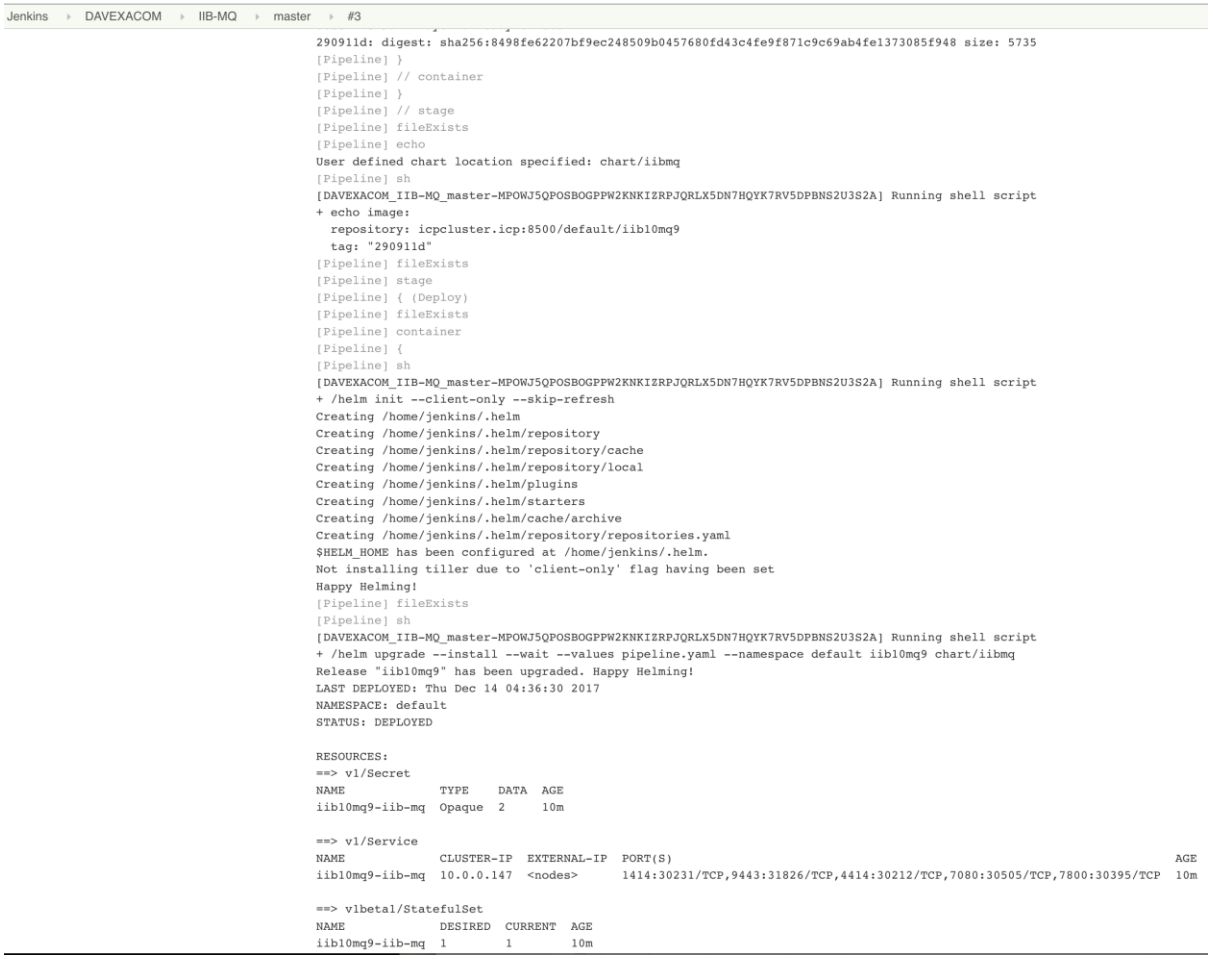
Following the Jenkins build logs

Click on the #1 in Build history



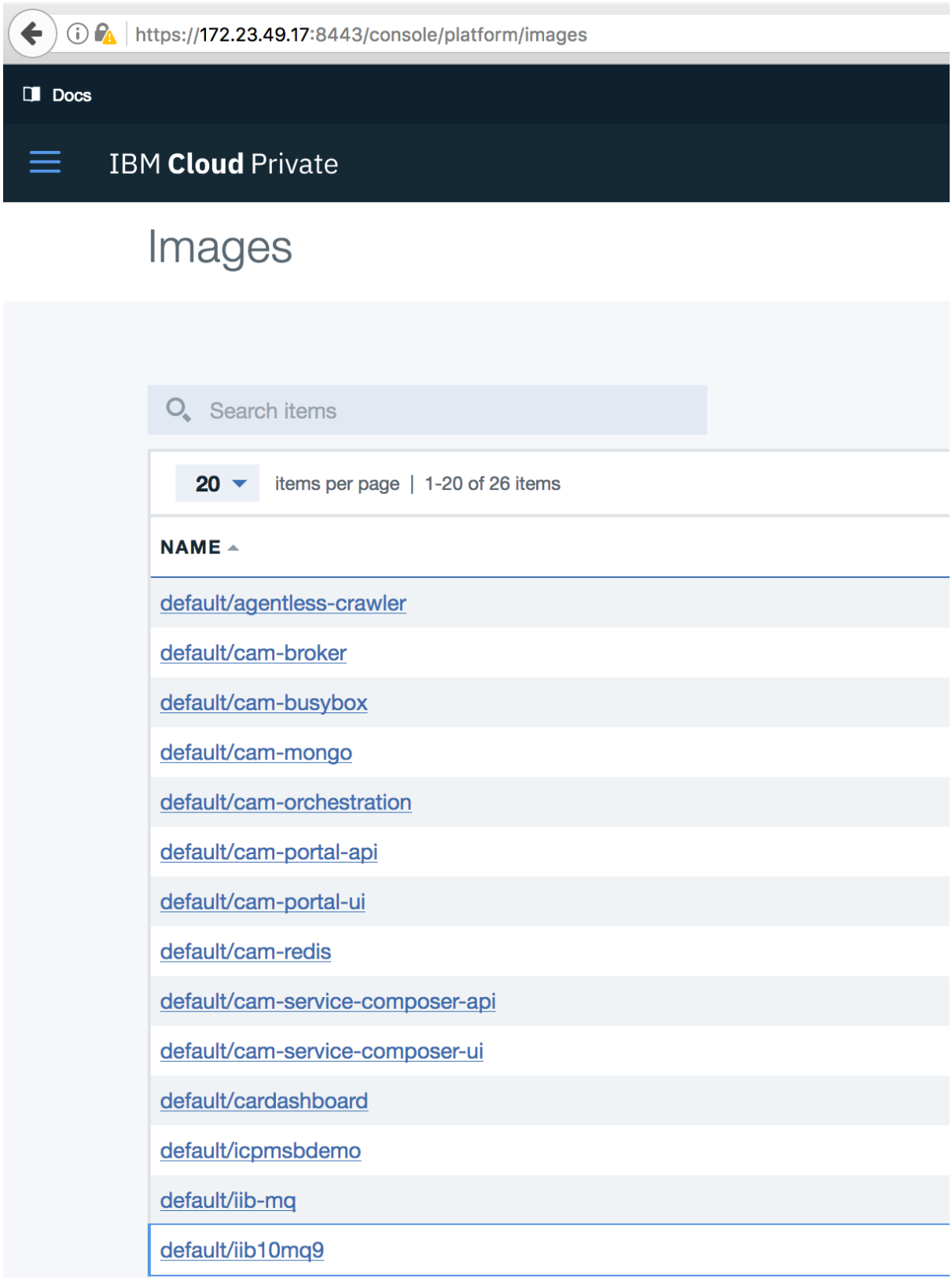
Click on Console Output


172.23.49.19:32051/job/DAVEXACOM/job/IIB-MQ/job/master/1/console
Search
Jenkins ▸ DAVEXACOM ▸ IIB-MQ ▸ master ▸ #1
MQSeriesRuntime-9.0.3-0 ##########
MQSeriesServer-9.0.3-0 ##########
Updated PAM configuration in /etc/pam.d/ibmmq
[91m
WARNING: System settings for this system do not meet recommendations for this product
See the log file at "/tmp/mqconfig.225.log" for more information
[0mMQSeriesJava-9.0.3-0 ##########
MQSeriesJRE-9.0.3-0 ##########
MQSeriesWeb-9.0.3-0 ##########
MQSeriesMsg_Zh_CN-9.0.3-0 ##########
MQSeriesMsg_Zh_TW-9.0.3-0 ##########
MQSeriesMsg_cs-9.0.3-0 ##########
MQSeriesMsg_de-9.0.3-0 ##########
MQSeriesMsg_es-9.0.3-0 ##########
MQSeriesMsg_fr-9.0.3-0 ##########
MQSeriesMsg_hu-9.0.3-0 ##########
MQSeriesMsg_it-9.0.3-0 ##########
MQSeriesMsg_ja-9.0.3-0 ##########
MQSeriesMsg_ko-9.0.3-0 ##########
MQSeriesMsg_pl-9.0.3-0 ##########
MQSeriesMsg_pt-9.0.3-0 ##########
MQSeriesMsg_ru-9.0.3-0 ##########
MQSeriesGSKit-9.0.3-0 ##########
[91m136 of 136 tasks have been completed successfully.
'Installation1' ((/opt/mqm) set as the primary installation.
[0m ---> 56e79efd609a
Removing intermediate container 31196718b202
Step 9/30 : COPY mq-dev-config.sh mq-license-check.sh mq.sh setup-mqm-web.sh setup-var-mqm.sh /usr/local/bin/
---> fb8b712f93d3
Removing intermediate container 92f83fba85ee
Step 10/30 : COPY *.mqsc /etc/mqm/
---> c675fadeb6b2
Removing intermediate container 20432lc1a468
Step 11/30 : COPY admin.json /etc/mqm/
---> 16f92e78ec3a
Removing intermediate container 5e5cddceb1bb
Step 12/30 : COPY mq-dev-config /etc/mqm/mq-dev-config
---> 5257fe8587f6
Removing intermediate container 6c629260af94
Step 13/30 : RUN chmod +x /usr/local/bin/*.sh
---> Running in 2524e9d6de87
---> c503053827f0
Removing intermediate container 2524e9d6de87
Step 14/30 : RUN mkdir /opt/ibm && curl http://public.dhe.ibm.com/ibmdl/export/pub/software/websphere/integration/10.0.0.10-IIB-LINUX64-DEVELOPER.tar.gz
iib-10.0.0.10/tools --directory /opt/ibm && /opt/ibm/iib-10.0.0.10/iib make registry global accept license silently
---> Running in b3b163d71998


Jenkins ▸ DAVEXACOM ▸ IIB-MQ ▸ master ▸ #3
290911d: digest: sha256:8498fe62207bf9ec248509b0457680fd43c4fe9f871c9c69ab4fe1373085f948 size: 5735
[Pipeline] }
[Pipeline] // container
[Pipeline] }
[Pipeline] // stage
[Pipeline] fileExists
[Pipeline] echo
User defined chart location specified: chart/iibmq
[Pipeline] sh
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ echo image:
repository: icpcluster.icp:8500/default/iib10mq9
tag: "290911d"
[Pipeline] fileExists
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] fileExists
[Pipeline] container
[Pipeline] {
[Pipeline] sh
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ /helm init --client-only --skip-refresh
Creating /home/jenkins/.helm
Creating /home/jenkins/.helm/repository
Creating /home/jenkins/.helm/repository/cache
Creating /home/jenkins/.helm/repository/local
Creating /home/jenkins/.helm/plugins
Creating /home/jenkins/.helm/starters
Creating /home/jenkins/.helm/cache/archive
Creating /home/jenkins/.helm/repository/repositories.yaml
$HELM_HOME has been configured at /home/jenkins/.helm.
Not installing tiller due to 'client-only' flag having been set
Happy Helming!
[Pipeline] fileExists
[Pipeline] sh
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ /helm upgrade --install --wait --values pipeline.yaml --namespace default iib10mq9 chart/iibmq
Release "iib10mq9" has been upgraded. Happy Helming!
LAST DEPLOYED: Thu Dec 14 04:36:30 2017
NAMESPACE: default
STATUS: DEPLOYED
RESOURCES:
==> v1/Secret
NAME TYPE DATA AGE
iib10mq9-iib-mq Opaque 2 10m
==> v1/Service
NAME CLUSTER-IP EXTERNAL-IP PORT(S) AGE
iib10mq9-iib-mq 10.0.0.147 <nodes> 1414:30231/TCP,9443:31826/TCP,4414:30212/TCP,7080:30505/TCP,7800:30395/TCP 10m
==> v1beta1/StatefulSet
NAME DESIRED CURRENT AGE
iib10mq9-iib-mq 1 1 10m

```
$HELM_HOME has been configured at /home/jenkins/.helm.
Not installing tiller due to 'client-only' flag having been set
Happy Helming!
[Pipeline] fileExists
[Pipeline] sh
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ /helm upgrade --install --wait --values pipeline.yaml --namespace default iib10mq9 chart/iibmq
Release "iib10mq9" has been upgraded. Happy Helming!
LAST DEPLOYED: Thu Dec 14 04:36:30 2017
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Secret
NAME             TYPE    DATA  AGE
iib10mq9-iib-mq  Opaque  2     10m

==> v1/Service
NAME             CLUSTER-IP  EXTERNAL-IP  PORT(S)                                                                  AGE
iib10mq9-iib-mq  10.0.0.147  <nodes>      1414:30231/TCP,9443:31826/TCP,4414:30212/TCP,7080:30505/TCP,7800:30395/TCP  10m

==> v1beta1/StatefulSet
NAME             DESIRED  CURRENT  AGE
iib10mq9-iib-mq  1        1        10m


[Pipeline] }
[Pipeline] // container
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // podTemplate
[Pipeline] End of Pipeline

Could not update commit status. Message: Server returned HTTP response code: 201, message: 'Created' for URL: https://api.github.com/repos/DAVEXACOM/IIB-MQ/statuses
/290911dc47be82503419de524a4caae5d903e037

Finished: SUCCESS
```

Checking ICP Image repository



IBM **Cloud** Private

Images / default/iib10mq9 /

default/iib10mq9

Overview

Image details

TYPE	DETAIL
Name	default/iib10mq9
Owner	default
Scope	global
Tags	

Checking ICP Helm Release

IBM **Cloud** Private

Helm releases

20 Items per page | 1-13 of 13 items

NAME	NAMESPACE	STATUS	UPDATED	CHART NAME
blockchain-network	default	DEPLOYED	Nov 28th 2017	ibm-blockchain-network
cam	default	DEPLOYED	Nov 27th 2017	ibm-cam-prod
cardashboard-msb-pipeline	default	DEPLOYED	Dec 7th 2017	cardashboard
daiibmq	default	DEPLOYED	Dec 12th 2017	iib-mq
db2server1	default	DEPLOYED	Nov 27th 2017	ibm-db2oltp-dev
ibm-msb-pipeline-da	default	DEPLOYED	Dec 13th 2017	ibm-microservicebuilder-pipeline
ibm-msb-pipeline	default	DEPLOYED	Dec 6th 2017	ibm-microservicebuilder-pipeline
ibm-msbf	default	DEPLOYED	Nov 28th 2017	ibm-microservicebuilder-fabric
icp-dp7	default	DEPLOYED	Dec 4th 2017	ibm-datapower-dev
icpmsbdemo	default	DEPLOYED	Dec 6th 2017	icpmsbdemo
iib-icp-100010	default	DEPLOYED	Dec 4th 2017	ibm-integration-bus-dev
iib10mq9	default	DEPLOYED	Dec 14th 2017	iib-mq

iib10mq9

DETAILS

TYPE	DETAIL
Name:	iib10mq9
Namespace:	default
Status:	DEPLOYED
Chart name:	iib-mq
Chart version:	1.1.0
updated:	Dec 14th 2017 at 3:36 PM
Type:	Release

SERVICE

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)
iib10mq9-iib-mq	10.0.0.147	<nodes>	1414:30231/TCP,9443:31826/TCP,4414:30212/TCP,7080:30505/TCP,7800:30395/TCP

STATEFULSET

NAME	DESIRED	CURRENT
iib10mq9-iib-mq	1	1

iib10mq9

DETAILS

TYPE	DETAIL
Name:	iib10mq9
Namespace:	default
Status:	DEPLOYED
Chart name:	iib-mq
Chart version:	1.1.0
updated:	Dec 14th 2017 at 3:36 PM
Type:	Release

SERVICE

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)
iib10mq9-iib-mq	10.0.0.147	<nodes>	1414:30231/TCP,9443:31826/TCP,4414:30212/TCP,7080:30505/TCP,7800:30395/TCP

STATEFULSET

NAME	DESIRED	CURRENT
iib10mq9-iib-mq	1	1

StatefulSet / iib10mq9-iib-mq / iib10mq9-iib-mq-0 /

iib10mq9-iib-mq-0

Overview Containers Events Logs

```
Starting syslog
----------------------------------------
Configuring db access
BIP8071I: Successful command completion.
Starting node icpnode1
Starting Switch Server
BIP8096I: Successful command initiation, check the system log to ensure that the component started without
----------------------------------------
Server mqweb started with process ID 459.
S starting Switch
Starting iibswitch, please wait...
iibswitch started.
BIP8071I: Successful command completion.
BIP8071I: Successful command completion.
BIP8096I: Successful command initiation, check the system log to ensure that the component started without
QMNAME(icpDA1)                                          STATUS(Running)
IBM MQ Queue Manager icpDA1 is now fully running
```

Connect MQ Console

ICP->Helm Releases->iib10mq9-iib-mq->services

SERVICE

NAME	CLUSTER-IP	EXTERNAL-IP	PORT(S)
iib10mq9-iib-mq	10.0.0.147	<nodes>	1414:30231/TCP,9443:31826/TCP,4414:30212/TCP,7080:30505/TCP,7800:30395/TCP

Services / iib10mq9-iib-mq /

iib10mq9-iib-mq

Overview

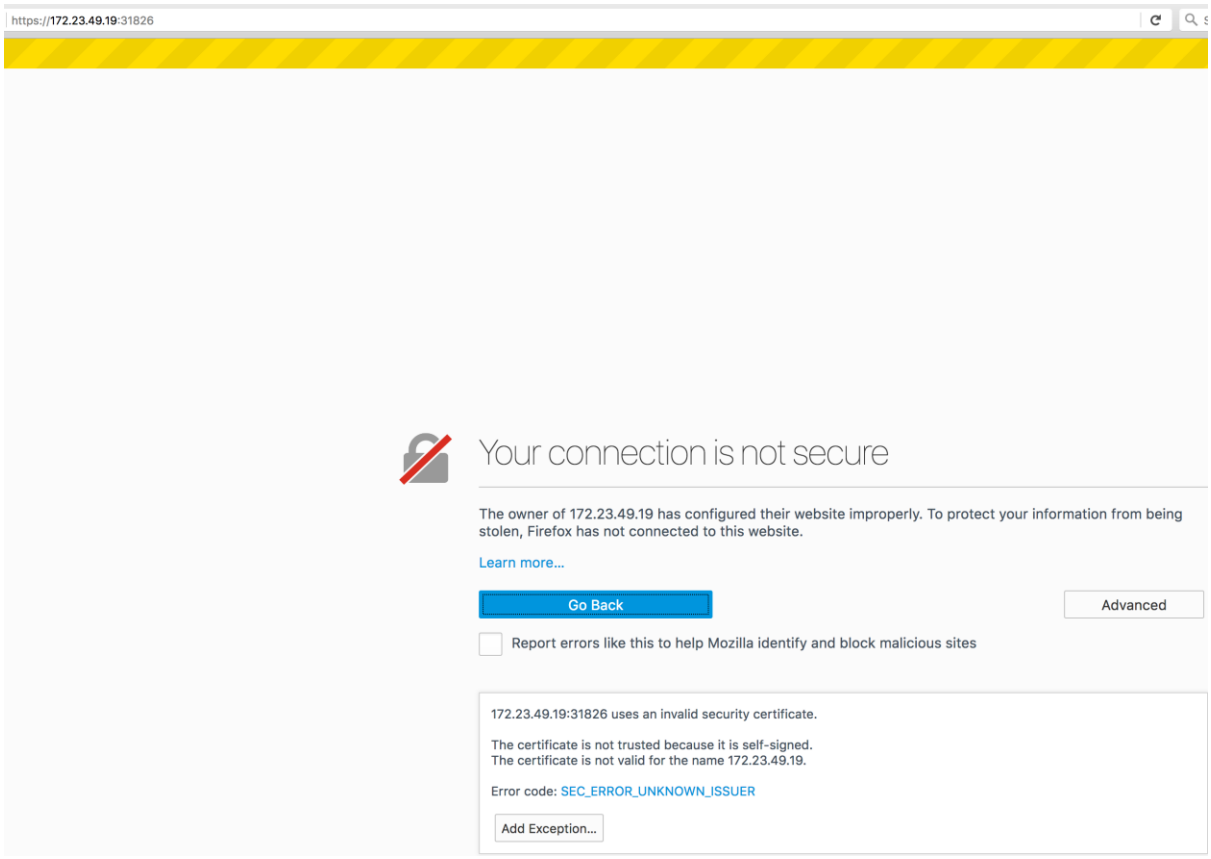
Service details

TYPE	DETAIL
Name	iib10mq9-iib-mq
Namespace	default
Creation time	Dec 14th 2017 at 3:26 PM
Type	NodePort
Labels	app=iib10mq9-iib-mq,chart=iib-mq-1.1.0,heritage=Tiller,release=iib10mq9
Selector	app=iib10mq9-iib-mq
IP	10.0.0.147
Port	iib-mq-server 1414/TCP; iib-mq-web 9443/TCP; iib-mq-console 4414/TCP; iib-mq-nodelistener 7080/TCP; iib-mq-serverlistener 7800/TCP
Node port	iib-mq-server 30231/TCP iib-mq-web 31826/TCP iib-mq-console 30212/TCP iib-mq-nodelistener 30505/TCP iib-mq-serverlistener 30395/TCP
Session affinity	None

