High Intensity Software Delivery

Part One - Setting up a Build Pipeline

In this exercise we will set up an automated build pipeline for a very simple web application. We'll use a set of tools that are available for free online: GitHub, TravisCI and Heroku.

The instructions below are deliberately left a little vague, so you'll need to look at the online documentation to figure out some of the details, but do ask for help if you get stuck.

First **choose a language**, we have prepared skeleton code in **Java**, **Python** and **JavaScript** (NodeJS). Follow the links below to go to the relevant GitHub repository:

https://github.com/rchatley/skeleton-java-app
https://github.com/rchatley/skeleton-python-app
https://github.com/rchatley/skeleton-nodejs-app

Fork this repo into your own GitHub account to create a copy that you can push to.
e.g. : https://github.com/your-github-id/skeleton-java-app

Clone your fork to get the code onto your local machine. See the ReadMe file in the repository to see how to run the application locally on your machine, and how to run the tests.

When it is running on your machine you should be able to make a request from your browser and get a response from your app, e.g. try

http://localgost:5000/

and then
http://localhost:5000/api?q=Romeo

Browse through the source code to make sure you can find the code that processes the query, and the unit tests that check the behaviour.

Setting up an automated build in Travis:

We will use Travis CI to set up a build server to test your code each time you push a change. Connect your new GitHub repository to Travis CI at https://travis-ci.org (you will need to authorise Travis to connect to GitHub to find your repository). Make sure that if you push a change to GitHub then it triggers a build on Travis. The skeleton repo will already include a basic .travis.yml config file to define the build process.

On the next page are instructions on deploying our app to run in the cloud.

Setting up deployment to the cloud:

When we have a good build, we want to deploy our software. We'll use Heroku do deploy our app to a server on the public internet. Heroku is free for low usage and is easy to set up.

Create an account on Heroku at to heroku.com (you'll get a confirmation email etc).
Install the Heroku command line client - see https://devcenter.heroku.com/articles/heroku-cli

Create an app on Heroku (either through the web UI or the command line client)
Choose a unique name (e.g. horse-battery-66)
Add the generated Heroku remote to your local git repo.
Push to that remote to deploy.
Check that you can see your application running (e.g. https://horse-battery-66.herokuapp.com/api).

Setting up automatic deployment from Travis to Heroku:

The idea of *Continuous Deployment* is that every time we have a good build we want to deploy this into production automatically. This helps us to release frequently with small batches of changes.

Configure Travis to deploy to Heroku on successful builds (only).
You need to add a "deploy" section to .travis.yml
See https://docs.travis-ci.com/user/deployment/heroku/ for how to do that.

To encrypt your auth token, you can use the form on this website:
http://rkh.github.io/travis-encrypt/public/index.html (in the box for *repository,* put the name of your GitHub account and repo, something like rchatley/skeleton-java-app).

When you change the .travis.yml file, make a git commit push it back to GitHub. That should cause Travis to read the file and follow the new configuration.

Pushing changes through the pipeline:

Make a change to your application, commit, push and check that it is deployed.

Now make a change to the query processor to make the app respond to a new query. Write a unit test first to check the behaviour. When the tests pass, commit and deploy again. Try intentionally breaking the build to make sure that only green builds are deployed.