Angular On Nginx build configuration for a new openshift application - multiple version support

Introduction

This work was done to upgrade the build process of the msp application to use the latest Jenkins 2 / openshift integration tools. The existing build process for the MSP enrolment application was basically a shell script running as a task inside the Jenkins 1. It a nutshell it performed the following steps in the msp-tools openshift project:

1. Connect to the MSP github page and obtain a clone of a specific branch.
2. Use a node execution environment (v6.9.1 as of the writing of this document).
3. Use a recent version of phatomjs for running tests.
4. A shell script that did:
    a. Copied the environment from /var/lib/jenkins to /tmp/msp-dev
    b. Configure access to both node/npm and phantomjs on the PATH.
    c. Run "npm install" to set up the environment in the /tmp/msp-dev directory.
    d. Run "npm test" in the /tmp/msp-dev directory.
    e. Build the app by running "npm run build" in the /tmp/msp-dev directory.
    f. tag the images build as msp-dev:latest both in the current (tools) openshift and the run time (msp-dev) openshift project.

Once the images are built and tagged, the Build Configuration artifacts in the msp-dev project deployed the image and run the application.

This was the old way.

The new way doesn't use a jenkins shell script, but relies on a Jenkinsfile in the project's top level github repository as well as openshift templates checked it with the msp project in github. This requires Jenkins 2+ and Openshift 3.3+ and uses the following technologies:

- Nginx to serve files and application.
- Uses Openshift's feature called Extended Builds, which lets you build with one s2i image (node 6+), but deploys in another (nginx) for runtime.

There is a division of work:

- Openshift builds the following:
    - Docker images (node itself).
    - s2i images
    - moving the s2i images to runtime.
    - deployment of images in runtime.
- Jenkins does the following:
    - Listens to github pushes (via github hooks).
    - Triggers the build/deploy pipeline.
    - Executes the Jenkinsfile, which does:
        - git checkout
        - triggers openshift to build images
        - triggers images to instruct openshift to deploy.
    - Notifications and repo tagging (via sed commands).

In addition, the methods described below allow for multiple versions of the application to be built and deployed in parallel, so that it will be possible, for example, to have both a 1.2 version and a 1.3 version of the application be deployed to the dev and test environments (but only one version to be deployed in prod).

Github Setup

In order for this to work, the project in github needs, on top of the regular src and nginx structures, the following artifacts:

- A Jenkins file at the top level (called Jenkinsfile), as well as the package.json at the top level.
- A set of templates. These should go to the openshift/templates directory:
    - openshift/templates/angular-builder:
        - angular-builder.json : template to build the angular application
        - Dockerfile : to obtain node distribution (version 6, for example of the centos7 distro).
    - openshift/templates/nginx-runtime:
        - nginx-runtime.json : template to create the nginx runtime environment.
        - Support files: nginx.conf.template, s2i/bin executables to run, assemble, and assemble-runtime.
    - openshift/templates/angular-on-nginx:
        - angular-on-nginx-build.json : to perform the build.
        - angular-on-nginx-deploy.json : to perform the deploy.

Please note that these are just templates, and that before creating the actual objects from them, there will be an opportunity to modify the default values.

Consult the Appendix sections below for the content of each file (in this case for the MSP project, but with the exception of the github URLs, all other fields should remain the same for all projects).

Instructions

Make sure that github area is setup with the angular code (ie. you can run "npm install", "npm run build" in a clone of it). Also make sure all the templates (Appendix A to Appendix K) are in the github repo in the proper places. The following instructions set up two versions of the deployables: 1.2 and 1.3, but the principles apply to any version and any number of them.

1. Create a clone of the github repo from the correct reference,

Or go to your cloned repo in your local drive.

Checkout the version you want to work with (for example 1.2):

- git checkout 1.2
- git fetch
- git rebase

2. Setup angular-builder for version 1.2

This step creates a "Builder Image" that compiles the angular source.

- Open the openshift console and go to your project. In our case the project name is gcpe-mygovbc-msp-tools.
- This should be an empty project at this stage. You should see "Add to Project" bar button.
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-builder/angular-builder.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the Name to "angular-builder-1.2".**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "angular-builder" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - The Dockerfile from openshift/templates/angular-builder/Dockerfile has been invoked (ie. node v.6 is installed).
    - A new angular-builder image is created in your image streams.

3. Setup nginx-runtime for version 1.2

- Open the openshift console and go to your project (gcpe-mygovbc-msp-tools).
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/nginx-runtime/nginx-runtime.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "nginx-runtime" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - The Dockerfile from openshift/templates/angular-builder/Dockerfile has been invoked (ie. nginx is installed).
    - A new nginx-runtime image is created in your image streams.