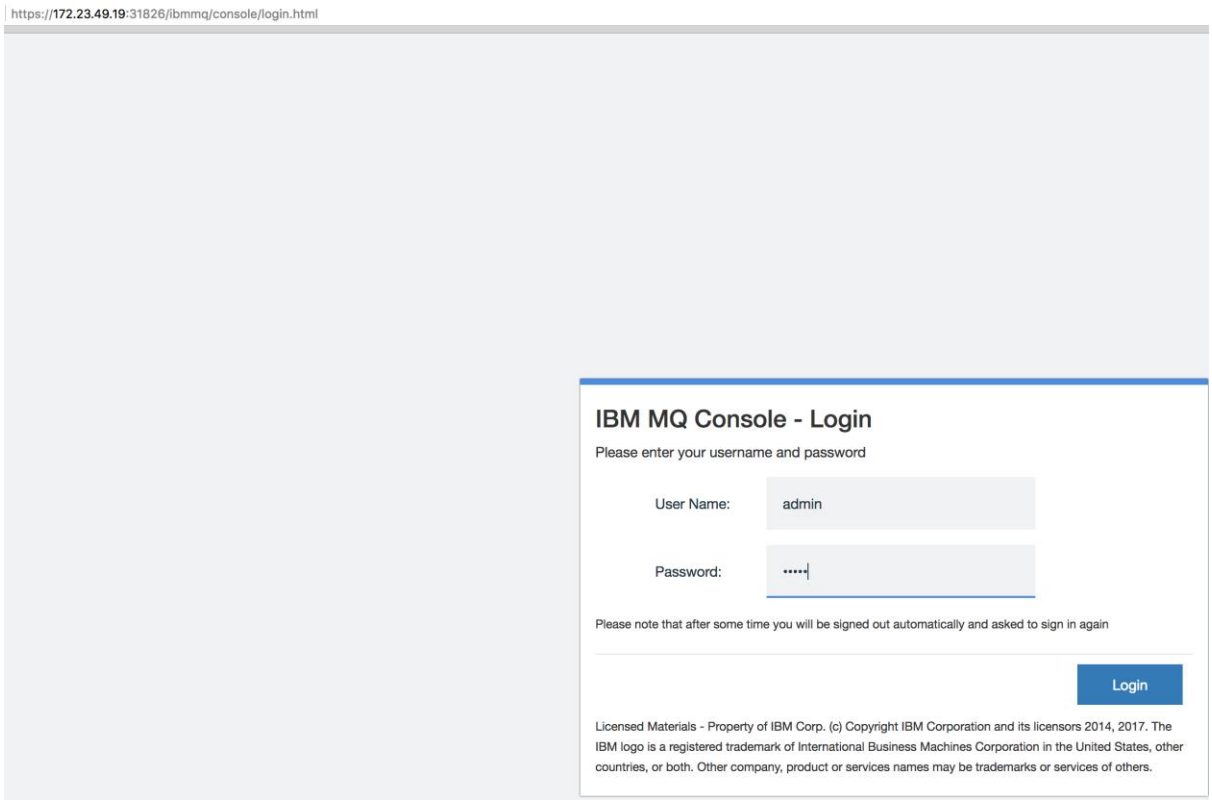
Select the iib-mq-web 31826/TCP link.

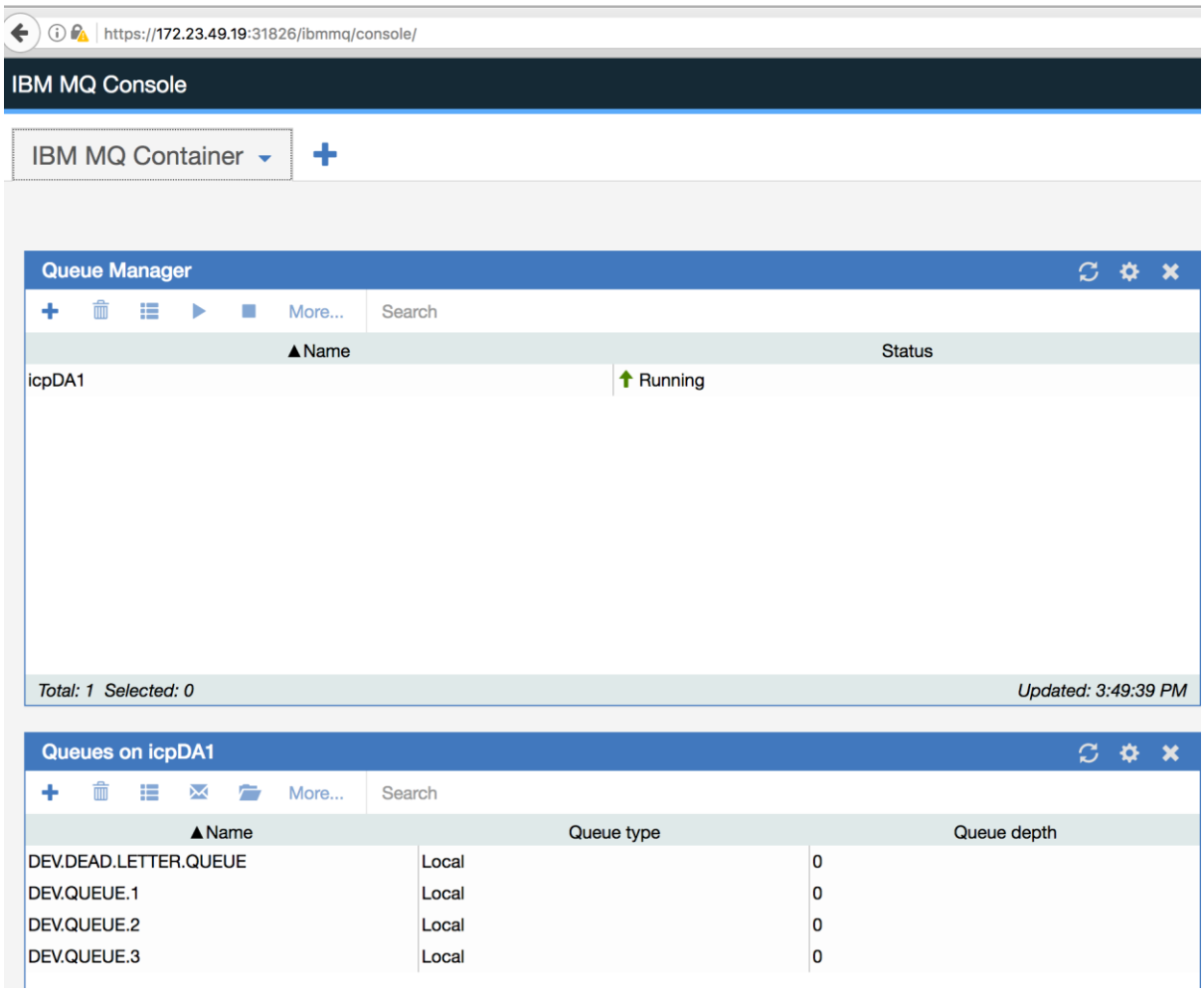
In the browser add the https:// up front

Add certificate

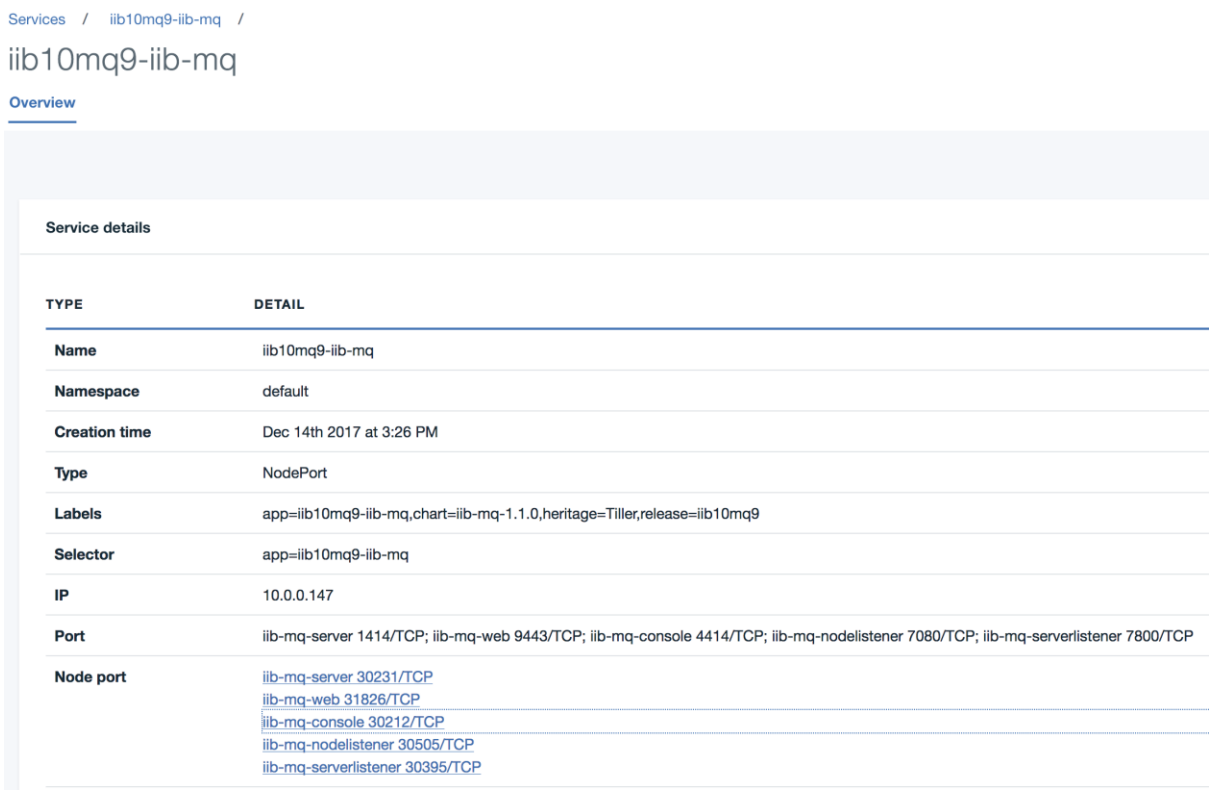


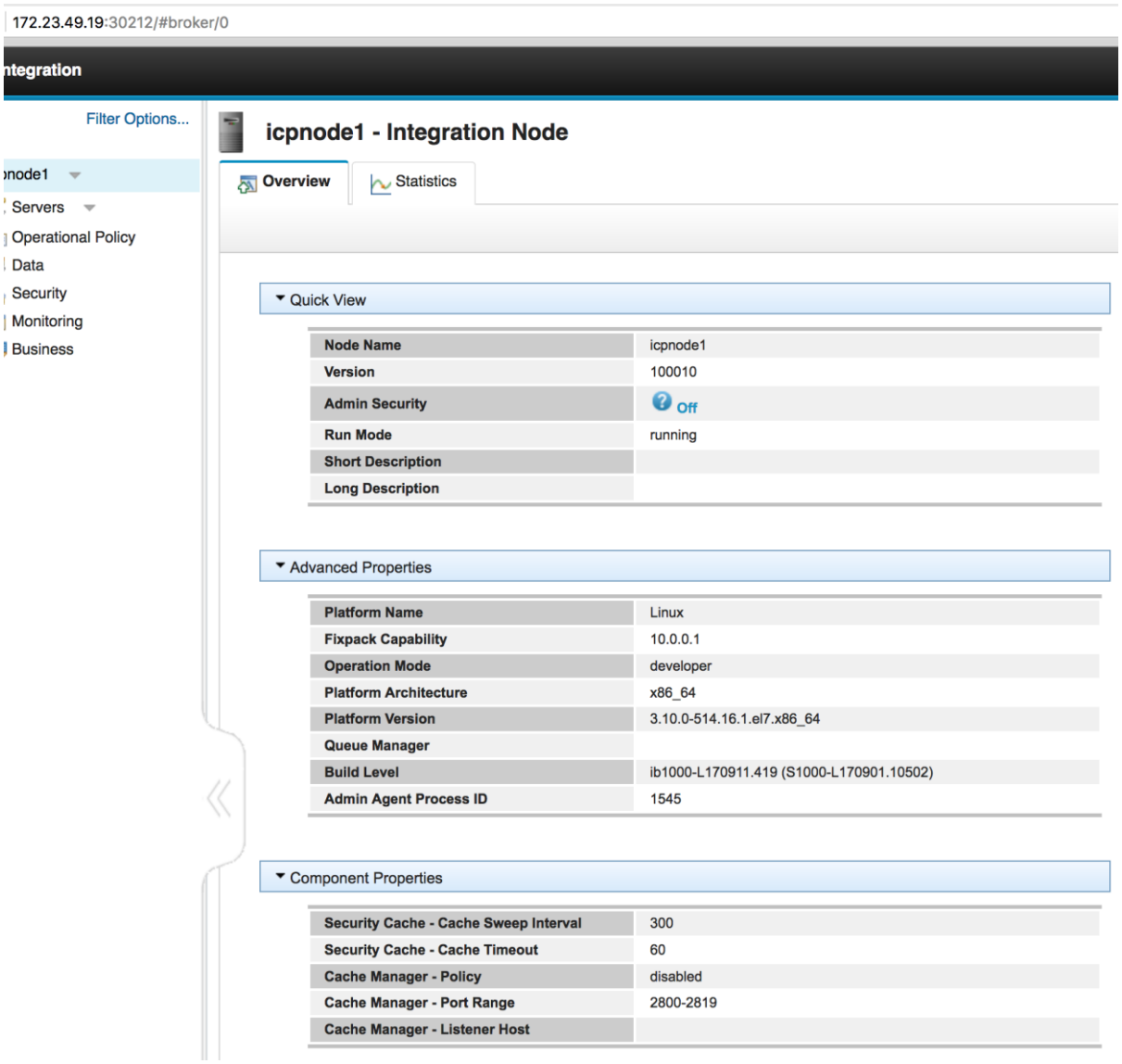
Log into MQ console with admin/passw0rd





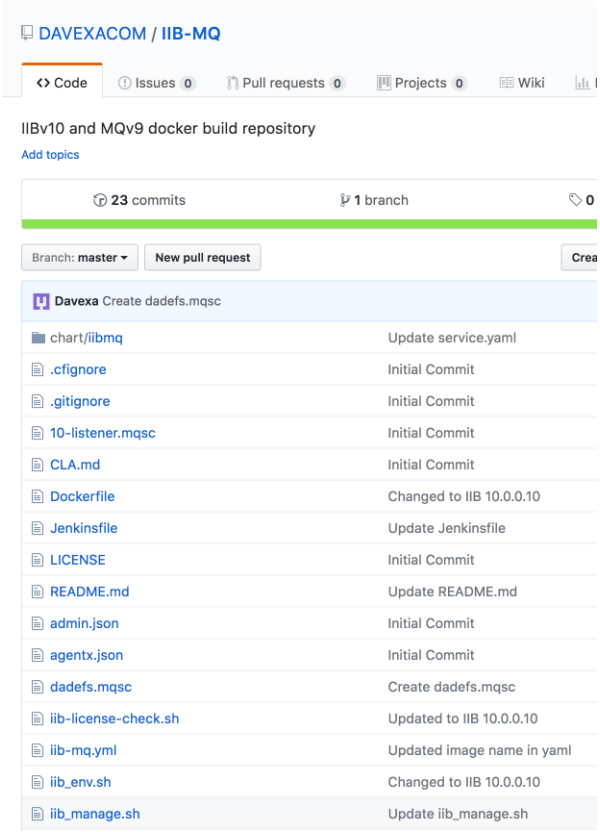
Connect to IIB Web UI




172.23.49.19:30212/#broker/0
ntegration
Filter Options...
node1
Servers
Operational Policy
Data
Security
Monitoring
Business
icpnode1 - Integration Node
Overview
Statistics
Quick View
Node Name
icpnode1
Version
100010
Admin Security
Off
Run Mode
running
Short Description
Long Description
Advanced Properties
Platform Name
Linux
Fixpack Capability
10.0.0.1
Operation Mode
developer
Platform Architecture
x86_64
Platform Version
3.10.0-514.16.1.el7.x86_64
Queue Manager
Build Level
ib1000-L170911.419 (S1000-L170901.10502)
Admin Agent Process ID
1545
Component Properties
Security Cache - Cache Sweep Interval
300
Security Cache - Cache Timeout
60
Cache Manager - Policy
disabled
Cache Manager - Port Range
2800-2819
Cache Manager - Listener Host

Adding your own MQSC for MQ files and rebuilding with Jenkins

How it hangs together



The docker file copies all ***.mqsc*** in the github repository to a /mqm/etc in the runtime container. In our example that is 10-listener.mqsc and dadefs.mqsc

```
COPY mq-dev-config.sh mq-license-check.sh mq.sh setup-mqm-web.sh setup-var-mqm.sh /usr/local/bin/
COPY *.mqsc /etc/mqm/
COPY admin.json /etc/mqm/

COPY mq-dev-config /etc/mqm/mq-dev-config
```

The docker file copies a bunch of ***.sh*** including **iib-manage.sh** to the /usr/local/bin

```
# Copy in script files
COPY iib_manage.sh /usr/local/bin/
COPY iib-license-check.sh /usr/local/bin/
COPY iib_env.sh /usr/local/bin/
```

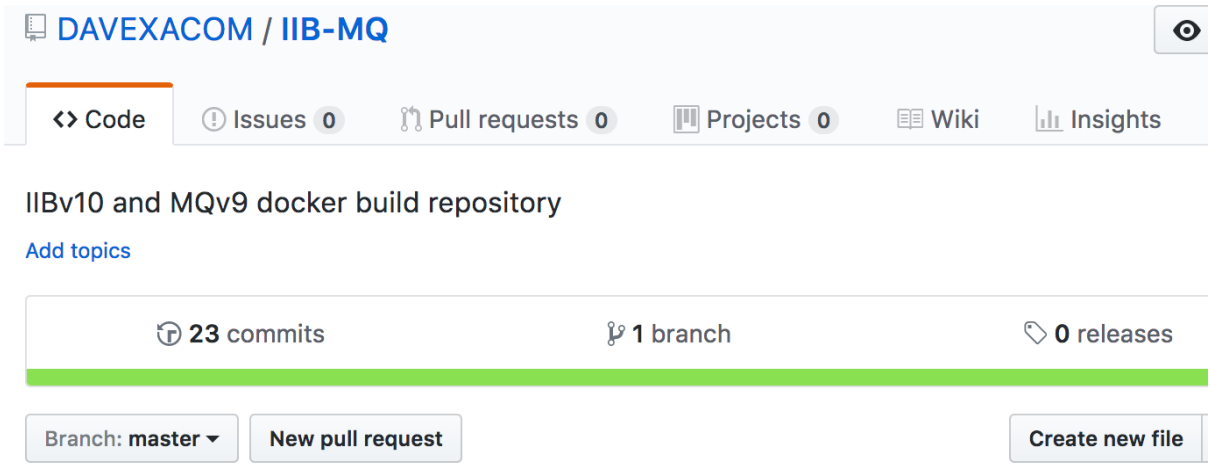
The docker file sets an ***entry point*** for the runtime container that is **iib-manage.sh**

```
127   # Set entrypoint to run management script
128   ENTRYPOINT ["iib_manage.sh"]
```

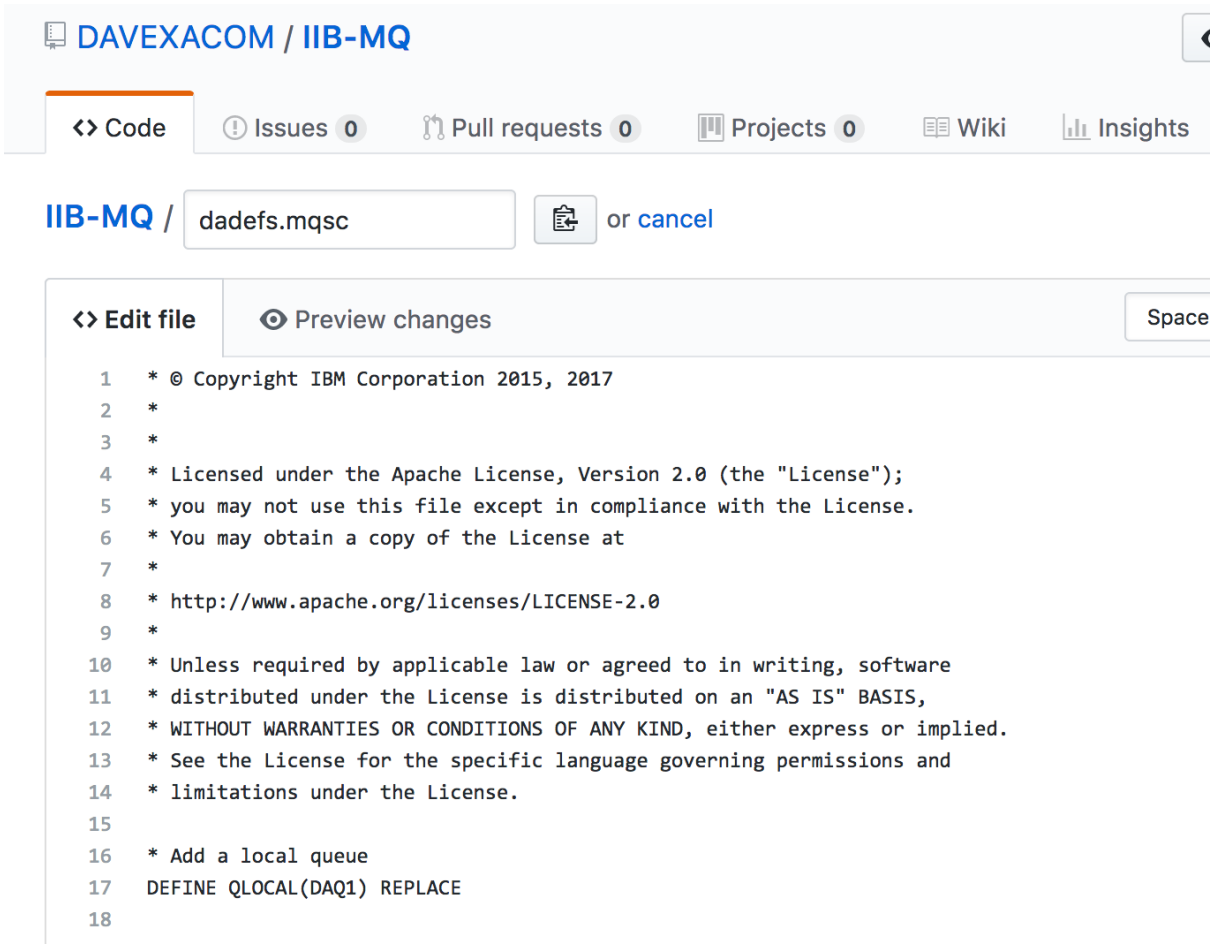
Iib_manage.sh is executed when the container starts up

