Red Hat Microservices

Red Hat Microservices

1. Admin

1.1. Objectives

The objectives of the Red Hat Microservices organization are multiple as we would like to centralize the information about the quickstarts, hands on lab, demo available to promote, educate the end users around the Microservice hype.

We would like also to define the vocabulary required to speak the Microservices language. Furthermore, we will cover and demystify the new patterns specific to a Microservice Architecture like the concepts inherent to that development model as; Greenfield, Brownfield, CI/CD …

1.2. Contributing

To contribute to this organisation, you can submit your Project's informations or the code in roder to create a new Git projet to host it. You can also keep & maintain your project separately. In both cases, here are the information required :

- Name of the project,
- Description,
- Technology involved : Language used, Domain (Middleware, mobile, …)
- Tags : categories, keywords describing the project
- Owner : Author or main contributor of the project
- Link to the git repository (github, bitbucket, …) - optional

1.3. Convention to follow to create a project

1.3.1. Technology driven

Every git hub project to be created will be named according to this convention :

- The name of the project will start with a 3-4 letters prefix describing what type of content it is related to (e.g lab = Lab, Hand on lab; quick = quickstart, …),
- Followed by the targeted container (e.g. swarm = WildFly Swarm; eap = JBoss EAP; fuse = JBoss Fuse; sb = SpringBoot) &
- Finally any keywords which will help to better define/describe what the project stands for (sec= security, sso = single sign on; tool = tooling).

Remark: The prefix, target & keywords will be separated by the underscore symbol.

So for the **Hands On Lab** project designed using **WildFly Swarm** as Java EE container and based on these technical concepts JBoss Forge as **tooling engine**, Keycloak as **SSO platform** to secure the application, we will name it as such **hol_swarm_tool-sso**.

1.3.2. User story based

A git hub project can also created by adopting a different convention where we favored the enterprise patterns, use case, domain, part of the project developed. In this case, we will adopt this convention :

- The name of the github repo will start with a 3-4 letters prefix describing what type of content it is related to (e.g lab = Lab, Hand on lab; quick = quickstart, demo = demo, …),
- Next, we will include the shorter name of the main technology domain covered (e.g javaee = Java EE, integr = integration, brms = business rules management system, ws = web service, rest = restfull services, api = api management, mob = mobile, msg = messaging, cloud = OpenShift Cloud Platform, …)
- To continue with the keywords and/or elements which can best describe the use case covered

By example the **Quickstart** project designed to demonstrate how **OpenShift** and **Kubernetes** supports the Microservices **Service Discovery pattern** will be named as such **quick_cloud_servicediscovery-dns**

2. Inventory of the material

2.1. Quickstarts

HelloWorld

- Hello World Microservices

Camel, SpringBoot, Karaf

- Fabric8 Quickstarts
- Fabric8 Archetypes
- FuseByExamples & Camel Microservices

2.2. Hands on Lab

Java One 2017

- Hol with Wildfly Swarm, Forge & Keycloak

2.3. Others

Book

- Source of the Christian Posta Book “Microservices”

Monolith to Micro

- Ticket Monster Application (Monolith to Microservices)
- Monolith Ticket Monster

- Simplified e-commerce application using a layered architecture (front (jsf) - logic (ejb) - data access (jpa))