4. Setup angular-on-nginx for version 1.2

- Open the openshift console and go to your project (gcpe-mygovbc-msp-tools).
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-on-nginx/angular-on-nginx-build.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the "Name" to the name of the application, in this case "msp-1.2".**
- **Change the Git Reference to the branch name, in this case "1.2".**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "**msp-1.2-build**" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - angular-builder is triggered to build the source code.
    - The application msp-1.2-build image is created in your image streams.

5. Setup the msp deployment for version 1.2

Typically, this is done in a deployment project (for example gcpe-mygovbc-msp-dev). Go to this project in the openshift console.

- Open the openshift console and go to your project (gcpe-mygovbc-msp-dev).
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-on-nginx/angular-on-nginx-deploy.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the "Source Name" to the name of the source image, in this case "msp-1.2".**
- **Change the "Name" to the name of the application, Cannot use '.' as it's not allowed, make sure to use "msp-1-2".**
- **Change the "Image Namespace" to the project where it's built, ie. "gcpe-mygovbc-msp-tools".**
- **Leave the OpenShift Cluster IP Range as the default value (172.51.0.0/16).**
- **Add the Additional real_ip_from Rules. This varies in different environments. For dev deployments, the value is:**
  set_real_ip_from 142.34.208.209; set_real_ip_from 142.34.63.248/29;
- **Leave the Ip Filter Rules as default for dev deployment, the value is:**
  allow 0.0.0.0/0; deny all;
- **Set the Application Hostname, for dev this should be "gcpe-mygovbc-msp-1-2-dev.pathfinder.gov.bc.ca" (notice the '-' instead of '.').**
- **Change the "Env TAG name" to suit (ie. dev, or test or prod), for dev: "dev".**
- **Leave HTTP Basic Nginx Config Line, Username and Password blank (default values).**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "angular-on-nginx" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - angular-builder is triggered to build the source code.
    - The output (/opt/app-root/src/dist/) is copied to the nginx-runtime directory (/tmp/app).
    - The application msp-build image is created in your image streams.

6. Install Jenkins pipeline for version 1.2

OpenShift has a special BuildConfiguration strategy - JenkinsPipeline - that integrates OpenShift deeply with Jenkins, providing triggering and tracking of pipeline execution from within the OpenShift web console.

A specifically configured and named Jenkins instance is required within your project in order for the Jenkins<->OpenShift pipeline synchronization to work. This instance can be provisioned explicitly (as explained in #ProvisioningJenkins, below), or it will be provisioned "lazily" upon creation of the first JenkinsPipeline BuildConfig within a project. The latter is the recommended approach.

When a JenkinsPipeline BuildConfig is created, it will cause a pipeline resource to be added to the OpenShift UI, as well as adding a pipeline to Jenkins. The state of the pipeline will be synchronized between Jenkins and OpenShift. This is achieved through direct support in OpenShift and the "OpenShift Sync" plugin on the Jenkins side.

Follow the steps below to create a BuildConfiguration with a JenkinsPipeline build strategy, pointing to the Jenkinsfile you created above. A new templates is needed for the creation of the pipeline. Please consult Appendix L for this. In the msp github project, this template is located in:

openshift/templates/jenkins2-templates/msp-jenkins2-template.json

- Go to the Openshift console.
- In the top right corner, you'll see the symbol ? in a circle with a down arrow next to it. Click on the down arrow.
- Select Command Line Tools.
- The first box has a "oc login https/....". To the very right on it there is a copy to clipboard button. Click on it.
- Open a terminal. Paste the contents of the clipboard and execute the login command.
- Change your project to the appropriate place (ie. oc project gcpe-mygovbc-msp-tools).
- Now go to the directory where you cloned the msp github repo. Then navigate to:
    - openshift/templates/jenkins2-templates/
- Execute the following command:
    - oc process -f msp-jenkins2-template.json | oc create -f -

Once this executes, you will have a new Jenkins2 pipeline created in your project. To access it:

- Go to the Openshift console.
- Go to the appropriate project (ie. project gcpe-mygovbc-msp-tools).
- Click on BuildsPipelines.
- You should see the msp-1.2-pipeline.
- Click on Start Pipeline.
- Once the pipeline finishes, you can view it (logs are broken from this screen) and should see:
    - Build #1 running (it will go all the way to deploying to dev but will stop waiting for input to push to test).
    - Build #1 will have the following steps:
        1. build nginx runtime.
        2. build msp-build.
        3. deploy-dev.
        4. Waiting on user input to deploy-test.