Updating IIB-MQ GitHub Repo in DAVEXACOM Org

Create a new file called dadefs.mqsc or similar

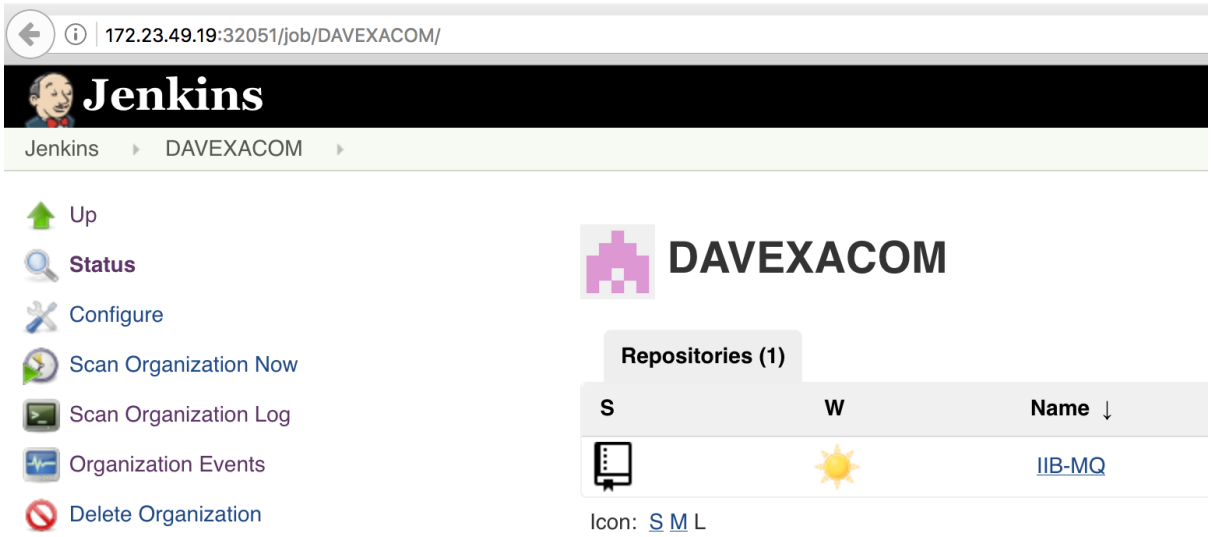


Add your definitions and commit the changes.



Re-run the ICP Microservices builder Jenkins pipeline

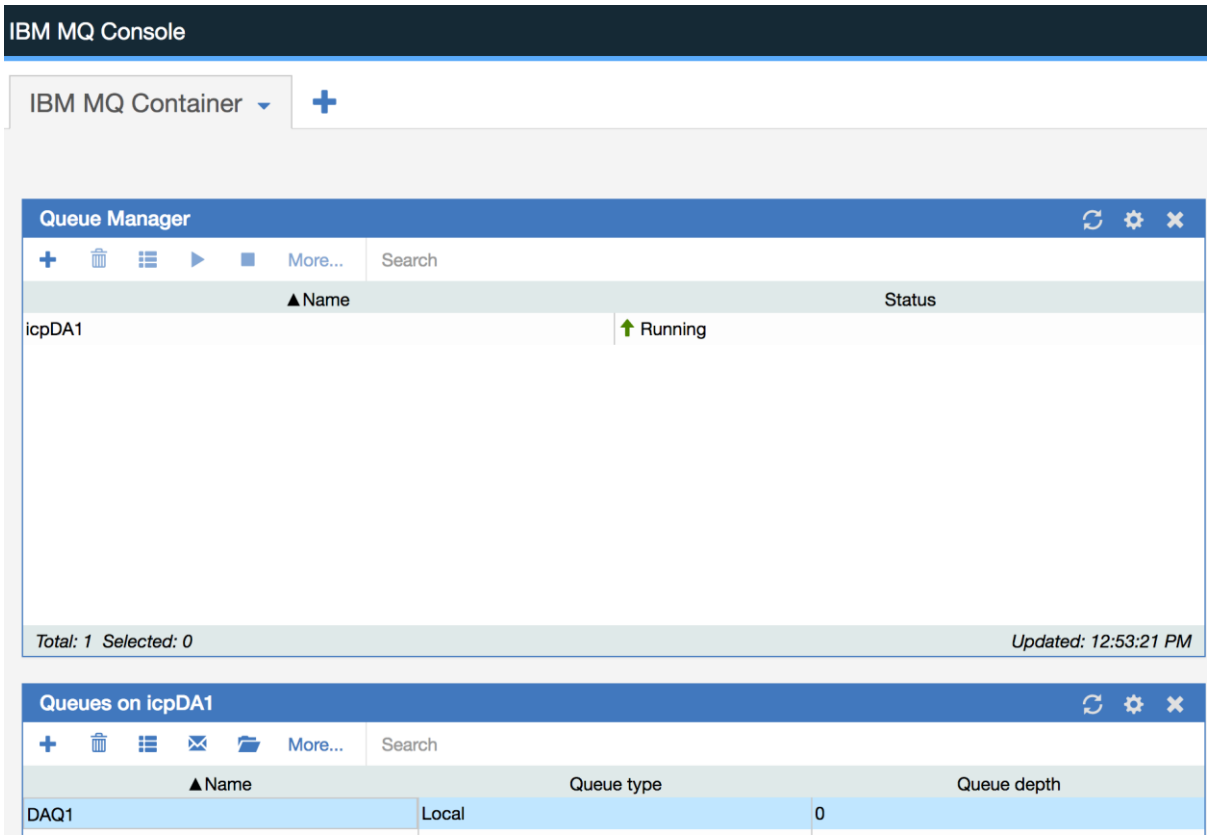
Select Scan Organization Now



The build will start and you can select the ***#N*** build number and then you can follow the build on the console

Check the results in the IBM MQ Console

You'll need to reconnect once the build has finished and the previous container replaced.



Adding your own BAR files for IIB and rebuilding with Jenkins

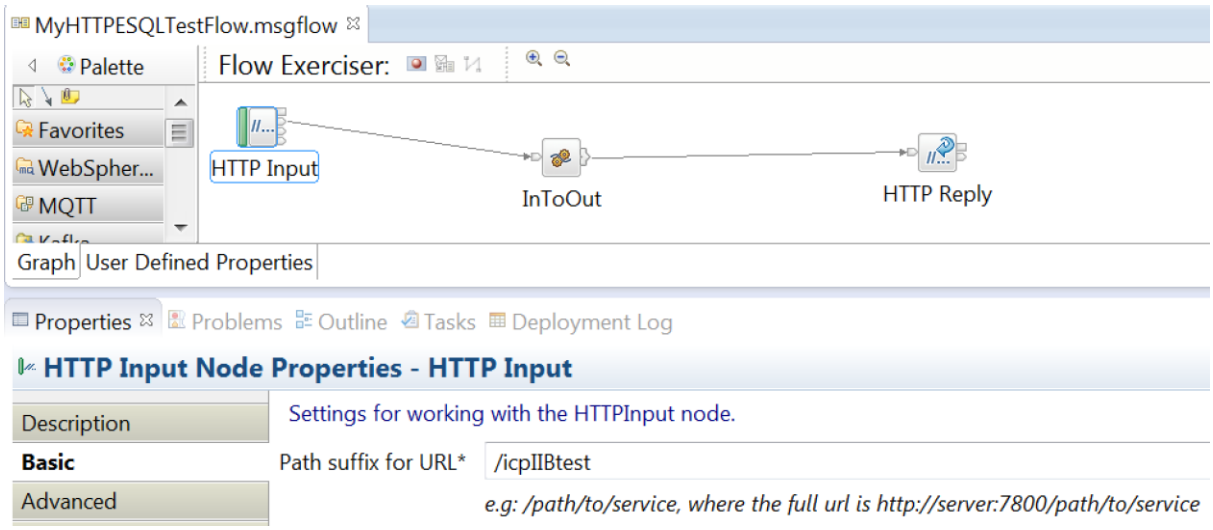
How it hangs together

Basically, we'll use the same approach with the following changes

1. Instead of creating a file we'll import a BAR file to the repository from a local IIB toolkit workspace.
2. Change the dockerfile to copy all ***.bar*** file to mqm/etc along with the mqsc files
3. Change iib_manage.sh to execute the mqsideploy command

Create a BAR file in an IIB Toolkit

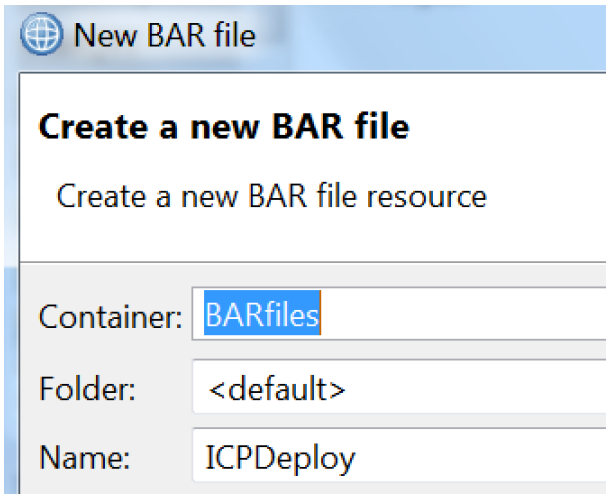
In this example Message flow MyHTTPESQLTestFlow is in MyICPTestApp IIB application



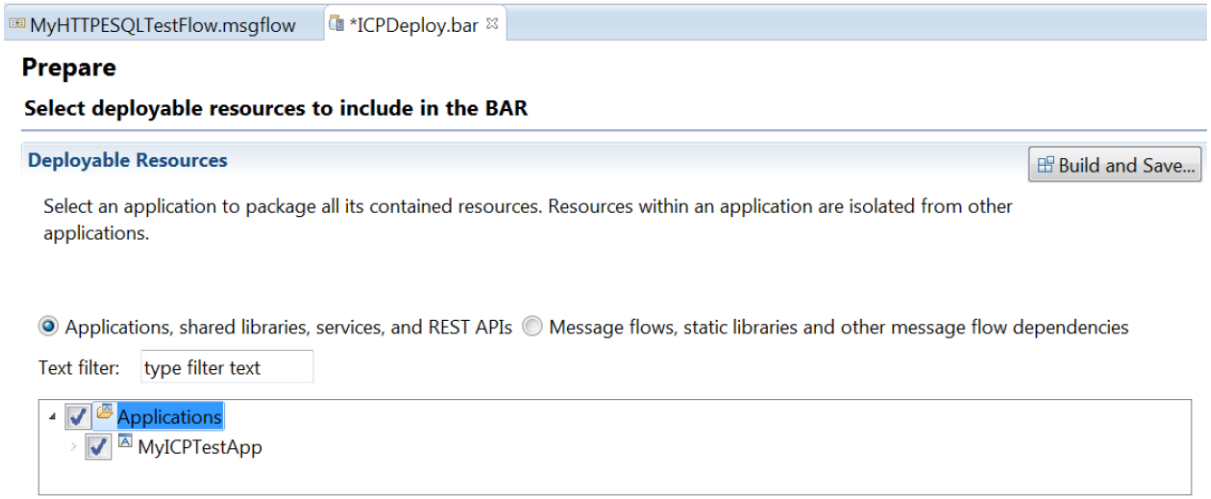
Compute Node InToOut references the following ESQL

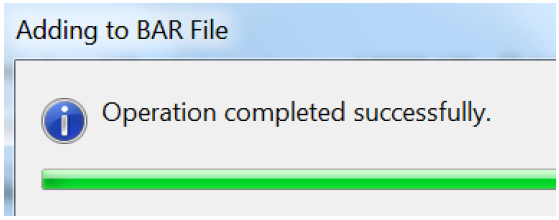
```
CREATE COMPUTE MODULE MyHTTPESQLTestFlow_InToOut
	CREATE FUNCTION Main() RETURNS BOOLEAN
	BEGIN
		CALL CopyMessageHeaders();

		DECLARE mystring CHARACTER;

		SET mystring='hello world';
		SET OutputRoot.BLOB.BLOB=CAST (mystring AS BLOB CCSID 1208);

		RETURN TRUE;
	END;
```

File->New-Broker Archive (BAR) file

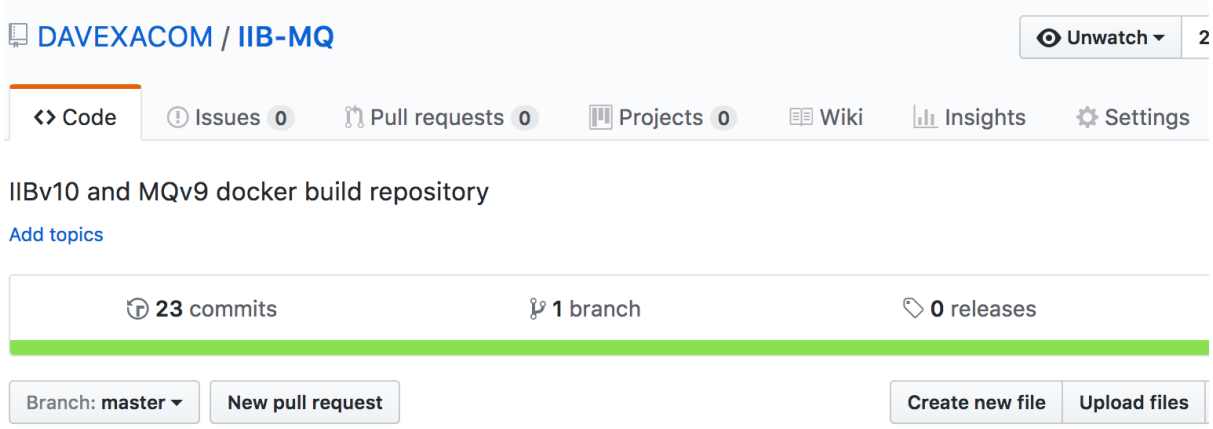


Select MyICPTestApp and hit Build and Save

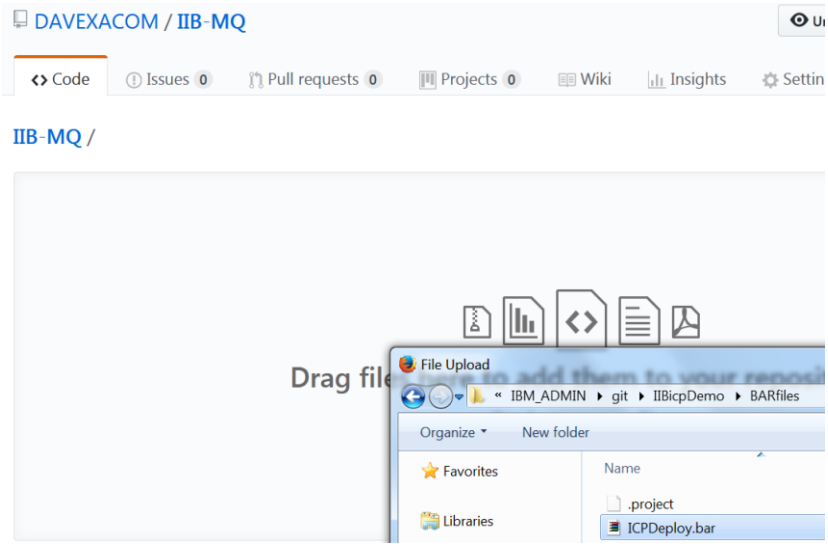




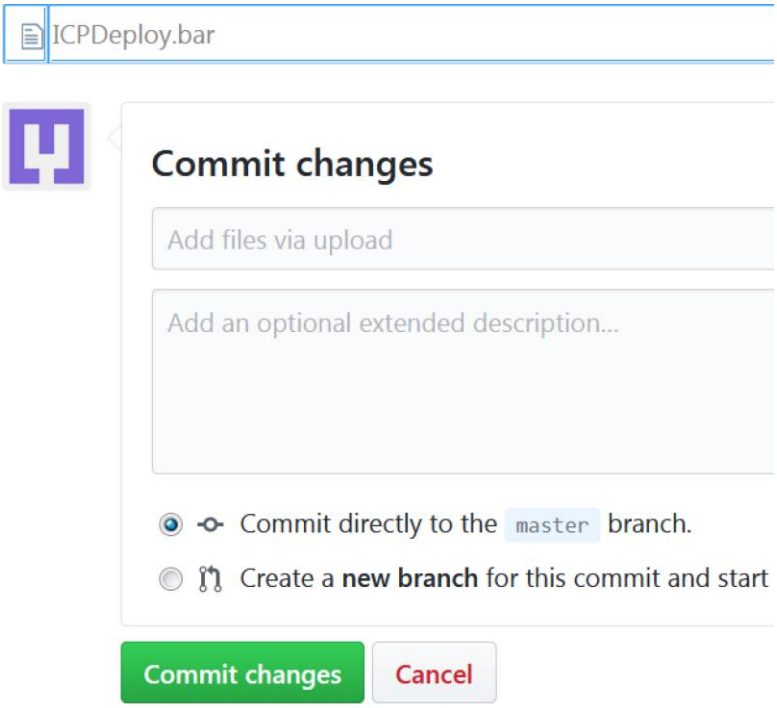
Upload the BAR file to the IIB-MQ repo on GitHub

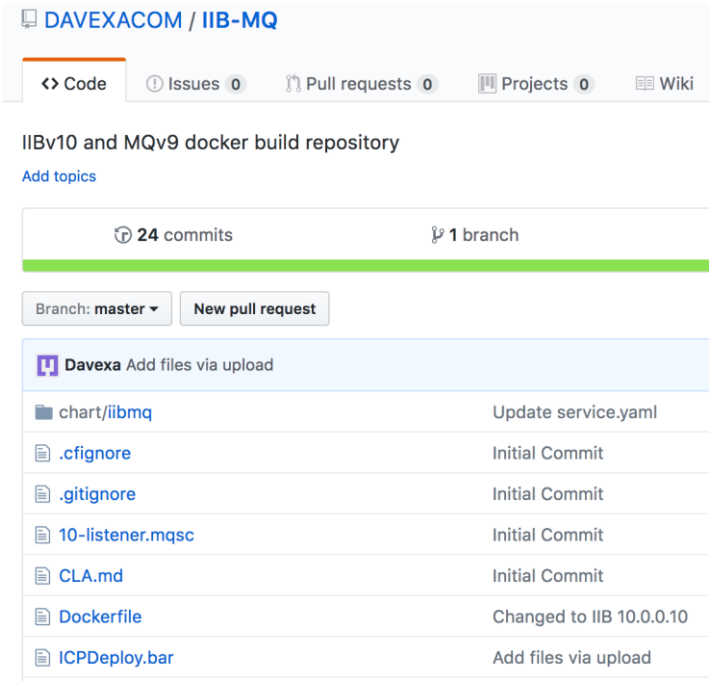


Navigate to your BAR file location in the IIB workspace on the local file system



Upload and commit changes





Edit the dockerfile on GitHub

```
67
68          COPY mq-dev-config.sh mq-license-check.sh mq.sh setup-mqm-web.sh setup-var-mqm.sh /usr/local/bin/
69          COPY *.mqsc /etc/mqm/
70          COPY *.bar  /etc/mqm/
71          COPY admin.json /etc/mqm/
72
73          COPY mq-dev-config /etc/mqm/mq-dev-config
74
75  N chmod +x /usr/local/bin/*.sh
```

Select commit

Edit the iib_manage.sh file on GitHub

Add the following line.

