Dev(eloper)Ex(perience) Next
Design Document
Ian 'Uther' Lawson
2-Jun-2020

Precis

This document contains the detailed design for the next iteration of the DevEx Lab. DevEx has been successful but as it has grown it has become inflexible in the way it is deployed and the course size has become too big for a single session.

The design goals for this next iteration were to produce a much easier to use framework from the perspective of generating the documentation, and to open up the project structure such that anyone can contribute labs without impacting the overall integrity of the project.

As such the main points to address were:

- Modularising the Labs such that they are all effectively standalone (with the exception of a generic Terminal application available to all Labs
- Simplifying the build and publish mechanism for the course documentation as it is currently a bit of a black art to get everything where you need it before a generation and the output where you need it after the generation
- Making the structure of the labs fluid and easy to add to, hopefully to encourage submissions from experts outside the DSA UKI team to the pool of available labs

High Level Design

The execution of a workshop remains the same - a Cluster needs to be stood up with x users (of name userx, password openshift). All prerequisites for labs will be installed via a single Ansible playbook; to make the installation simple we will blanket install all Operators required for all labs regardless of whether the labs are included in the workshop.

Creation of a workshop will now entail the use of a 'DocBuilder' image and the execution of a 'DocBuilder' application.

The DocBuilder image contains a base OS, Maven, JAVA1.8+ and *possibly* Apache WebServer **

The DocBuilder application is a standalone JAVA application that executes within the DocBuilder image and works as follows:

- Application is driven by a configurable manifest file*** injected via a ConfigMap to the image, or provided externally if the application is executed away from the DocBuilder image
- Application clones the latest new DevEx repo****
- Application scaffolds the file system required for a Maven asciidoc build
- Application constructs a lab specific index asciidoc file based on the required labs provided via the manifest***
- Application executes the Maven asciidoc build and delivers the customised lab documentation to a target point specified in the manifest***
- Application removes scaffolded file system

Asterisk notes are detailed in Low Level Design that follows.

New Lab Categorisation

The original DevEx became too big so we split it into two 'courses', Basic and Advanced. This wasn't the best way to categorise so we shifted to Essentials and Innovation. With the new approach you generate a single doc per workshop (or many different ones) specifically tailored for the audience.

To address this approach we now will categorise *all* the labs and each lab has the category in its title, i.e. 'Introduction to 'oc' [ESSENTIALS]'. This categorisation is cosmetic so there is no restriction on the number. The initial suggestion for the categories is:

- [INTRODUCTION] - lab covering the basics and a low level of expertise required
- [ESSENTIALS] - lab covering mechanisms of OpenShift, medium level of expertise
- [DEVTOOLS] - lab specific to tools provided as part of OCP for devs (i.e. CodeReady, ODO)
- [INNOVATION] - lab specific to innovative enhancements provided by OCP (i.e. Knative, camel)

Low Level Design

** The intention is to provide a WebServer as part of the DocBuilder image - in that case when a Red Hatter sets up the workshop they download the DocBuilder image and it functions as the mechanism to auto-generate the lab documentation *and* serve it within the Cluster, removing the necessity to move generated documentation from the point of creation to the Cluster for user consumption.

**** The new DevEx repo is simplified and contains the following structure:

- labs/src - where *all* individual lab asciidocs are placed
- labs/images - where *all* images consumed by the labs are placed

- labs/material - where all the lab specific material is placed depending on the lab (i.e. camelfiles would be a separate directory here)
- labs/manifests - example manifests for different labs, i.e. full, innovations, essentials, devtools
- playbooks - where the lab setup playbook is hosted and run from
- apps - containing subdirectories for the various apps used by the labs (i.e. nodeatomic and the like)

Note that the new devex repo does *not* contain the esoteric directory structure needed for the Maven asciidoc build - this is scaffolded by the DocBuilder app

*** The asciidoc build is driven by a top level index file that specifies which of the labs are to be rendered in the output documentation. The new index for a lab is constructed from a template which contains the introduction and pre-req documents, which introduce the lab at a high level and then walk the users through creating their first project (terminalx) and installing the DevEx terminal image (quay.io/ilawson/devex4) which contains all the CLI for all labs). These two components are identical regardless of the final lab documentation. DocBuilder works by adding entries into the index.asciidoc for labs required in the order specified in the manifest. This manifest, provided to the DocBuilder via ConfigMap in the case of the use of the DocBuilder image or provided as a reachable file if the DocBuilder is run standalone, takes the following form:

```
Facilitator: Ian 'Uther' Lawson
Email: ian.lawson@redhat.com
Title: Principal Code Monkey
ClusterURL: https://mycluster/console
DocumentTitle: DevEx for Customer x
----
introduction_introtooc.asciidoc
essentials_deployment.asciidoc
essentials_sdn.asciidoc
essentials_rbac.asciidoc
devtools_codeready.asciidoc
innovation_knative.asciidoc
----
```

Note that the DocBuilder automatically adds the intro and prereq sections to all documents

DocBuilder sources the files directly from the (repo)/labs/src when scaffolding the build process.

Collaborations

Experts within Red Hat (or even partners and the like) will now be able to submit labs in the asciidoc format. These will be vetted, categorised and added to the src pool, and then available to anyone as part of a workshop via the manifest file.

At some point the manifest and submission may be changed to make the references more human-readable - at the moment creating a lookup table for label->asciidoc seems overkill but if we get to the realm of 100+ labs then another application will be written allowing people to select labs, order them and then auto-generate the manifest.