
Hangar Documentation

Release 1,0

Stuart Davidson

Apr 16, 2017

Contents

1	User Guide	3
1.1	User Guide - Java	3
2	Installation	5
3	Contributors Guide	7
3.1	Acceptance Testing	7
4	About	9

Welcome to Hangar!

Hangar is an open-source, distributed, container-based artefact management system.

So you've been asked to use an instance of Hangar for your artifacts? Great! Congratulations, you have someone in your company with great taste and you should buy them cake.

User Guide - Java

Starting to use Hangar as your Artifact Store is simple.

Maven - Download

To use Hangar as your main repository, add the following directive to your *settings.xml*

```
<mirrors>
  <mirror>
    <id>hangar</id>
    <url><your-hangar-url>/java</url>
    <mirrorOf>central</mirrorOf>
  </mirror>
</mirrors>
```

where *<your-hangar-url>* is the URL given to you by your friendly system administrator for where it's running. Simple!

Maven - Upload

Obviously, you've written something incredible and you want to push your new artifact back into Hangar. Add the following directive into your *pom.xml*

```
<distributionManagement>
  <snapshotRepository>
    <id>hangar-snapshots</id>
```

```
<url>http://<your-hangar-url>/java/snapshots</url>
</snapshotRepository>
<repository>
  <id>hangar-releases</id>
  <url>http://<your-hangar-url>/java/releases</url>
</repository>
</distributionManagement>
```

where *<your-hangar-url>* is the URL given to you by your friendly system administrator for where it's running. Simple!

CHAPTER 2

Installation

Each team, squad, mob will have a different way of wanting to approach artifact management - and so they should! It's a wonderful diverse world after all!

With this ethos in mind, there are a variety of ways of running Hangar.

- Dirty Hack

You want to just give it a go? See if it works with what you want to do? Run it in a container and have it disappear once you are done!

- All Local

In this format, we have a single instance with local storage for artifacts and an in-memory index. Handy for debugging or just giving it a try.

- S3 Local

In this format the index is still held in memory, but the storage of artifacts is in S3.

- S3 Fully Distributed

In this format, we have multiple API hosts with a shared zookeeper based index and S3 storage.

So you want to contribute to this project? Thank you! Even for opening the documentation to figure out how, you've already taken a big step.

This guide is here to help you contribute as effectively as possible. If there's something missing, please raise an [issue](#) or have a go at adding it yourself.

Acceptance Testing


To improve confidence in the system, there are several containers that will execute a range of commands on tools that would use this tool (pip, maven etc) and then confirm the output.

Building the Containers

As an example, let's build the Maven acceptance test container, called hangar-test-maven.

1. Navigate to `/etc/docker/hangar-test-maven` within the project
2. Run `docker build -t hangar-test-maven .` (note the dot to specify the local directory)

With the container built, you would want to specify the IP and Port of the instance of Hangar you want to test against.

```
docker@default:~$ docker run --rm -e HANGAR_IP='10.0.2.2' -e HANGAR_PORT='8080'  hangar-test-maven
Hangar Integration Test - Maven
-----
Scenario 1 - Package Snapshot... [OK - 76s]
---- System Tests
1..1
ok 1 Compiled target jar exists
Scenario 2 - Deploy Snapshot.... [OK - 8s]
1..1
ok 1 Compiled target jar exists
Scenario 3 - Deploy Release.... [OK - 6s]
```

```
1..1
ok 1 Compiled target jar exists
Scenario 4 - Get custom artifacts [OK - 76s]
```

Note: In the instance above, we're testing this from within docker-machine running on Windows using VirtualBox to a instance of Hangar running within the Eclipse IDE, local to my machine. To do this, we pass in the Gateway IP which is normally 10.0.2.2 - which will route this traffic to be localhost on the parent instance. Handy!

CHAPTER 4

About