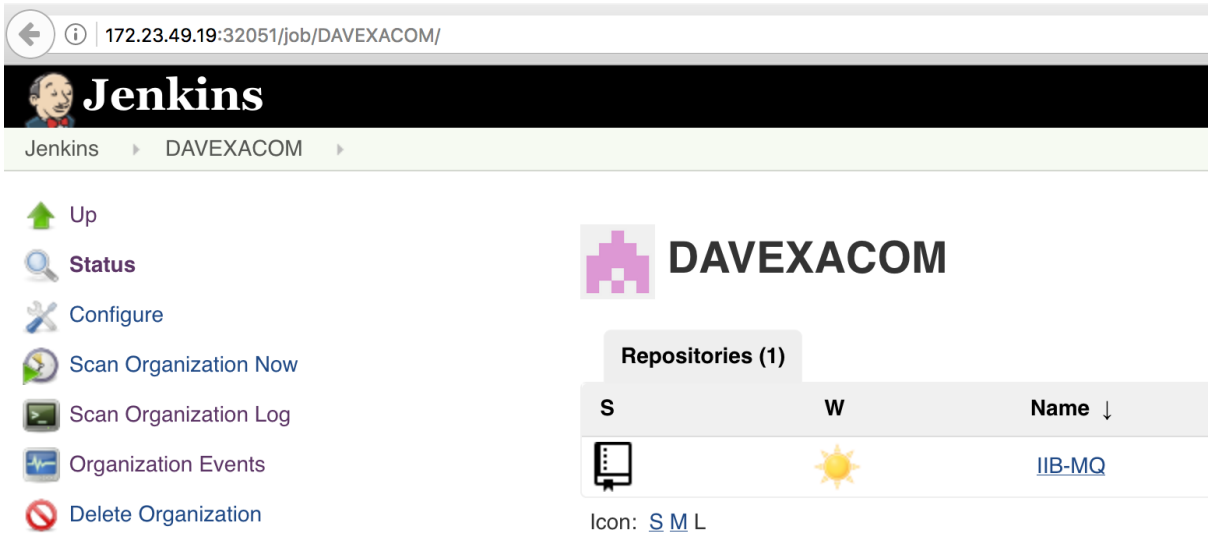
mqsideploy $NODE_NAME -e $EXEC_NAME -a /etc/mqm/ICPDeploy.bar -m

```
140
141             fi
142             mqsichangeproperties $NODE_NAME -e $EXEC_NAME -o ComIbmIIBSwitchManager
143
144             mqsistop $NODE_NAME
145             mqsistart $NODE_NAME
146
147             mqsideploy $NODE_NAME -e $EXEC_NAME -a /etc/mqm/ICPDeploy.bar -m
148
149     }
150
151     monitor()
```

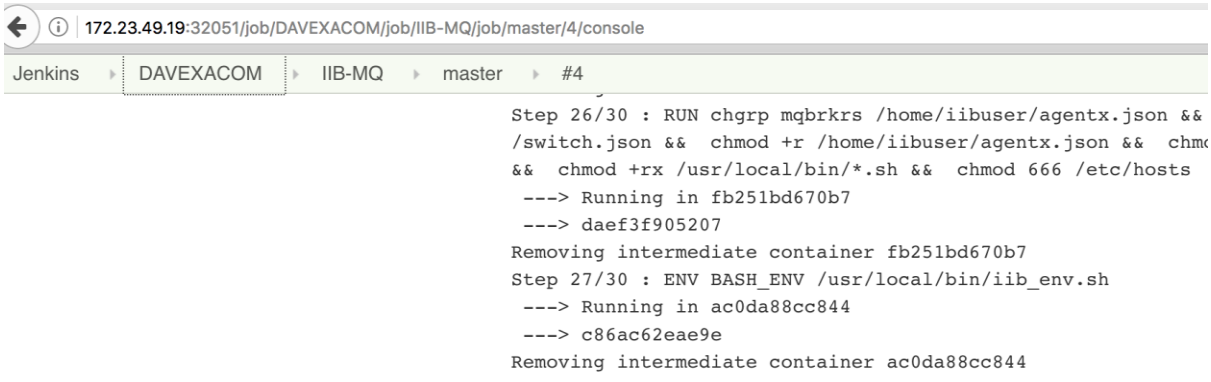
Select commit

Re-run the ICP Microservices builder Jenkins pipeline

Select Scan Organization Now

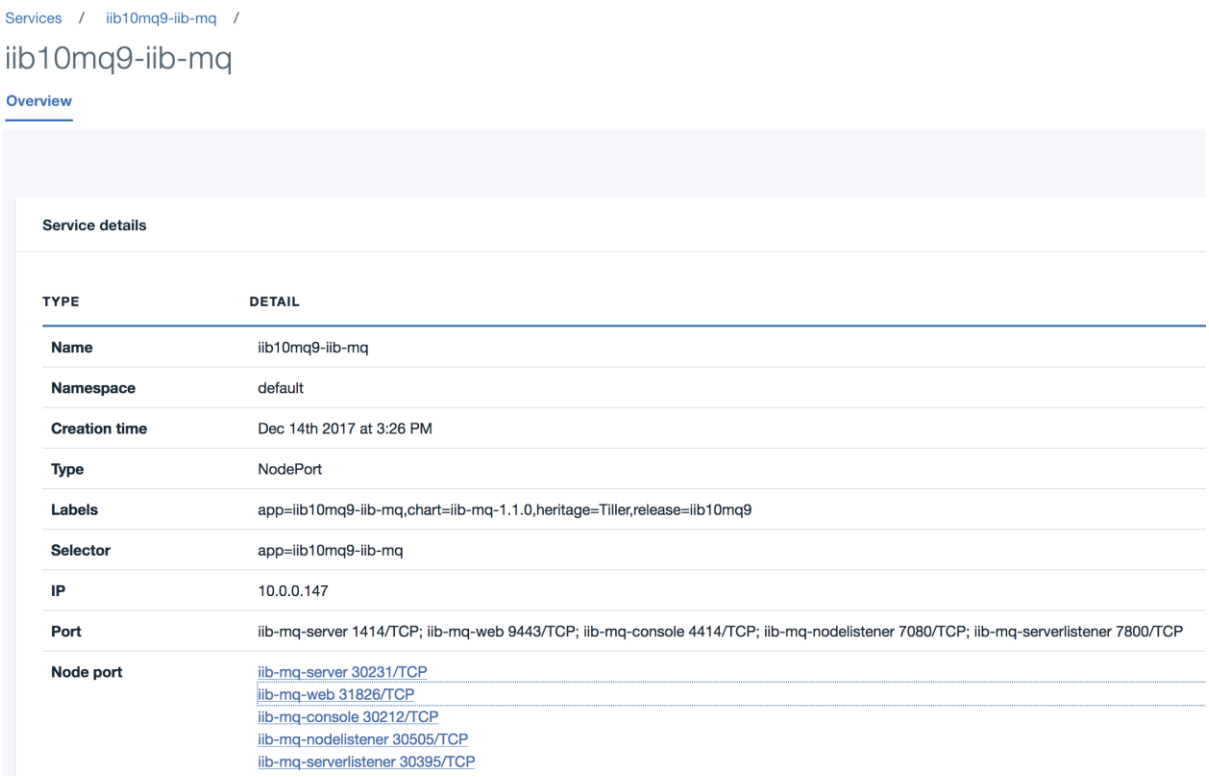


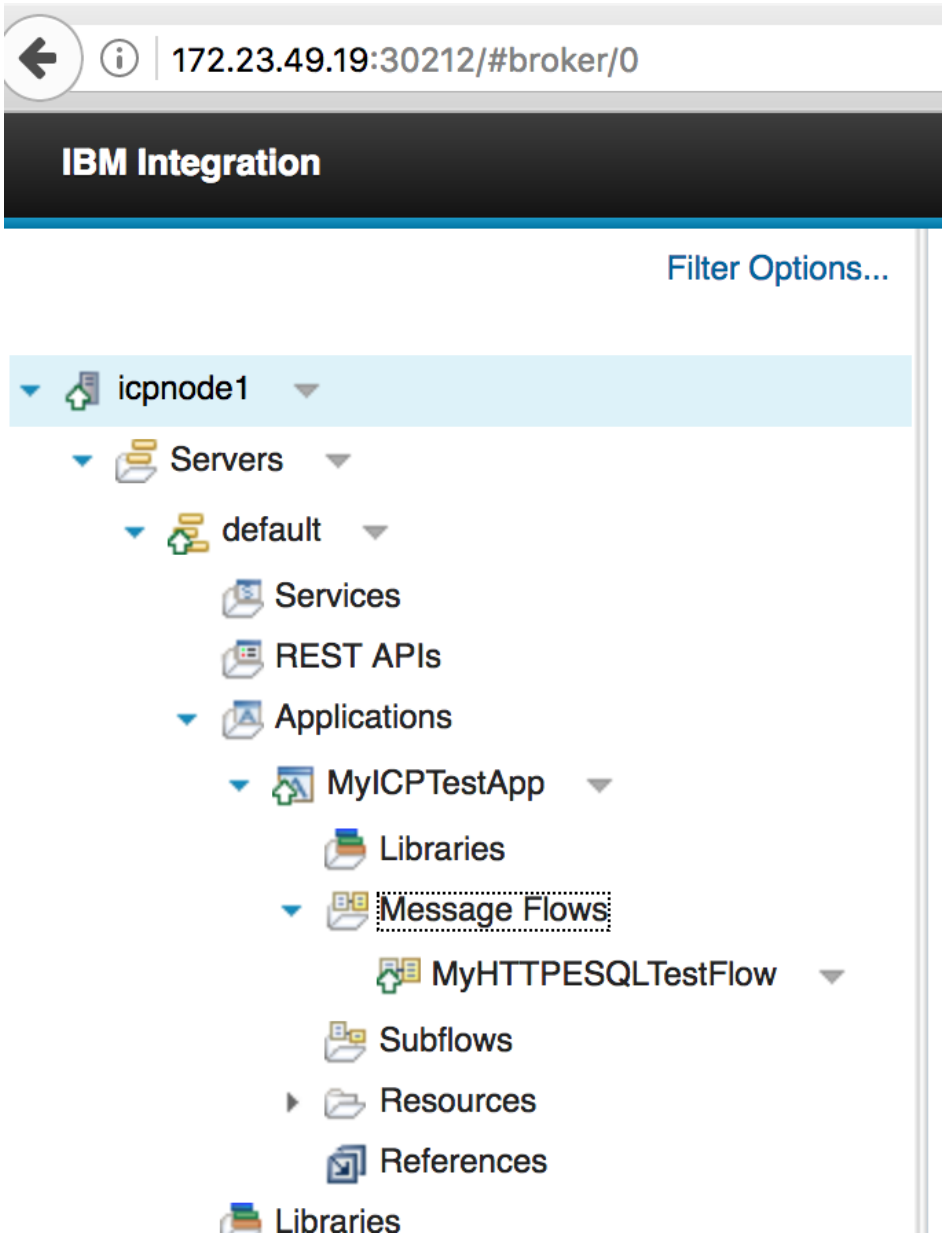
The build will start and you can select the ***#N*** build number and then you can follow the build on the console



Check the results in the IIB Web UI

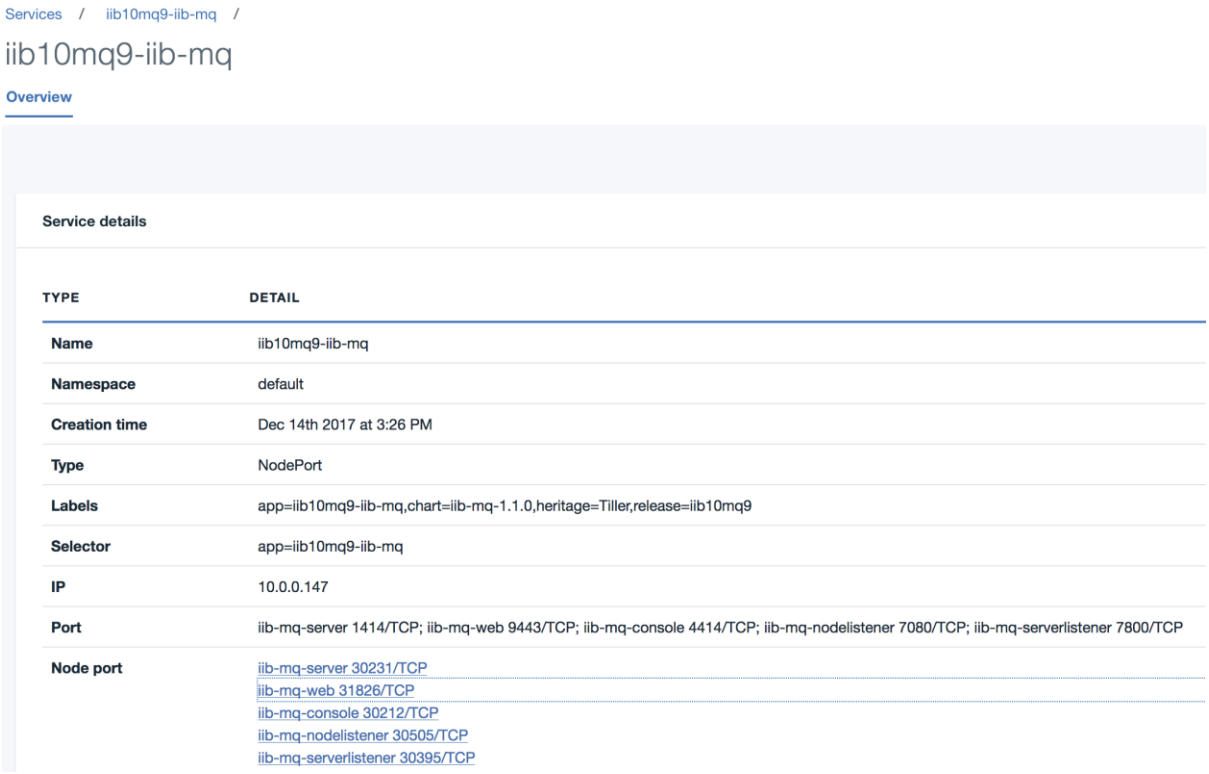
From ICP console -> Workloads -> Helm releases select iib10mq9 then select Services to get to the link to the Web UI (iib-mq-web 31826 in this example)





Test the deployed message flow

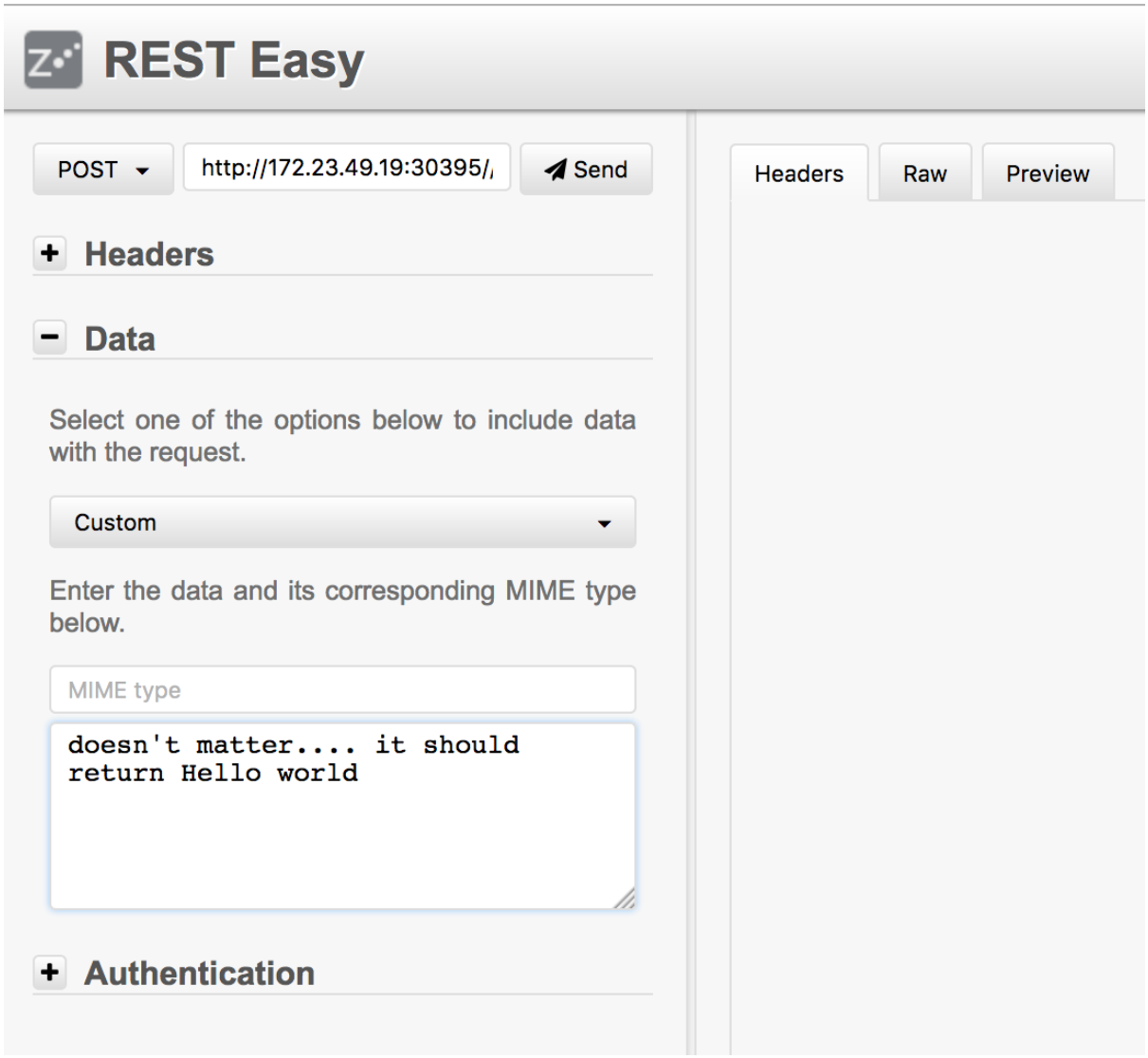
From the ICP Console identify the IIB server listener port.

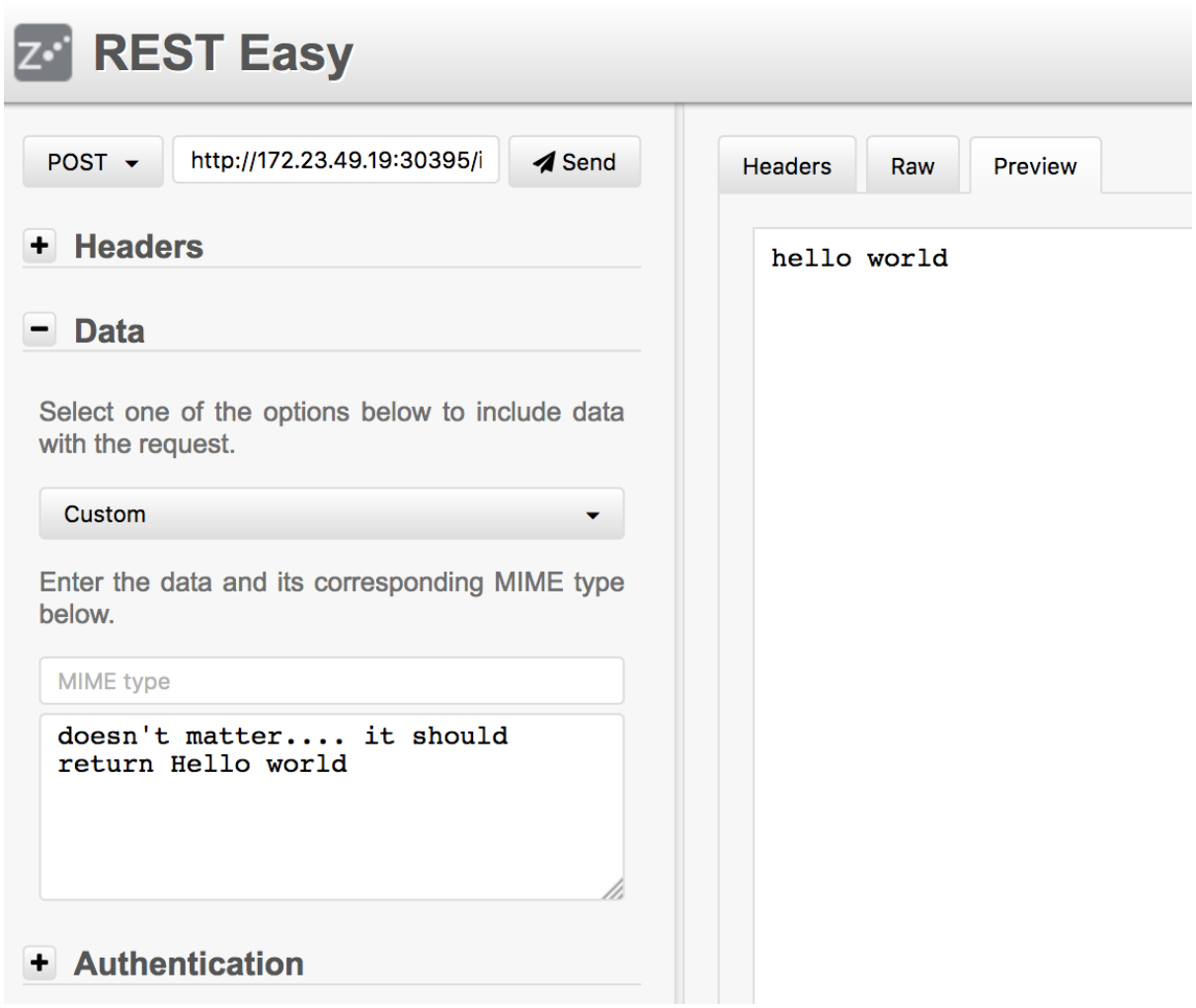


In this example 7800 is represented by 30395

Using an HTTP/REST test client like RestEasy for example

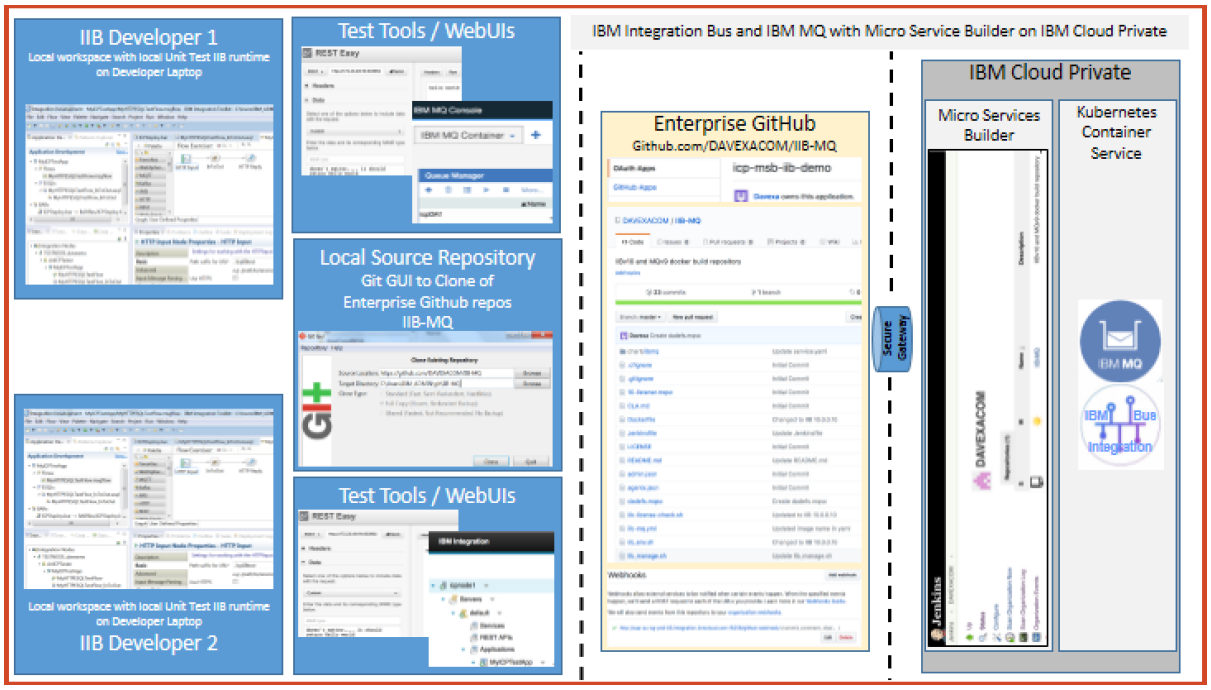
URL: http://172.23.50.111:30395//icpIIBtest