7. Jenkins configuration

To access the Ad-hoc Jenkins that was installed:

- Go to the Openshift console.
- Go to the appropriate project (ie. project gcpe-mygovbc-msp-tools).
- Click on ApplicationsRoutes.
- You should see one entry for the application itself (ie. msp), and another entry for the Jenkins Pipelines called jenkins-pipeline-svc.
- Next to jenkins-pipeline-svc, the next field titled Hostname, is the link to jenkins. In our case it will be something like: https://jenkins-pipeline-svc-gcpe-mygovbc-msp-tools.pathfinder.gov.bc.ca (depending on your projects, etc..).
- You can click on it (and/or bookmark this for future reference).

8. Switch to another branch

Checkout the next version you want to work with (for example 1.3):

- git checkout 1.3
- git fetch
- git rebase

9. Setup angular-builder for version 1.3

This step creates a "Builder Image" that compiles the angular source.

- Open the openshift console and go to your project. In our case the project name is gcpe-mygovbc-msp-tools.
- This should be an empty project at this stage. You should see "Add to Project" bar button.
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-builder/angular-builder.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the Name to "angular-builder-1.3".**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "angular-builder" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - The Dockerfile from openshift/templates/angular-builder/Dockerfile has been invoked (ie. node v.6 is installed).
    - A new angular-builder image is created in your image streams.

10. Skip the setup for nginx-runtime, it will the the same as 1.2.

11. Setup angular-on-nginx for version 1.3

- Open the openshift console and go to your project (gcpe-mygovbc-msp-tools).
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-on-nginx/angular-on-nginx-build.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the "Name" to the name of the application, in this case "msp-1.3".**
- **Change the Git Reference to the branch name, in this case "1.3".**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "**msp-1.3-build**" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - angular-builder is triggered to build the source code.
    - The application msp-1.3-build image is created in your image streams.

5. Setup the msp deployment for version 1.3

Typically, this is done in a deployment project (for example gcpe-mygovbc-msp-dev). Go to this project in the openshift console.

- Open the openshift console and go to your project (gcpe-mygovbc-msp-dev).
- Add to Project Import YAML/JSON
- On a terminal, cat the content of openshift/templates/angular-on-nginx/angular-on-nginx-deploy.json. Select all of it.
- Click on the big square clipboard (not the one that has a Browse button) and paste the content (use Ctrl-V, the paste button doesn't work).
- Click on Create at the bottom.
- On the Add Template pop-up, have "Process the template" selected, and click Continue.
- You will be given the opportunity to change configurable parameters in the next screen. Make sure the github repo is yours and contains all the templates.
- **Change the "Source Name" to the name of the source image, in this case "msp-1.3".**
- **Change the "Name" to the name of the application, Cannot use '.' as it's not allowed, make sure to use "msp-1-3".**
- **Change the "Image Namespace" to the project where it's built, ie. "gcpe-mygovbc-msp-tools".**
- **Leave the OpenShift Cluster IP Range as the default value (172.51.0.0/16).**
- **Add the Additional real_ip_from Rules. This varies in different environments. For dev deployments, the value is:**
  set_real_ip_from 142.34.208.209; set_real_ip_from 142.34.63.248/29;
- **Leave the Ip Filter Rules as default for dev deployment, the value is:**
  allow 0.0.0.0/0; deny all;
- **Set the Application Hostname, for dev this should be "gcpe-mygovbc-msp-1-3-dev.pathfinder.gov.bc.ca" (notice the '-' instead of '.').**
- **Change the "Env TAG name" to suit (ie. dev, or test or prod), for dev: "dev".**
- **Leave HTTP Basic Nginx Config Line, Username and Password blank (default values).**
- Create it.
- Go back to the openshift console, select BuildsBuilds.
- You should see the "angular-on-nginx" build. Select it, and start the build.
- You can click on the build number and follow or view the log.
- Once this build is successful:
    - angular-builder is triggered to build the source code.
    - The output (/opt/app-root/src/dist/) is copied to the nginx-runtime directory (/tmp/app).
    - The application msp-build image is created in your image streams.

6. Install Jenkins pipeline for version 1.3

Follow the steps below to create a BuildConfiguration with a JenkinsPipeline build strategy, pointing to the Jenkinsfile you created above. A new templates is needed for the creation of the pipeline. **The key here is that new jenkins template (coming from a different github branch) specifies a different name for the pipeline** (with a 1-3 suffix instead of 1-2). Please consult Appendix L for this. In the msp github project, this template is located in:

openshift/templates/jenkins2-templates/msp-jenkins2-template.json

- Go to the Openshift console.
- In the top right corner, you'll see the symbol ? in a circle with a down arrow next to it. Click on the down arrow.
- Select Command Line Tools.