REST Easy
POST
http://172.23.49.19:30395/
Send
Headers
Raw
Preview
Headers
Data
Select one of the options below to include data with the request.
Custom
Enter the data and its corresponding MIME type below.
MIME type
doesn't matter.... it should return Hello world
Authentication


REST Easy
POST
http://172.23.49.19:30395/i
Send
Headers
Data
Select one of the options below to include data with the request.
Custom
Enter the data and its corresponding MIME type below.
MIME type
doesn't matter.... it should return Hello world
Authentication
Headers
Raw
Preview
hello world

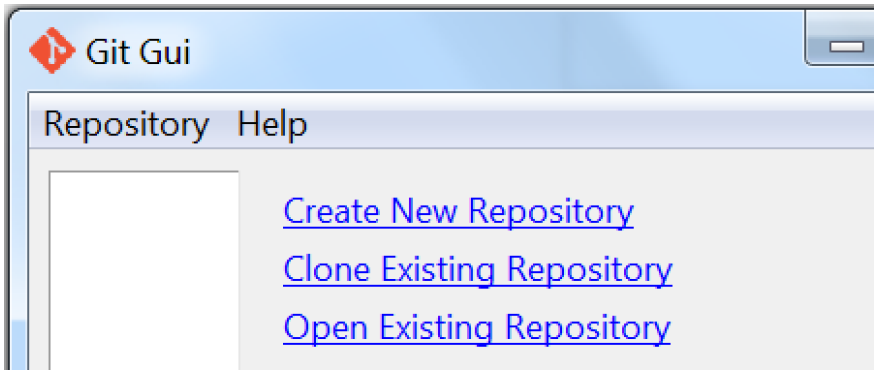
Automating the Build end to end

Solution Overview Diagram

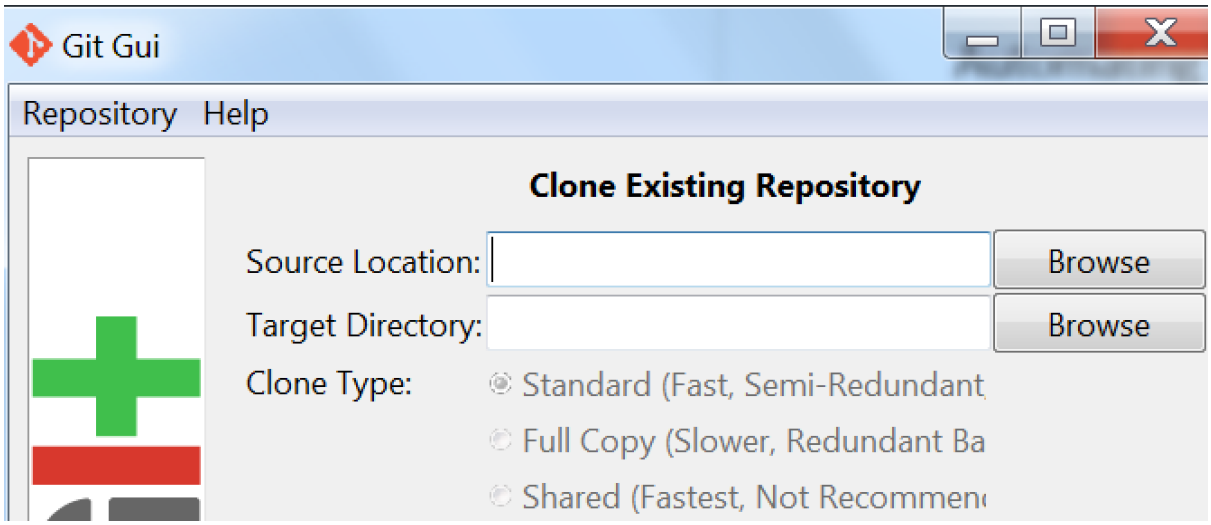


Creating a local clone of the GitHub Repos IIB-MQ

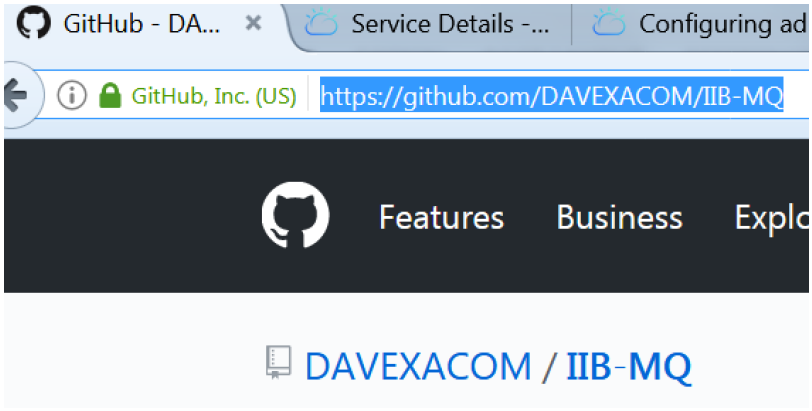
Example using GitGUI



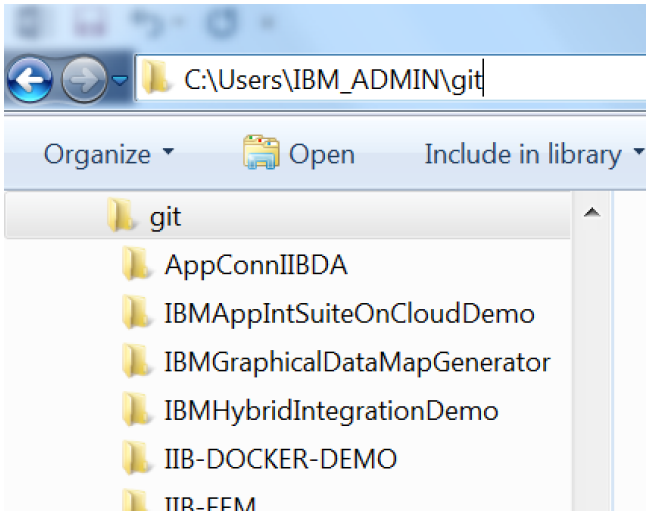
Select Clone Existing Repository



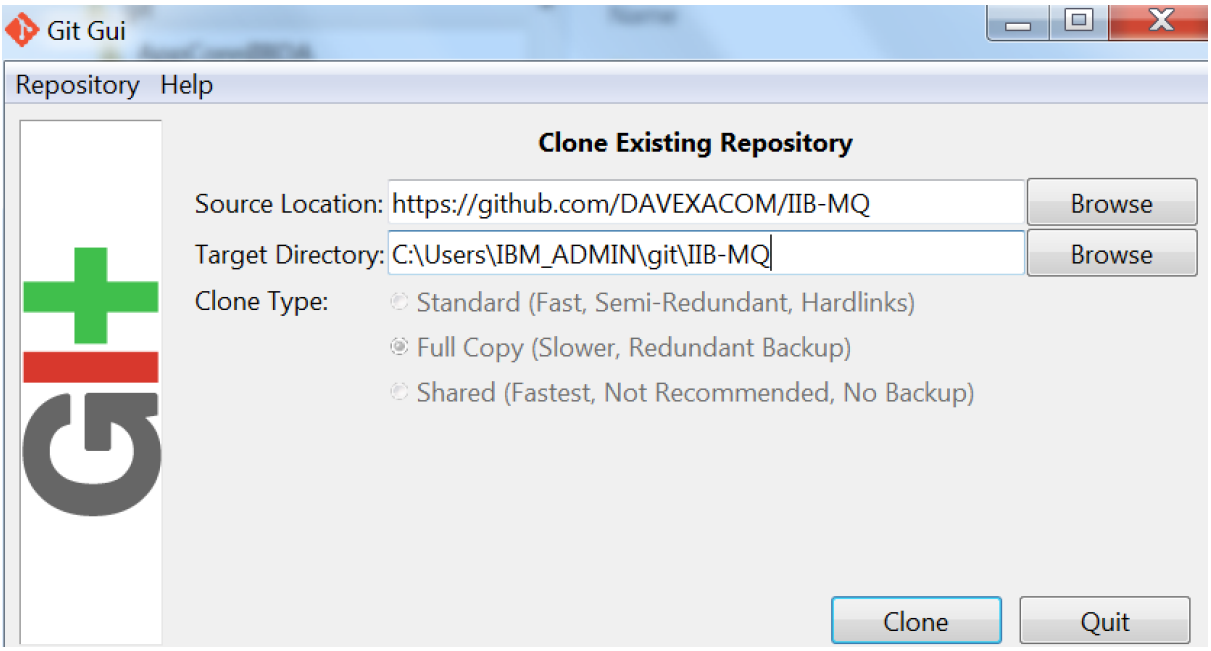
Cut and paste the URL from the GitHub repository to the Git Gui Source Location



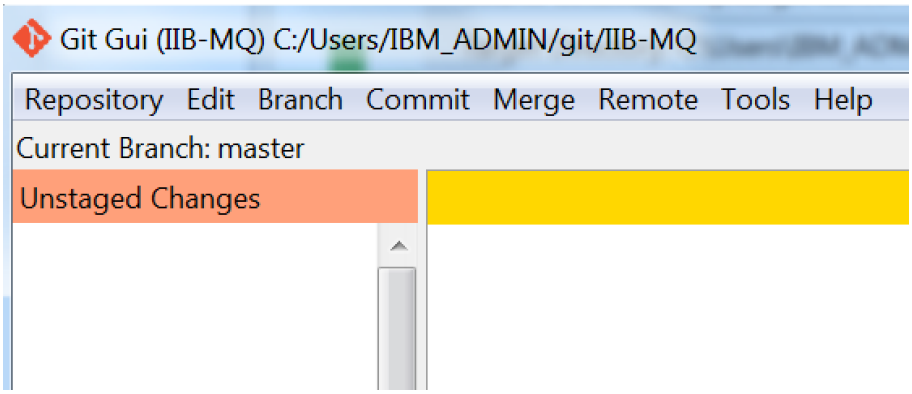
Ut and paste the directory path to your local Git Client to the Target Directory… for example



Then add the “IIB-MQ” and hit clone



You now have a local clone.



Creating the GitHub WebHook to drive Jenkins

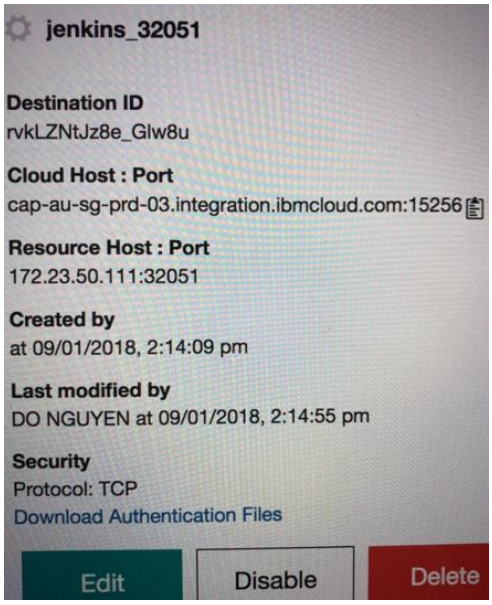
Optional – Use IBM Secure Gateway to connect Github to ICP on Pure

Because the ICP instance I am using in my example sits on a non public IBM network, for inbound traffic such as a web hook publication from GitHub I need a secure gateway. This section will show you how to configure this IBM Public Cloud (Bluemix) service.

The secure gateway will offer a public URL that Github can use to trigger the Jenkins build pipeline on the ICP instance on the private network.

So now you have a Public IP address and port for Github to securely connect to Jenkins on the ICP instance running in a private network (The IBM Innovation Center Sydney in this example).

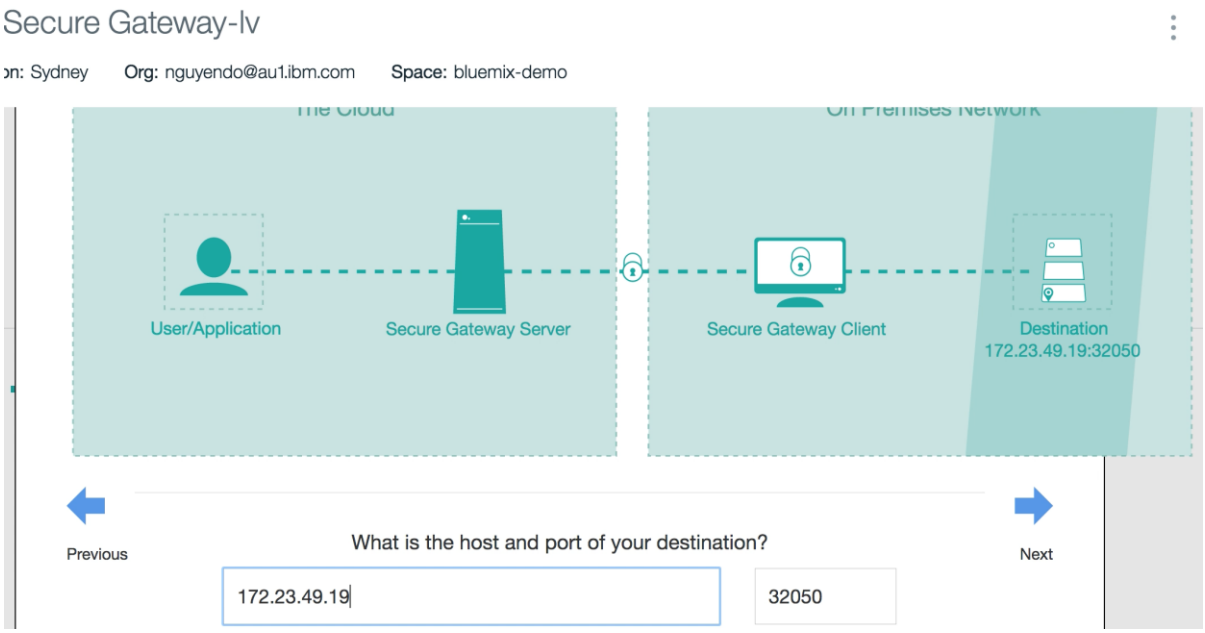
Here is the secure gateway I am using



The public IPAddr:port http://cap-au-sg-prod-03.integration.ibmcloud.com:15256 is offered with secure connection to the actual Jenkins IPAddr:port http://172.23.49.111:32051

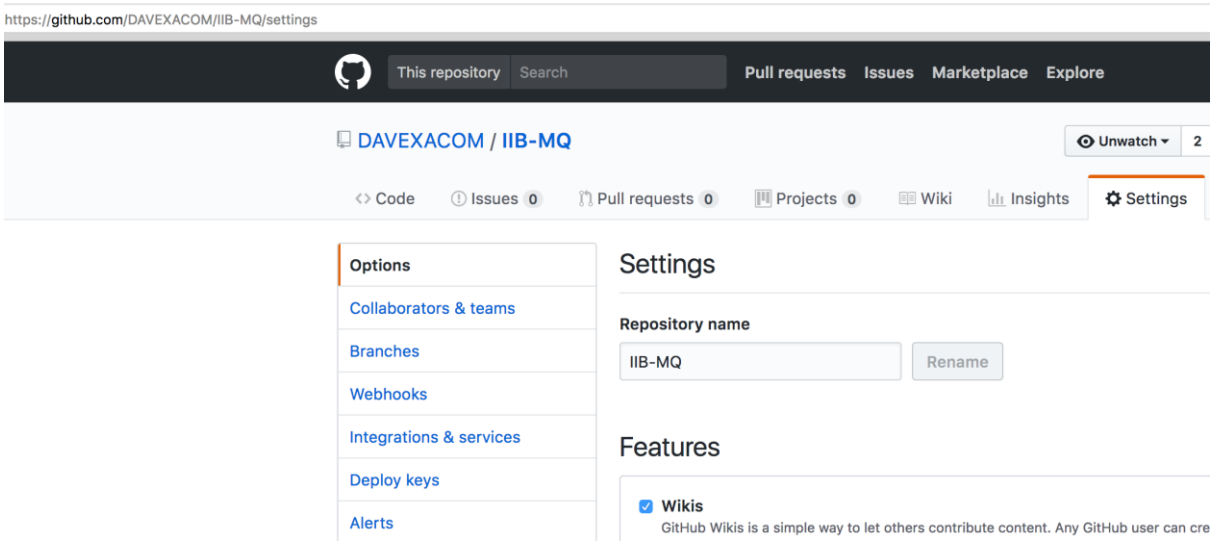
You can use the kubeclt command to pull the IBM Secure Gateway image and push it into ICP. You will then need to ssh into the container and set up the ACLs. In a similar way to that shown below (IP/Ports are not a match for this lab doc)

```
-bash-4.2# docker run -it ibmcom/secure-gateway-client r32C030W8kd_prod_au-syd -t eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
eWQiLCJpYXQiOjE1MTE4NDMyOTksImV4cCI6MTUxOTYxOTI5OX0.Iy0YWunAg-20ajiVDeeYK1lEBzoAXlNpTULonsP5Ja8
IBM Bluemix Secure Gateway Client Version 1.8.0fp3
********************************************************************************************
You  are running the  IBM Secure  Gateway Client for Bluemix. When you enter the provided docker
command the IBM Secure Gateway Client  for Bluemix automatically downloads as a Docker image and
is executed on your system/device. This is released under an IBM license. The  license agreement
for IBM  Secure Gateway Client for Bluemix is available at the following location:

http://www.ibm.com/software/sla/sladb.nsf/lilookup/986C7686F22D4D3585257E13004EA6CB?OpenDocument

Your use of the components of the package and  dependencies constitutes your acceptance  of this
license agreement. If you do  not want to accept the license, immediately quit  the container by
closing the  terminal  window or by  entering 'quit' followed by the ENTER key. Then, delete any
pulled Docker image from your device.

For client documentation, please view the ReadMe located at:
.rpm and .deb installers: /opt/ibm/securegateway/docs/
.dmg installer:           <installation location>/ibm/securegateway/docs/
.exe installer:           <installation location>\Secure Gateway Client\ibm\securegateway\docs\
********************************************************************************************


<press enter for the command line>
[2017-11-28 08:32:22.244] [INFO] (Client ID 1) No password provided. The UI will not require a password for access
[2017-11-28 08:32:22.259] [WARN] (Client ID 1) UI Server started. The UI is not currently password protected
[2017-11-28 08:32:22.259] [INFO] (Client ID 1) Visit localhost:9003/dashboard to view the UI.
cli> [2017-11-28 08:32:22.595] [INFO] (Client ID 10) Setting log level to INFO
[2017-11-28 08:32:23.032] [INFO] (Client ID 10) The Secure Gateway tunnel is connected
[2017-11-28 08:32:23.056] [INFO] (Client ID r32C030W8kd_dIp) Your Client ID is r32C030W8kd_dIp
[2017-11-28 08:32:23.059] [INFO] (Client ID r32C030W8kd_dIp) Synchronizing ACL rules
r32C030W8kd_dIp> acl allow 172.23.49.19:32050
r32C030W8kd_dIp> show acl
r32C030W8kd_dIp>
--------------------------------------------------------------------------------------------
                        -- Secure Gateway Client Access Control List --

 Connections to unlisted rules are currently: denied

 hostname:port                          path                value
 172.23.49.19:32050                                         Allow
 172.23.49.19:30656                                         Allow
--------------------------------------------------------------------------------------------
```

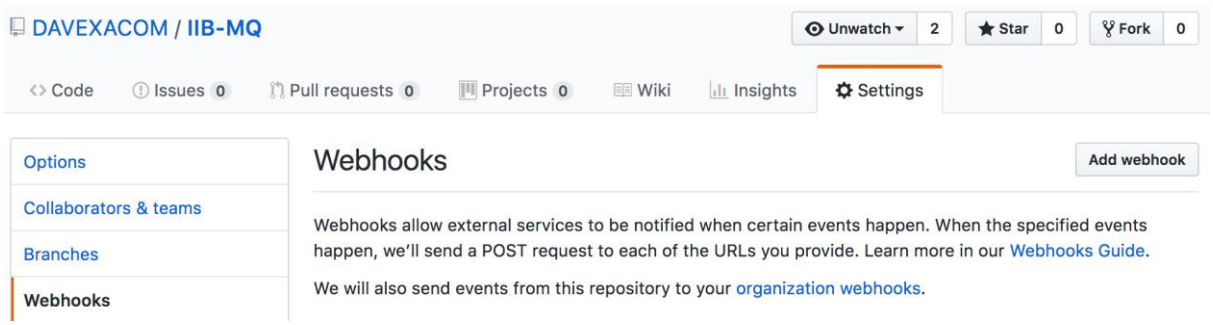


Setting up the Webhook in Github

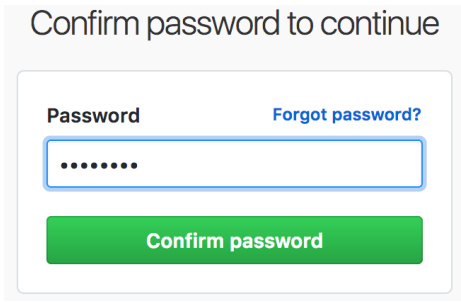
Go to your Github browser, and your organization (DAVEXACOM in this example) and select the IIB-MQ repos. Click on settings.

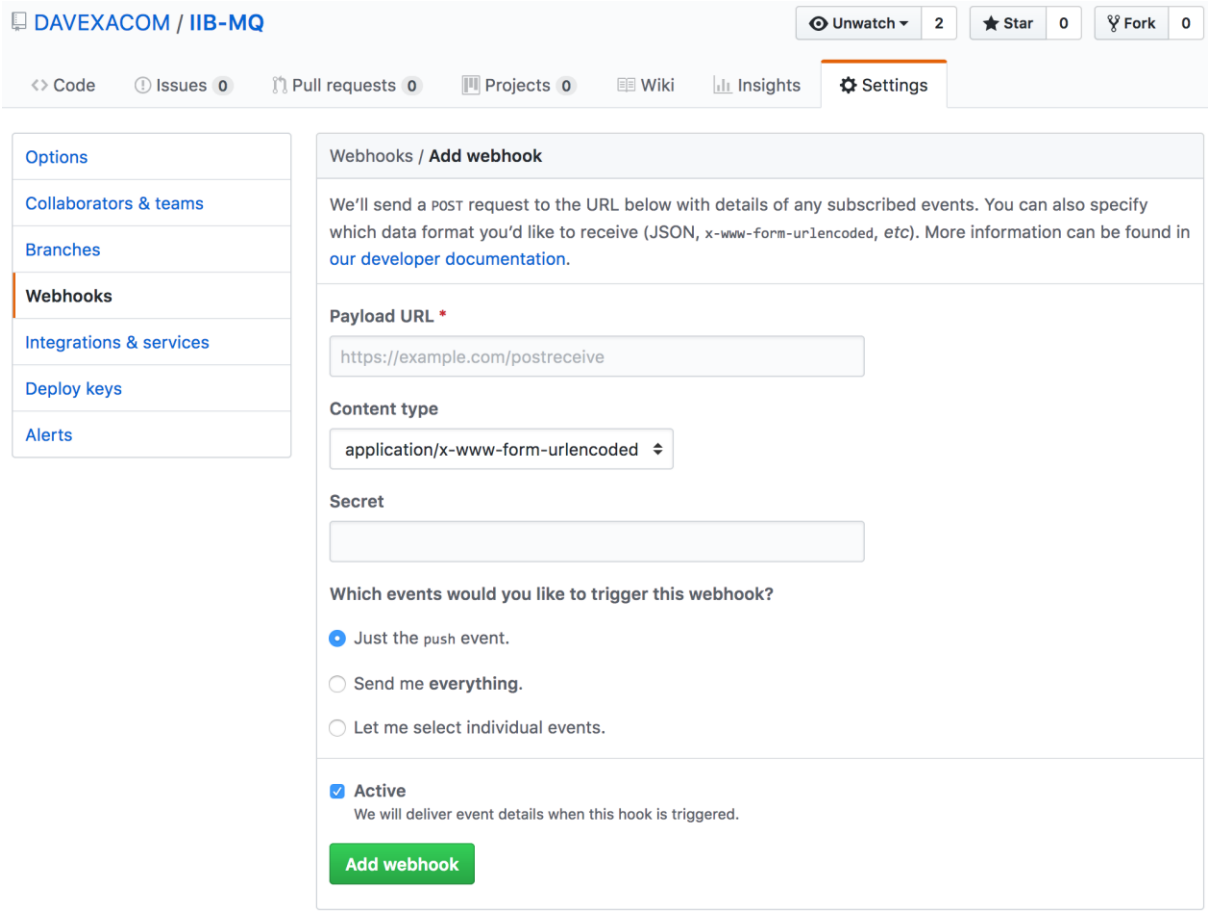


Select Webhooks->Add Webhooks



Confirm password if prompted



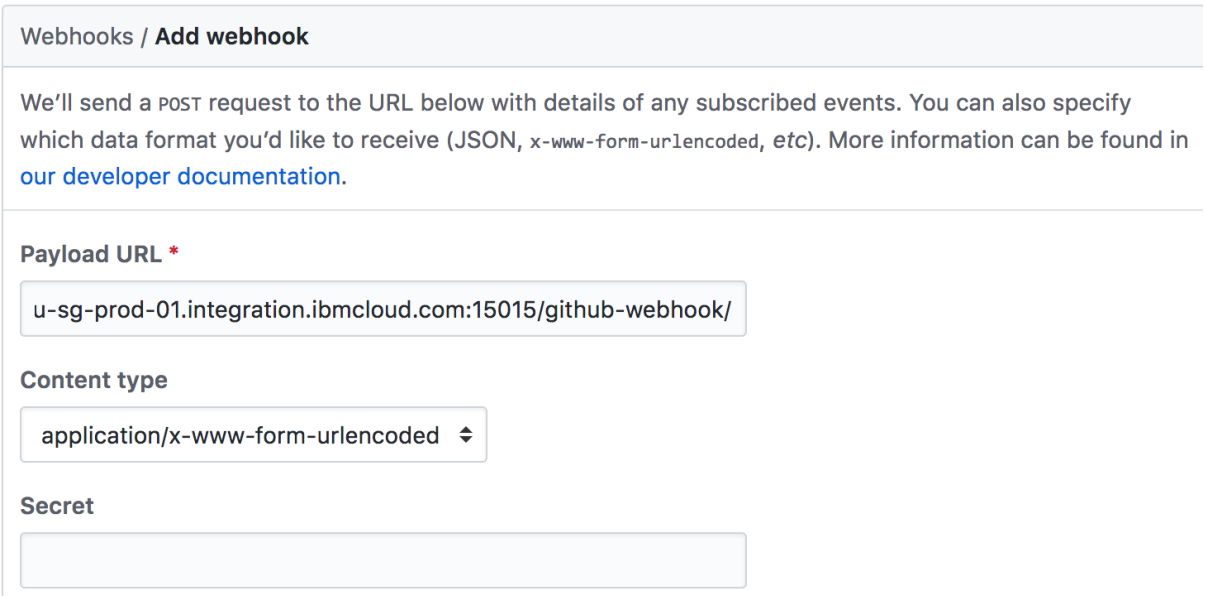


The Payload URL is the address and port number of the Jenkins instance on ICP. However, because in this case ICP sits on a private network we will use the public URL offered by the IBM Cloud (Bluemix) secure gateway service to resolves to the ICP jenkins endpoint.

http://cap-au-sg-prod-03.integration.ibmcloud.com:15256 (or similar)

And add the ***/github-webhook/*** suffix

http://cap-au-sg-prod-03.integration.ibmcloud.com:15256/github-webhook/ or similar



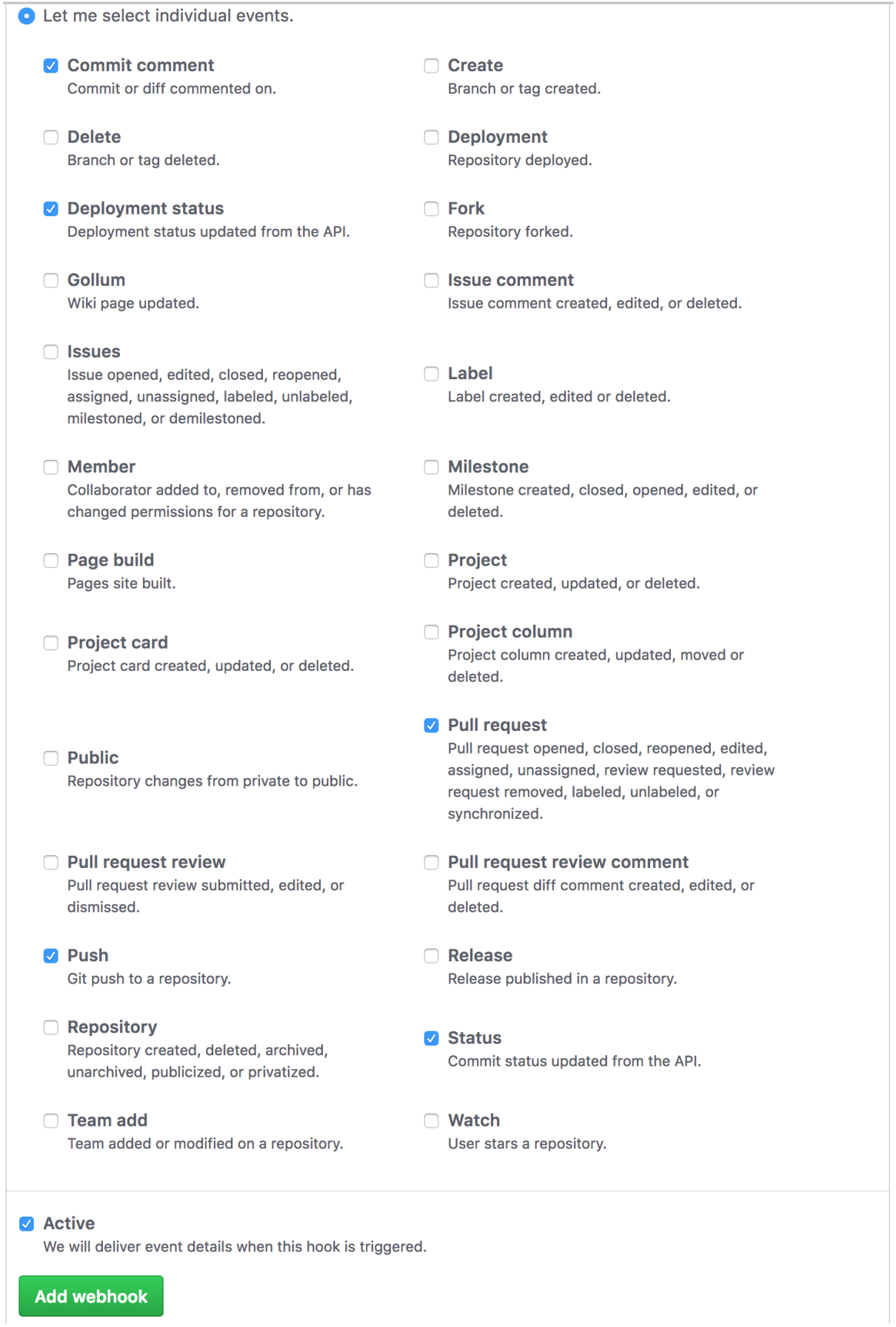
Next select which events to cause the webhook to trigger.

Select the radio button “Let me select individual events”

Then tick

- Push
- Pull Request
- Commit Comment
- Status
- Deployment status

Ensure “Active” is ticked and then hit the Add Webhook button



You should see the WebHook added with a green tick next to it.

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

We will also send events from this repository to your organization webhooks.

✓ http://cap-au-sg-prd-03.integration.ibmcloud.com:15256/github-webhook/ *(commit_comment, depl...)*

Edit Delete

If you get

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.

We will also send events from this repository to your organization webhooks.

⚠ http://cap-au-sg-prod-01.integration.ibmcloud.com:15015/github-webhook/ *(commit_comment, depl...)*

Edit Delete

Hit edit to check your settings and also check connectivity all the way to the ICP Jenkins instance.

Once corrected you'll need to hit the "redeliver" button before you get the green tick (or wait for a retry which could take some time)

Recent Deliveries

✓ 53999a30-f500-11e7-921e-b599cf2b0580 2018-01-09 16:45:40 ...

Request Response 200 Redeliver Completed in 0 seconds.

Headers

```
Request URL: http://cap-au-sg-prd-03.integration.ibmcloud.com:15256/github-webhook/
Request method: POST
content-type: application/x-www-form-urlencoded
Expect:
User-Agent: GitHub-Hookshot/8f9d70f
X-GitHub-Delivery: 53999a30-f500-11e7-921e-b599cf2b0580
X-GitHub-Event: ping
```

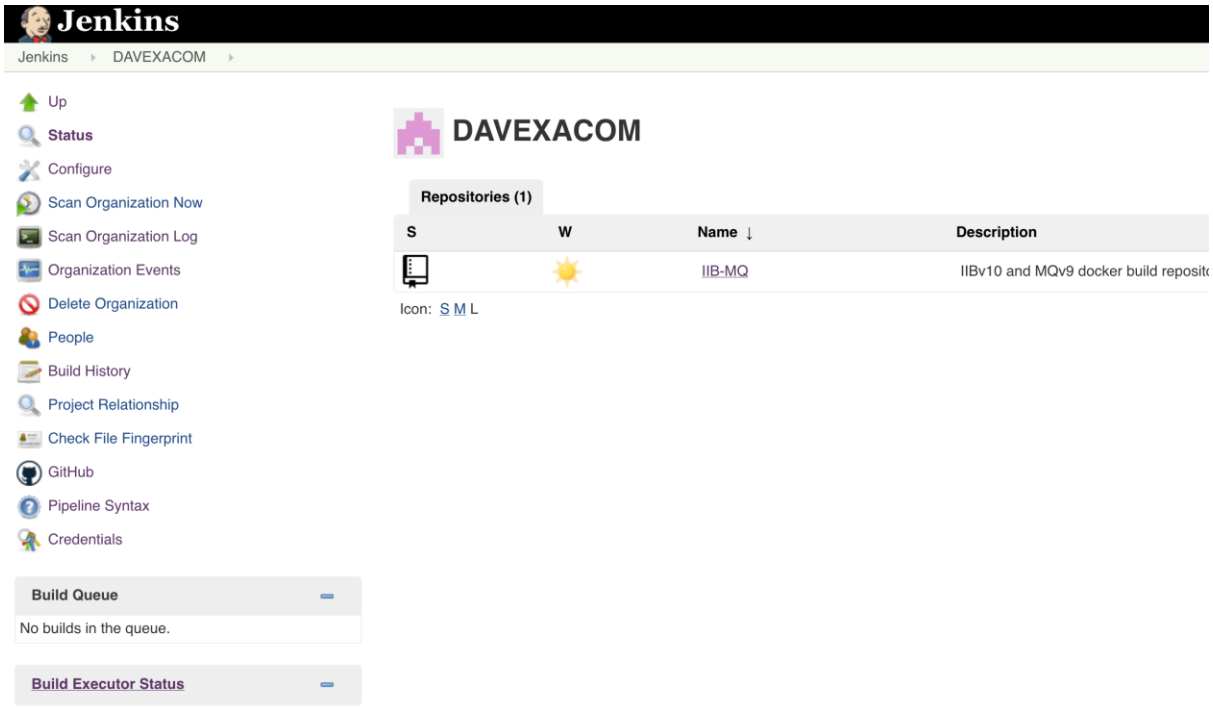
Payload

```
{
  "zen": "Avoid administrative distraction.",
  "hook_id": 19803163,
```

Testing the automated build process

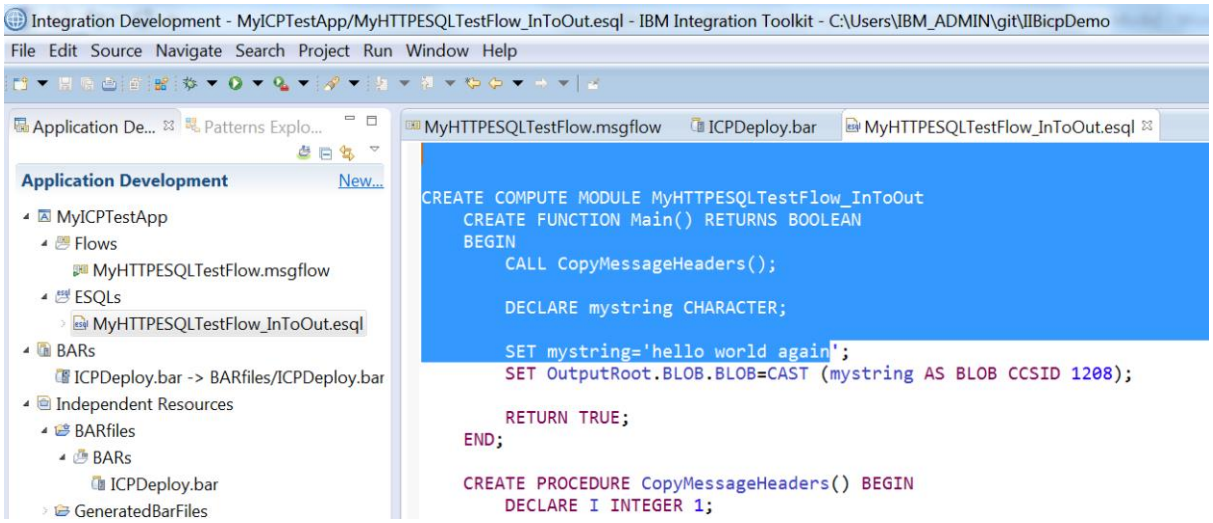
Review Jenkins Status

There should be no builds in the queue at this point

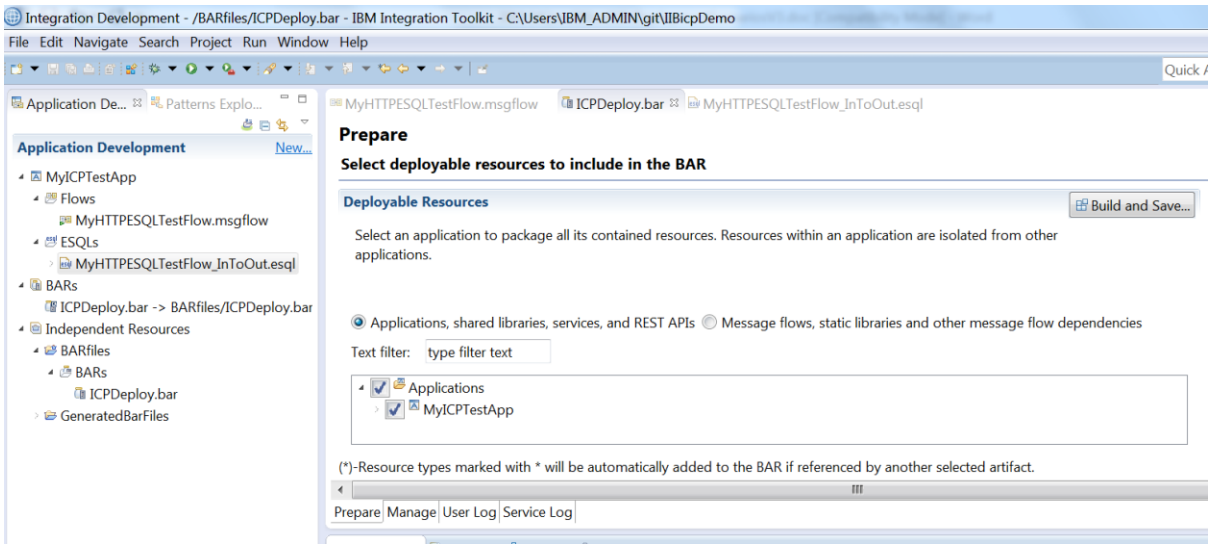


Update the IIB message flow

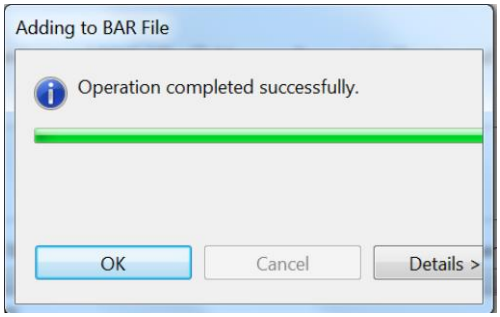
Make a change to the IIB message flow that will be deployed into the ICP IIB runtime as part of the build. Change “Hello world” to “Hello world again” for example.



Change the mystring text to be returned when MyHttpESQLTestFlow is called.