- The first box has a "oc login https/....". To the very right on it there is a copy to clipboard button. Click on it.
- Open a terminal. Paste the contents of the clipboard and execute the login command.
- Change your project to the appropriate place (ie. oc project gcpe-mygovbc-msp-tools).
- Now go to the directory where you cloned the msp github repo. Then navigate to:
    - openshift/templates/jenkins2-templates/
- Execute the following command:
    - oc process -f msp-jenkins2-template.json | oc create -f -

Once this executes, you will have a new Jenkins2 pipeline created in your project. To access it:

- Go to the Openshift console.
- Go to the appropriate project (ie. project gcpe-mygovbc-msp-tools).
- Click on BuildsPipelines.
- You should see the msp-1.3-pipeline.
- Click on Start Pipeline.
- Once the pipeline finishes, you can view it (logs are broken from this screen) and should see:
    - Build #1 running (it will go all the way to deploying to dev but will stop waiting for input to push to test).
    - Build #1 will have the following steps:
        1. build nginx runtime.
        2. build msp-build.
        3. deploy-dev.
        4. Waiting on user input to deploy-test.

Build Process

The build process should be automatic upon any check-ins to the github source (in the branch specified by SOURCE_REPOSITORY_REF, ie. 1.2).

To view it:

- Go to the jenkins pipeline addressed as described in the Jenkins configuration above.
- Log in as admin. The password can be obtained by the following method:
    - Go to your tools build project in openshift.
    - Find the jenkins-pipeline-svc section, and under it its Deployment Config jenkins-pipeline-svc. Click on the Deployment Config.
    - Go to the Environment tab.
    - Cut and paste the JENKINS_PASSWORD.
- Once you're logged into Jenkins, you should see the Build #1 in progress.
- Click on it, it will take you to the pipeline view.
- Now you can hover over any pipeline stage and should be able to access its execution log.
- If you want to deploy to test, hover over deploy-test and select "Proceed". This of course pre-requires that you have set up an appropriate test project with the correct Build Configurations for msp.

PS: Don't forget to Abort to proceed to test. Otherwise, these threads will linger.

Appendix A: Jenkinsfile

Jenkinsfile version 1.2

Jenkinsfile version 1.3

Appendix B : openshift/templates/angular-builder/angular-builder.json

angular-builder version 1.2

angular-builder version 1.3

Appendix C: openshift/templates/angular-builder/Dockerfile

```
# Get community edition of nodejs v6.x
FROM centos/nodejs-6-centos7
```

Appendix D: openshift/templates/nginx-runtime/nginx-runtime.json

nginx-runtime version 1.2

nginx-runtime version 1.3

Appendix E : openshift/templates/nginx-runtime/Dockerfile

Dockerfile version 1.2

Dockerfile version 1.3

Appendix F: openshift/templates/nginx-runtime/nginx.conf.template

nginx.conf.template version 1.2

nginx.conf.template version 1.3

Appendix G: openshift/templates/nginx-runtime/s2i/bin/assemble

```
#!/bin/bash
echo no assemble needed
```

Appendix H: openshift/templates/nginx-runtime/s2i/bin/assemble-runtime

```
#!/bin/bash
echo no assemble-runtime needed
```

Appendix I: openshift/templates/nginx-runtime/s2i/bin/run

```
#!/bin/bash

echo "---> replacing configuration"
sed "s~%RealIpFrom%~${RealIpFrom:-172.51.0.0/16}~g; s~%IpFilterRules%~${IpFilterRules}~g;
s~%AdditionalRealIpFromRules%~${AdditionalRealIpFromRules}~g; s~%HTTP_BASIC%~${HTTP_BASIC}~g" /tmp/nginx.conf.template >
/etc/nginx/nginx.conf

if [ -n "$HTTP_BASIC_USERNAME" ] && [ -n "$HTTP_BASIC_PASSWORD" ]; then
echo "---> Generating .htpasswd file"
`echo "$HTTP_BASIC_USERNAME:$(openssl passwd -crypt $HTTP_BASIC_PASSWORD)" > /tmp/.htpasswd`
fi

echo "---> starting nginx"
/usr/sbin/nginx -g "daemon off;"
```

Appendix J: openshift/templates/angular-on-nginx/angular-on-nginx-build.json

angular-on-nginx-build.json version 1.2

angular-on-nginx-build.json version 1.3

Appendix K: openshift/templates/angular-on-nginx/angular-on-nginx-deploy.json

angular-on-nginx-deploy.json version 1.2

angular-on-nginx-deploy.json version 1.3

Appendix L : openshift/templates/jenkins2-templates/msp-jenkins2-template.json

msp-jenkins2-template.json version 1.2

msp-jenkins2-template.json version 1.3