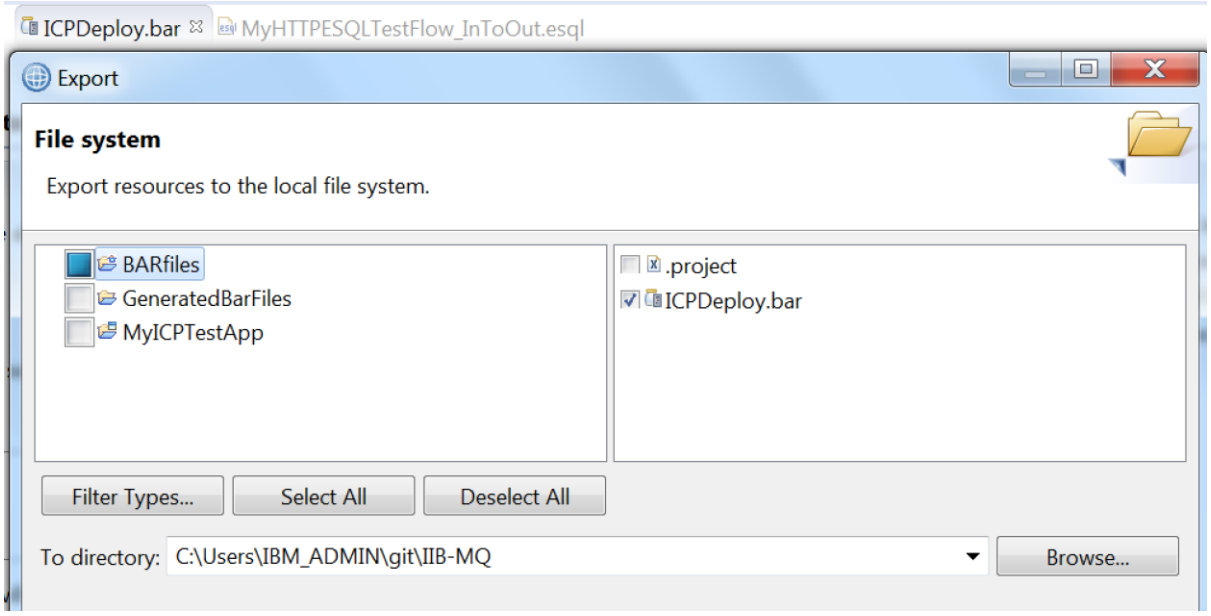
Build and save the BAR file.



You can perform a local deployment and test of the updated bar file if you like.

Export the BAR file to the local clone of IIB-MQ

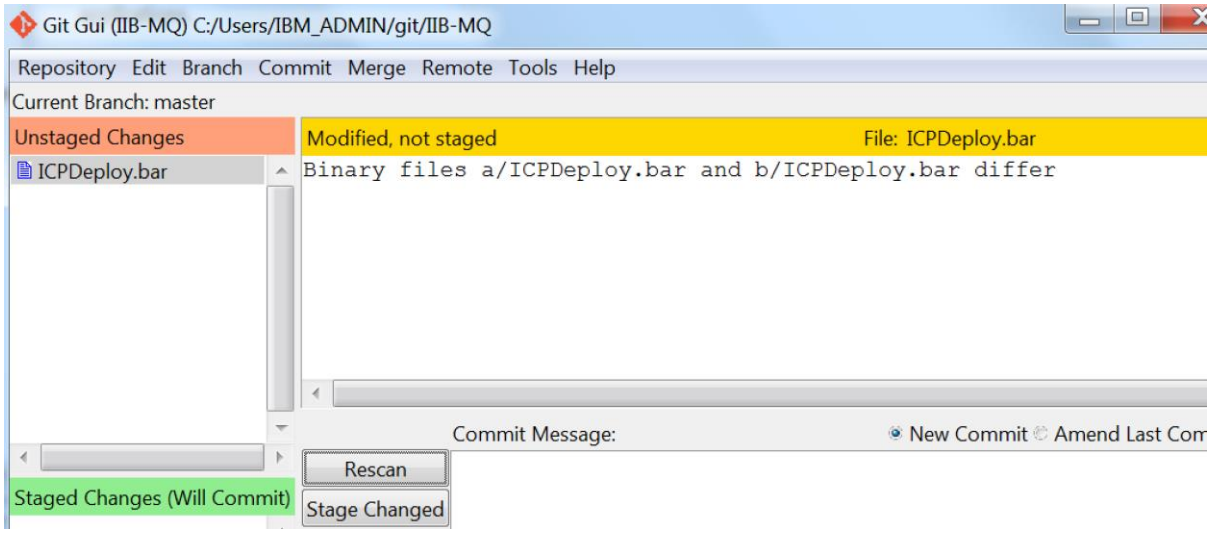
Export the updated BAR file to the File System. File->Export->File System



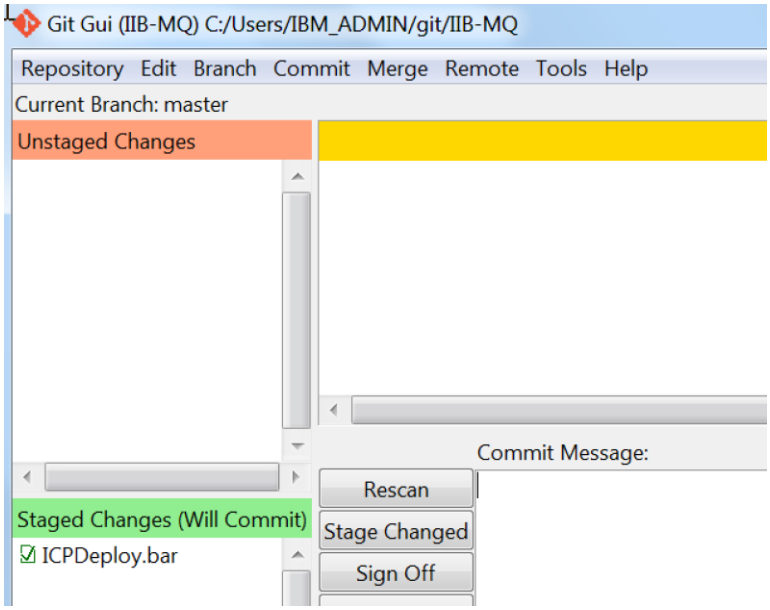
Set the target directory to your local clone of GitHub IIB-MQ repository

Push to GitHub IIB-MQ repository

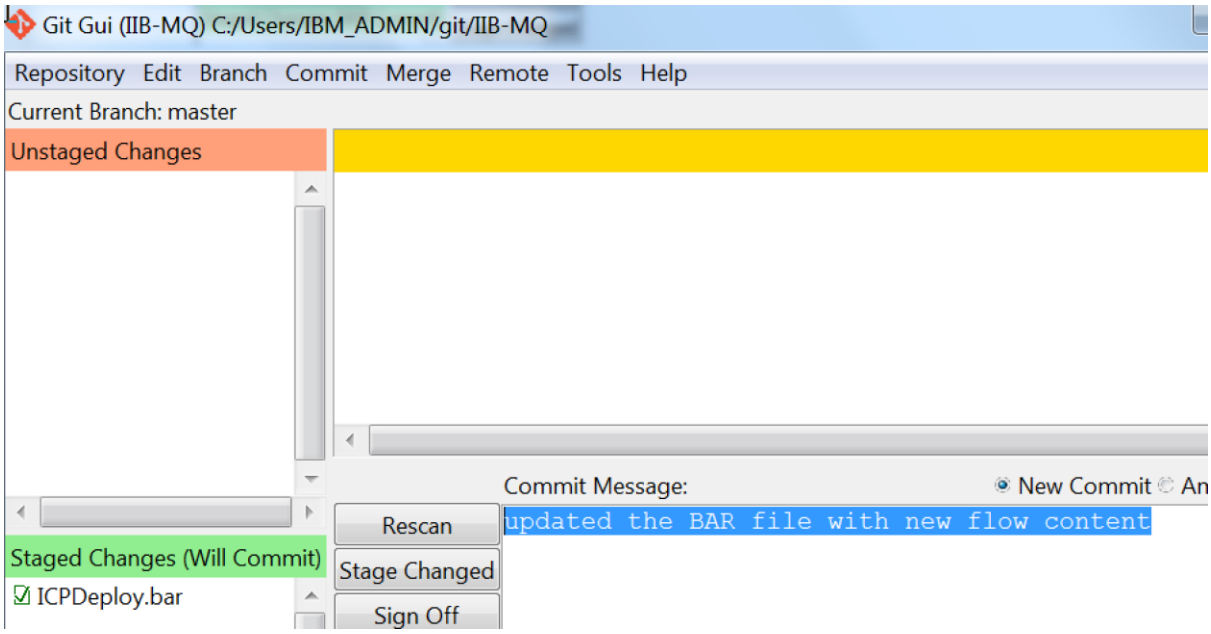
In Git GUI or equivalent hit rescan



Stage changed

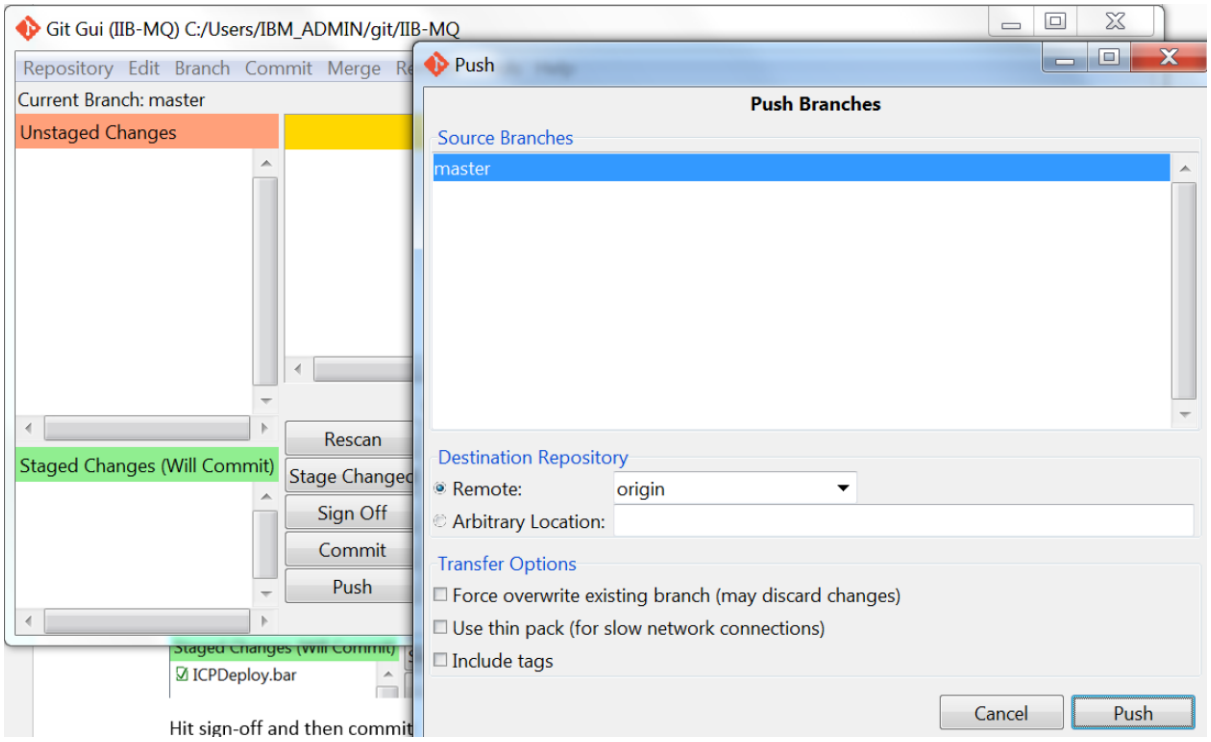


Add a commit message

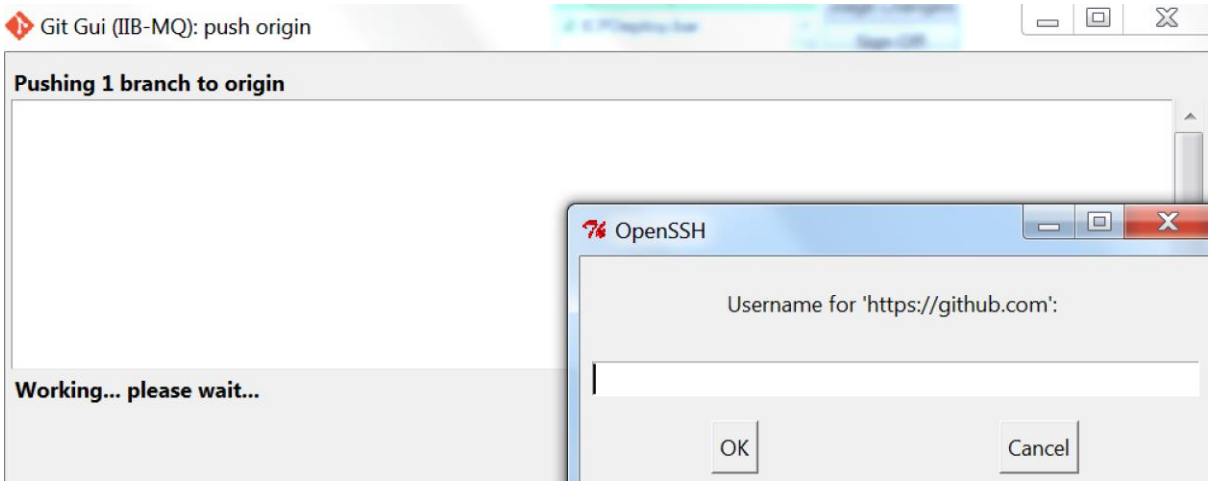


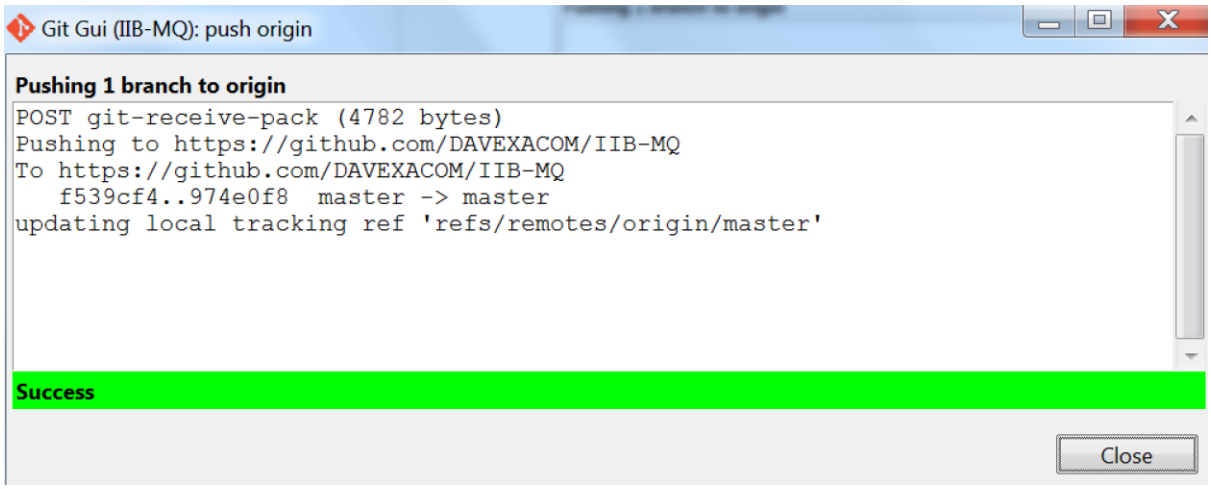
Hit sign-off and then commit

Then start the Push



Sign in with the credentials for your github organization.





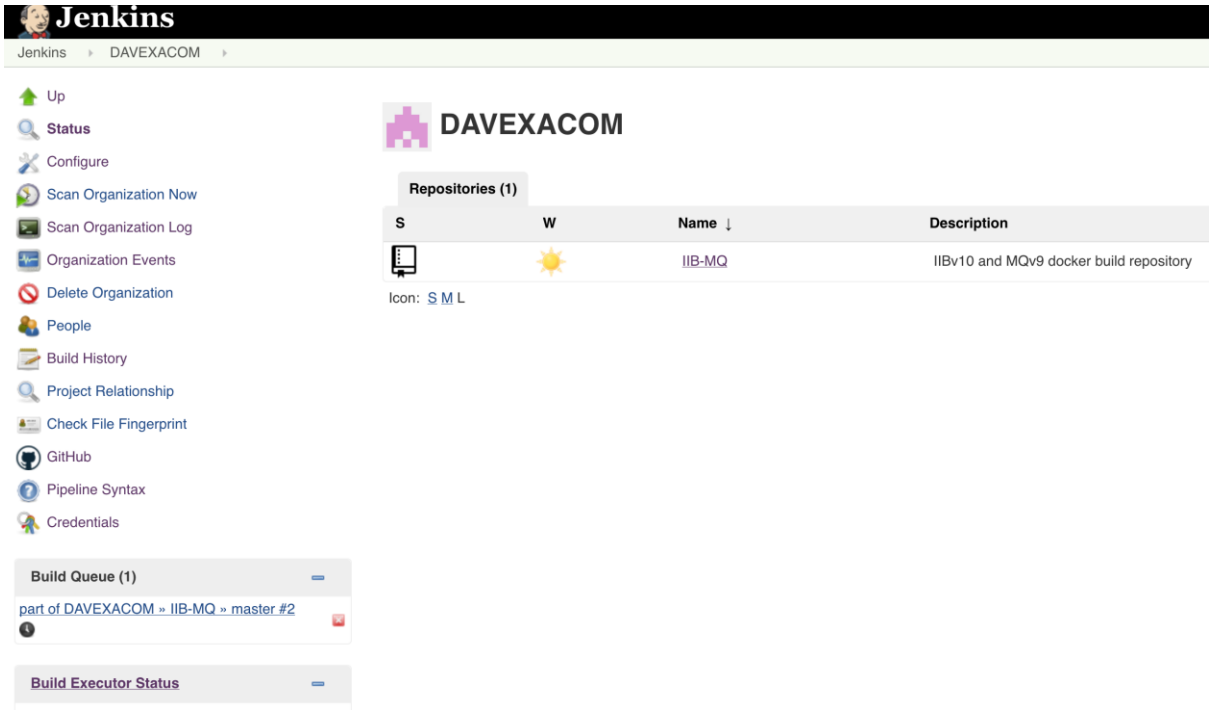
Check the GitHub repository Status

You should see ICPDeploy.bat has been updated

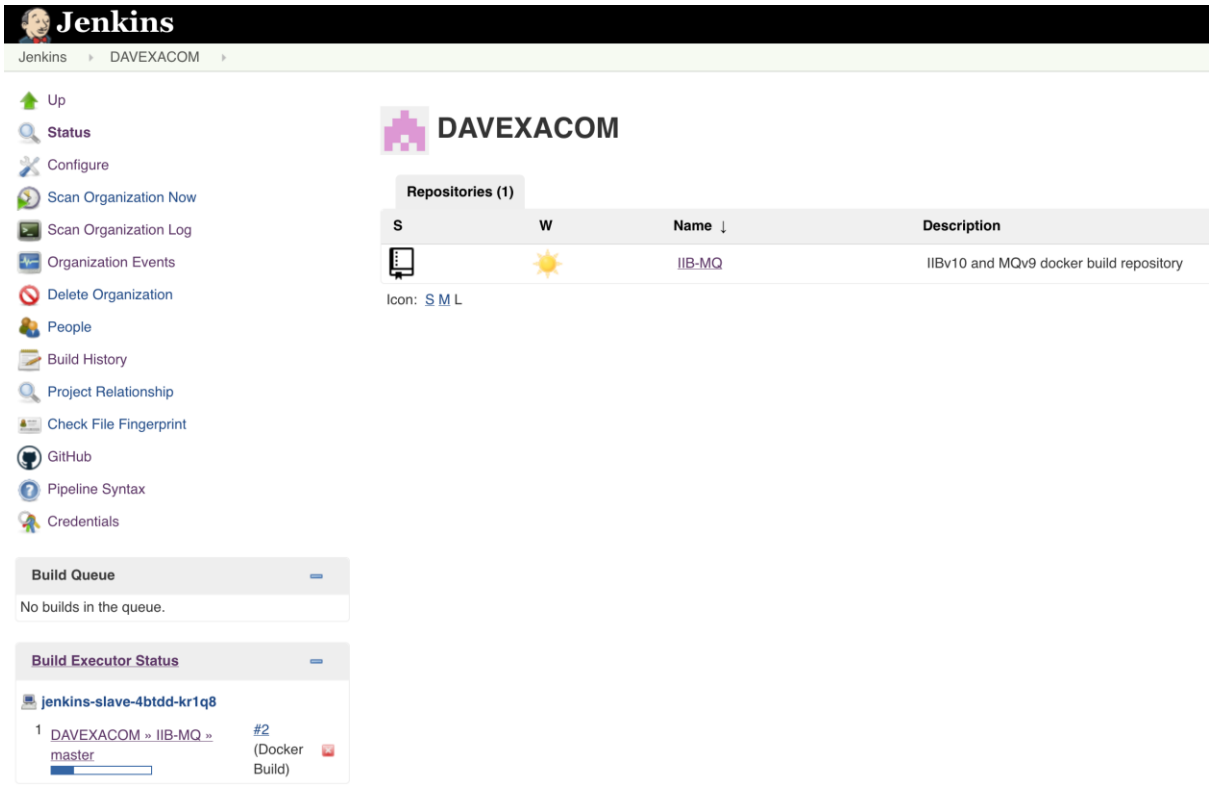
chart/iibmq	Update service.yaml	20 days ago
.cfignore	Initial Commit	9 months ago
.gitignore	Initial Commit	9 months ago
10-listener.mqsc	Initial Commit	9 months ago
CLA.md	Initial Commit	9 months ago
Dockerfile	Update Dockerfile	19 days ago
ICPDeploy.bar	updated the BAR file with new flow content	4 minutes ago

Review Jenkins Status Again

Things may take a moment to kick off

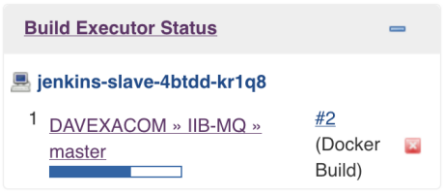


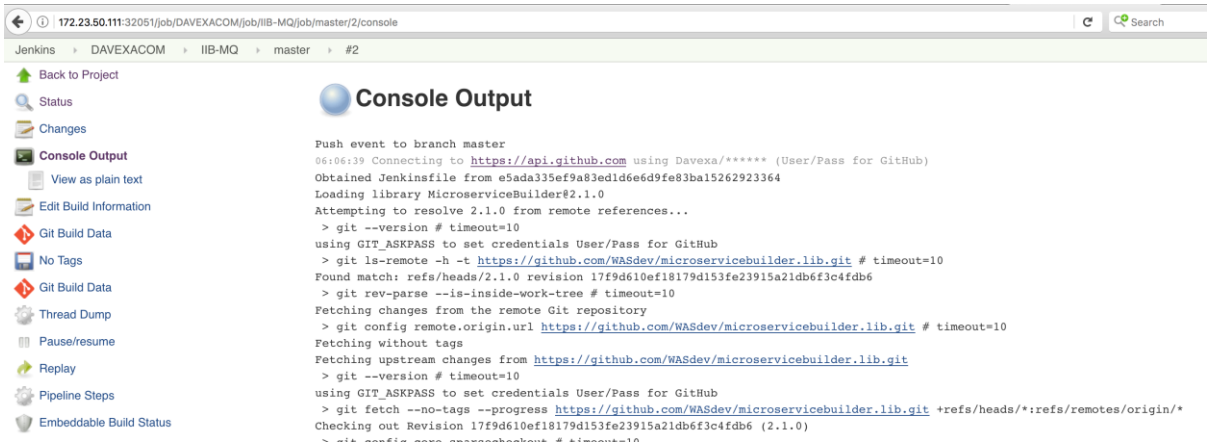
And once an Executor becomes available you should see the following.



Check the Jenkins build logs

Click on the **#n** and then select Console Output



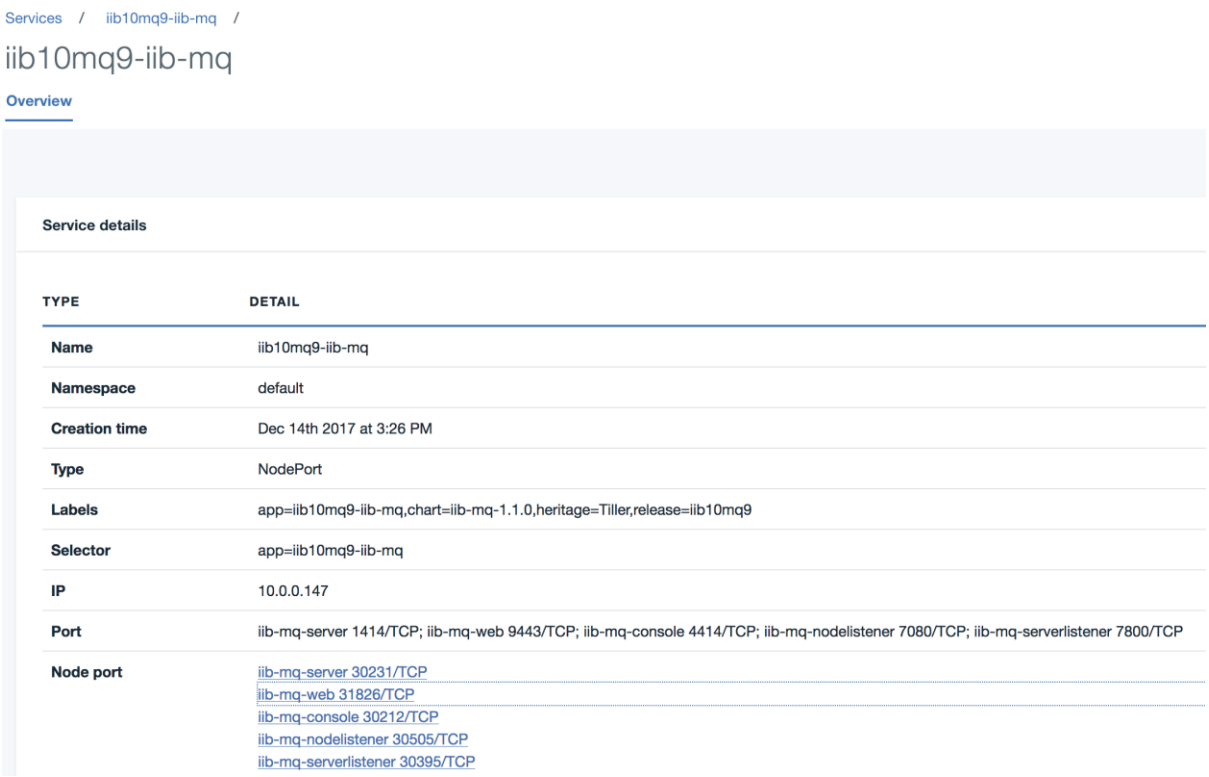


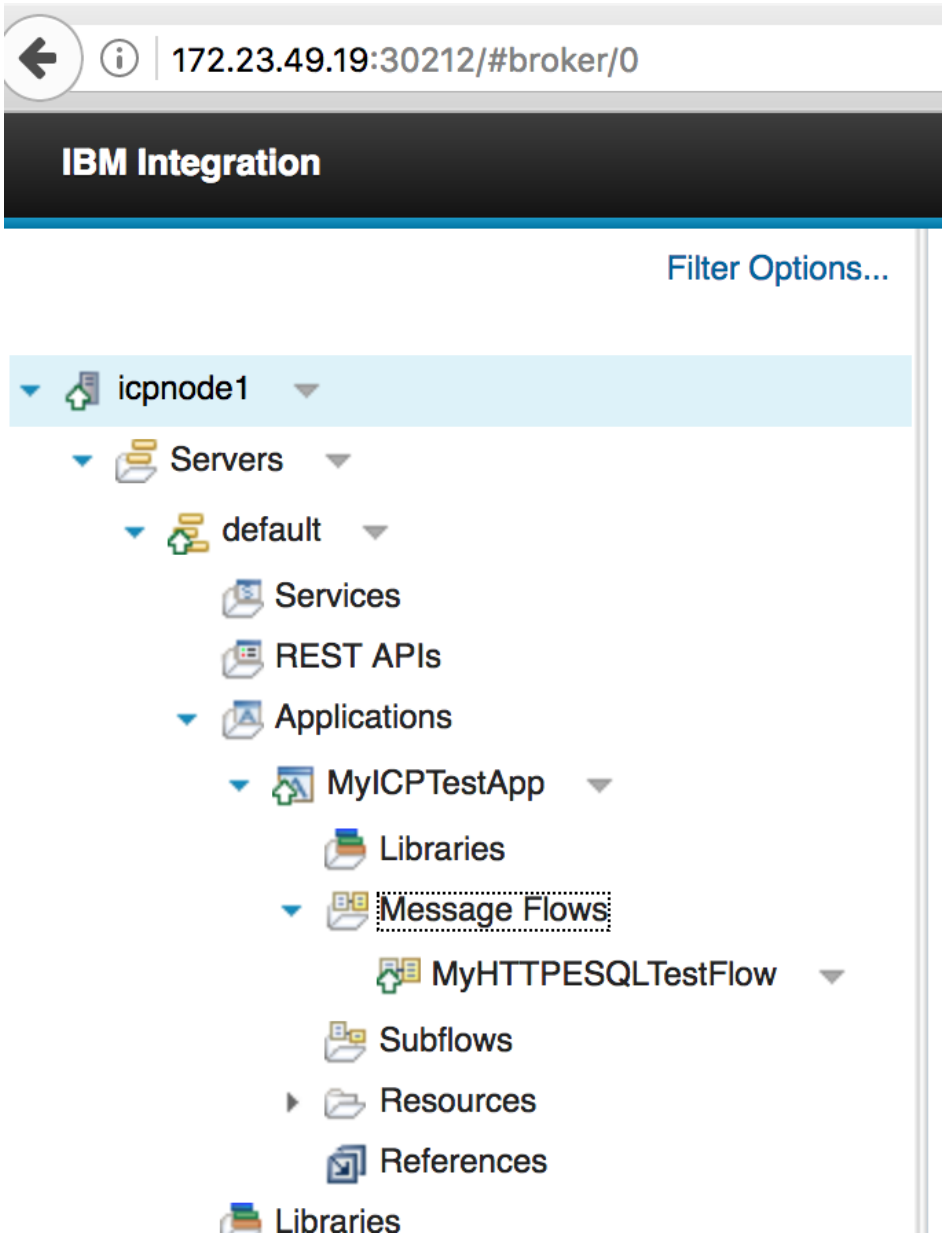
Once complete the tail of the output will look similar to below.

```
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ /helm init --client-only --skip-refresh
Creating /home/jenkins/.helm
Creating /home/jenkins/.helm/repository
Creating /home/jenkins/.helm/repository/cache
Creating /home/jenkins/.helm/repository/local
Creating /home/jenkins/.helm/plugins
Creating /home/jenkins/.helm/starters
Creating /home/jenkins/.helm/cache/archive
Creating /home/jenkins/.helm/repository/repositories.yaml
$HELM_HOME has been configured at /home/jenkins/.helm.
Not installing Tiller due to 'client-only' flag having been set
Happy Helming!
[Pipeline] fileExists
[Pipeline] sh
[DAVEXACOM_IIB-MQ_master-MPOWJ5QPOSBOGPPW2KNKIZRPJQRLX5DN7HQYK7RV5DPBNS2U3S2A] Running shell script
+ /helm upgrade --install --wait --values pipeline.yaml --namespace default iib10mq9 chart/iibmq
Release "iib10mq9" has been upgraded. Happy Helming!
LAST DEPLOYED: Tue Jan  9 06:17:48 2018
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1beta1/StatefulSet
NAME             DESIRED  CURRENT  AGE
iib10mq9-iib-mq  1        1        49m

==> v1/Secret
NAME             TYPE    DATA  AGE
iib10mq9-iib-mq  Opaque  2     49m

==> v1/Service
NAME             CLUSTER-IP  EXTERNAL-IP  PORT(S)                                                                  AGE
iib10mq9-iib-mq  10.0.0.14   <nodes>      1414:31892/TCP,9443:31416/TCP,4414:32681/TCP,7080:30010/TCP,7800:31207/TCP  49m


[Pipeline] }
[Pipeline] // container
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // podTemplate
[Pipeline] End of Pipeline
```

Check the results in the IIB Web UI

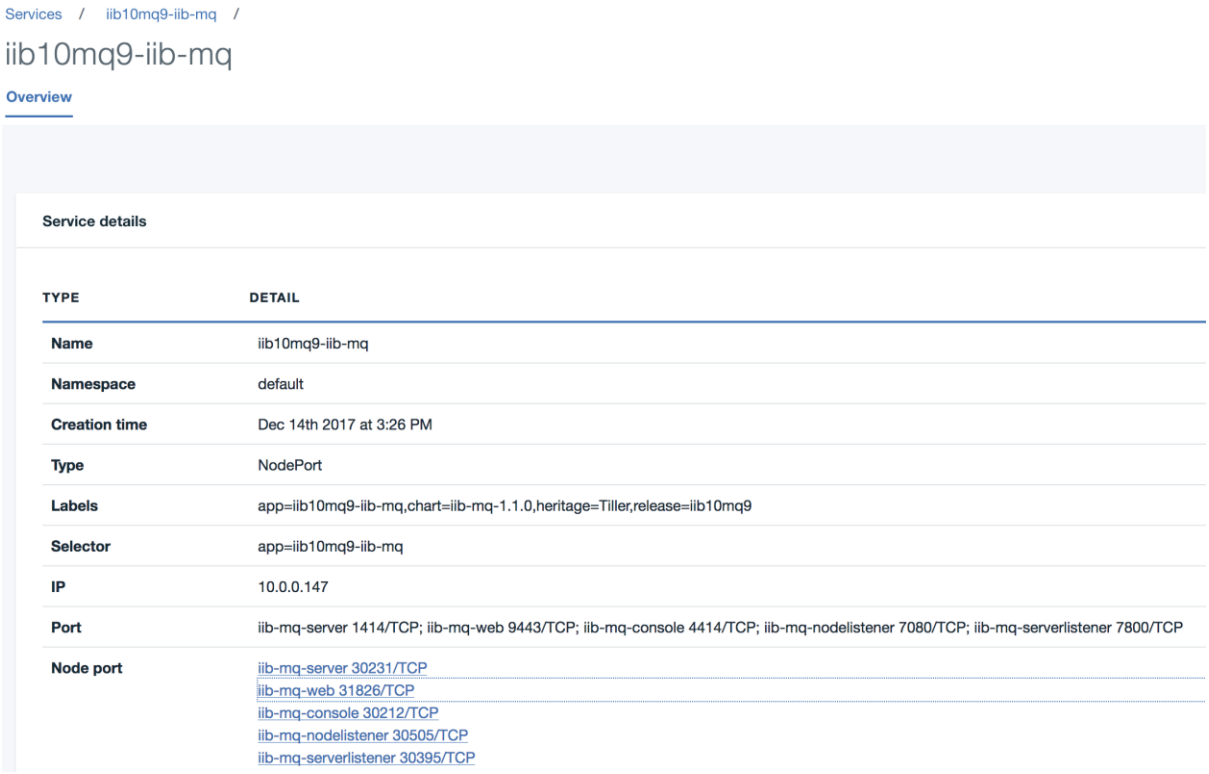
From ICP console -> Workloads -> Helm releases select iib10mq9 then select Services to get to the link to the Web UI (iib-mq-web 31826 in this example)





Test the deployed message flow

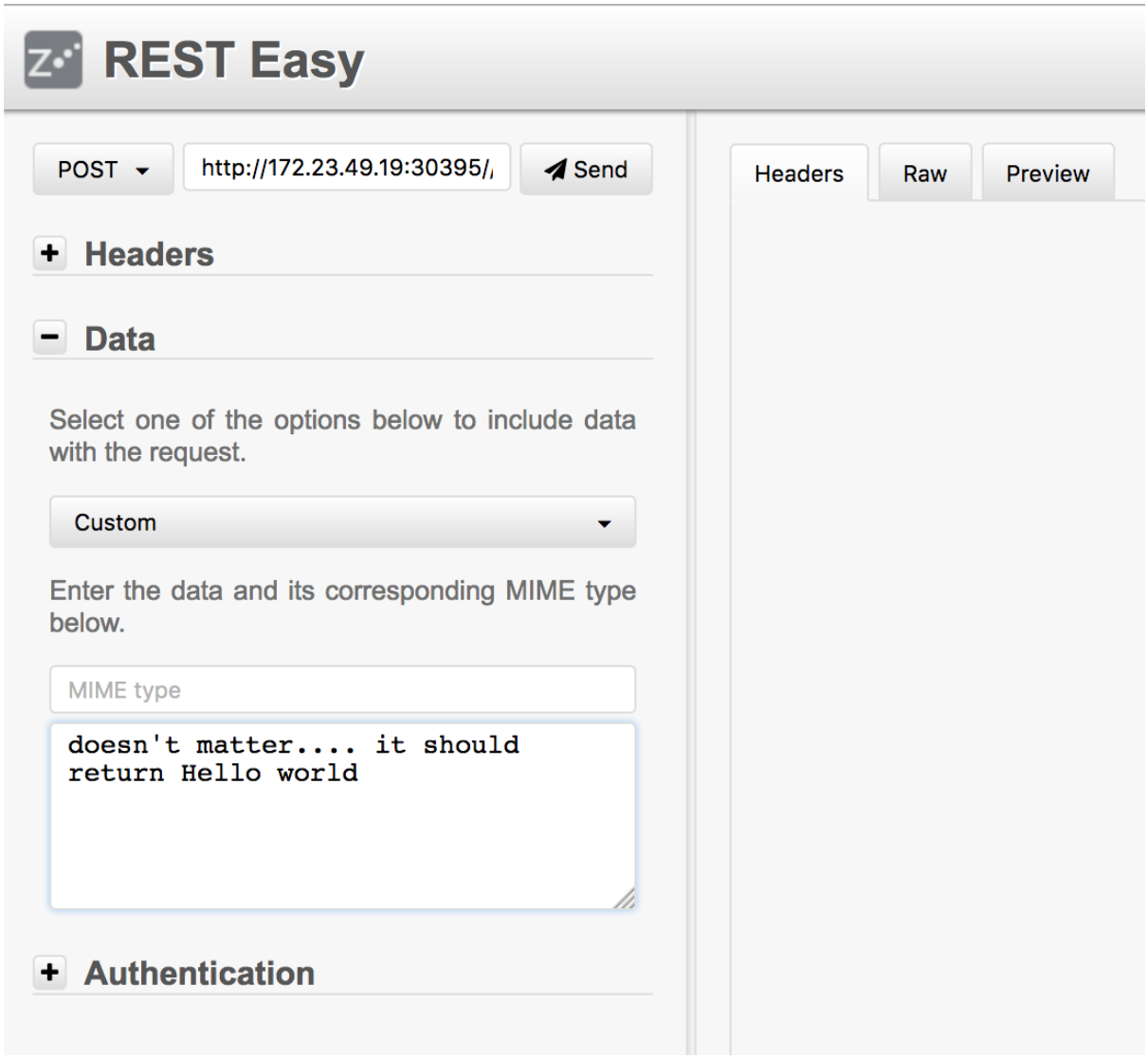
From the ICP Console identify the IIB server listener port.



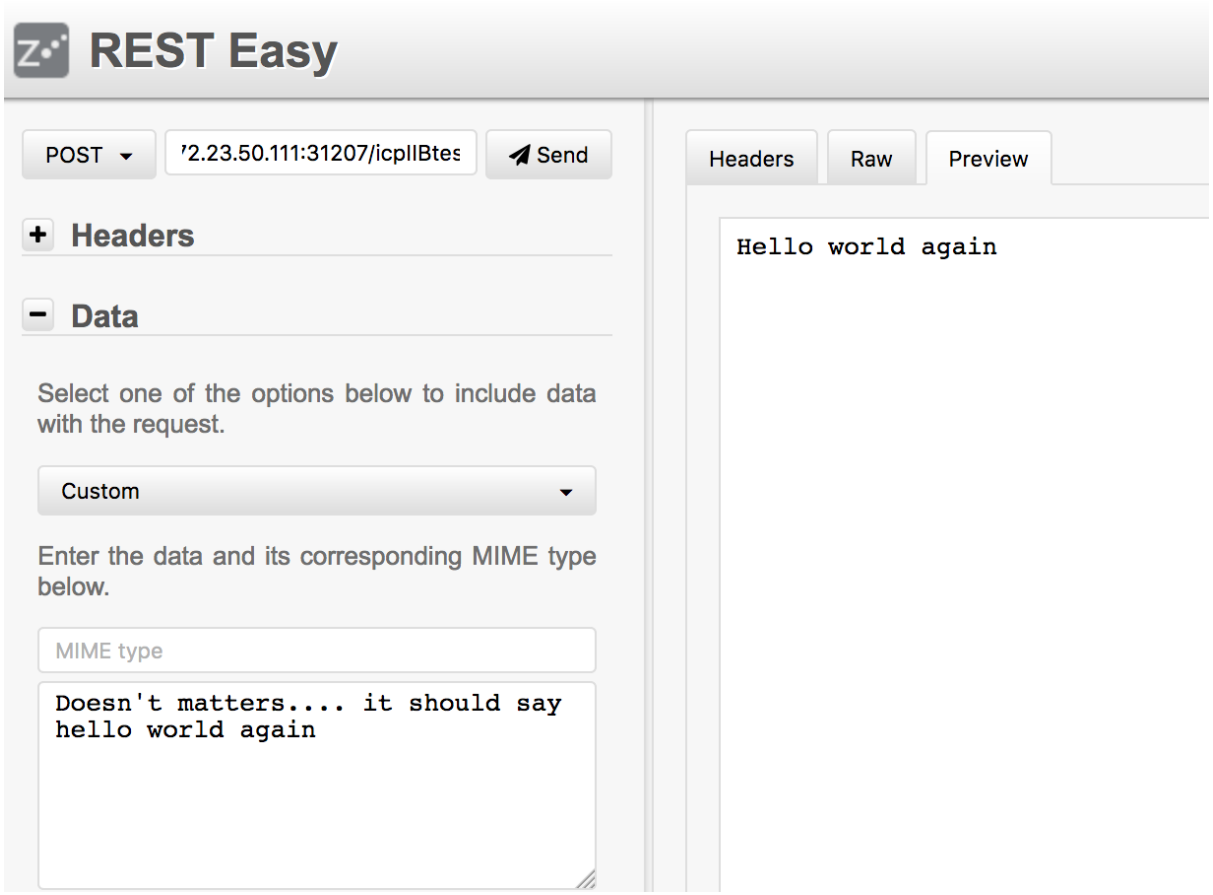
In this example 7800 is represented by 30395

Using an HTTP/REST test client like RestEasy for example

URL: http://172.23.50.111:30395//icpIIBtest



Hit Send and check the result in preview


REST Easy
POST
'2.23.50.111:31207/icpllBtes
Send
Headers
Data
Select one of the options below to include data with the request.
Custom
Enter the data and its corresponding MIME type below.
MIME type
Doesn't matters.... it should say hello world again
Headers
Raw
Preview
Hello world again