Google Summer of Code 2020 ideas

General Information

The JBoss Community is planning to participate in Google Summer of Code in 2020.

All students & developers are welcome to participate in the https://summerofcode.withgoogle.com/ program with the JBoss Community (once JBoss Community is accepted by Google)!

You can take look on org page of Summer of Code website for proceeding with the application process.

If you are a student looking forward to participate in the GSoC 2020 with the JBoss Community, feel free to browse the growing idea list below. Please don't hesitate to contact the mentor(s) indicated in the proposal for any related clarification and to discuss proposals. Students can submit the proposals in 2020.

If you want to suggest an additional idea, please use the page GSoC 19 Student Ideas (you'll need a free JBoss Community account to edit the page). Interested mentors can check the student ideas page and sign up to be a mentor by moving the idea onto the main page.

You can also look at GSoC-16 Ideas, 2017 Ideas, 2018 Ideas and 2019 Ideas for suggestions from last years.

A note to mentors

MENTORS: Red Hat employees can change this page directly to add ideas. Please be extra careful to not get other mentor's edits discarded. Red Hatters should have linked their jboss.org account with Red Hat and can be checked on https://sso.jboss.org/login

Non Red Hatters can add a comment to the page and admins will make sure the idea is added to the page.

Table of Contents

Administrators and Mentors

We will list the potential mentors in this place. For now, if you have any questions, please contact the GSoC administrators:

George Zaronikas (gzaronikas) Ali Ok (@aliok_tr) and Anuj Garg (@KeenWarrior).

Communication channels

Gitter : **JBossOutreach/GSoC - Gitter**

Please take note - These channels are about generic doubts. For project specific doubts you will need to contact project mentors and channels specified in the project description.

Notes for students

Points to consider while choosing any project

1. You meet at least 50 percent of prerequirements. Remaining skills can be honed on the go, so don't worry if you lack some.

2. You can relate with the project idea and you have used something related to the project as user.

3. You are willing more toward learning the skills and less toward boasting about the skills you have already.

Suggested steps after choosing favourite project

1. Start to use product/tool/api as user or hello world client application.

2. Setup the development environment for project and start to use your own build.

3. Look for new comer bugs and try to get your head around them.

4. Let us know if you feel stuck at any stage.

Idea template (for mentors)

Project title

Summary of idea:

-Idea

-Feature A

-Feature B

Knowledge prerequisite: Languages/Technologies goes here

Github repo:

Skill level: Beginner/Intermediate/Advanced

Contact(s) / potential mentors(s): Mentor(s) name and contact details

Associated JBoss community project(s):

Idea Proposals

AeroGear - NodeJS based Data Synchronization engine on top of Apache Kafka, Debezium and GraphQL Subscriptions

Summary of the idea:

GraphQL Subscriptions allow developers to build reactive data-driven applications where each client actively subscribes to new data and receive it immediately after is available.
However in traditional messaging systems when a client subscribes for the first time it is not

going to get any previous messages. Purpose of this will be to provide offline enabled data synchronization platform on top of GraphQL.

Project references

Candidates are going to work closely with the JBoss community by contributing to the following projects:

- Graphback: https://github.com/aerogear/graphback
- Debezium: https://github.com/debezium/debezium
- Offix: https://github.com/aerogear/offix

Note: This project can incorporate more than one topic/student as it can be focused on client and server-side exclusively.
Candidates should pick client or server-side implementations when doing a proposal.

Required knowledge

- Node.js (Understanding performance implications of event loop etc.),
- TypeScript
- Basics of GraphQL, React
- Apache Kafka (Basic concepts, Consuming messages, Partitioning)
- Knowledge of one of the relational and NoSQL databases (Postgress, MongoDB)
- Familiarity with the event streaming principles
- Engagement with AeroGear and Debezium communities is essential!

Definition of the problem

When building microservices architecture various parts of the system can interact with the source database. Using publishing events from the server application is not always possible and it might require modifications in various servers that are often legacy.

When utilizing Debezium and Kafka we can connect directly to the database to get notified about any change that happened in the database instantly.

Debezium will be only involved in collecting and publishing the data to Kafka. The purpose of this work is to build an opinionated Node.js layer that will consume Kafka messages to apply some specific filtering and publish messages back to clients using GraphQL subscriptions.

Possible goals:

- Introduce a stateful live query layer for Graphback based on Kafka stream (https://www.youtube.com/watch?v=g-asVW9JFPw)
- Check the performance of the solution and identify bottlenecks for scalability. (https://github.com/hasura/graphql-engine/blob/master/architecture/live-queries.md#implementing-graphql-live-queries)
- Build a suite of the integration tests for various Graphback use cases
- Provide example app for Debezium
- Build a Node.js OpenShift template application for Graphback that can connect with Kafka
- Build out of the box subscription connector for Offix that will update underlying cache/db

Diagram: https://media.discordapp.net/attachments/632235423300714507/669530559701319691/Screenshot_2020-01-22_at_13.15.08.png

Technologies: Nodejs, GraphQL, AeroGear DataSync, React, Apache Kafka

Getting Started: https://github.com/aerogear/GSoC-2020

Skill level: Intermediate

Contacts: wtrocki@redhat.com dzuccare@redhat.com ephelan@redhat.com

Associated JBoss community project(s): AeroGear

AeroGear - A GraphQL Engine on top of Knative and Node.js

Summary of idea:

The purpose of this project is to be able to port functional Node.js based GraphQL applications into Knative.
The biggest challenge for that is an implementation of GraphQL Subscriptions.

GraphQL subscriptions work with websockets or some other persistent connection. When a mutation happens in the GraphQL engine, a subscribed client receives the event of that mutation over that persistent connection.
In the current case, the piece of software that sends the subscription events runs all the time.
How about scaling to zero if there are no subscribed clients?

This project will need some level of research in these areas, before the implementation:

- How to use Websockets with Knative
- How to scale resources with regards to Websocket connections

- How to design application to work efficiently with GraphQL

Possible tasks for this project :

- A prototype of simple GraphQL Node.js server on top of Knative
- Research Graphback Runtime to be used on top of Knative

Required knowledge

- Kubernetes
- Node.js/TypeScript
- Knative
- Containers and tooling (e.g. Docker)
- GraphQL
- Websockets

NOTE: all the skills listed above are big things. We don't expect students to be experts in any of these. However, we expect students to have experimented with these technologies and have experience in majority of them.

Github repo: https://github.com/aerogear/graphback (separate Knative repository might be created)

Getting Started: https://github.com/aerogear/GSoC-2020

Contact / potential mentors: Wojciech Trocki (wtrocki@redhat.com) Ali Ok (aliok@redhat.com)

Associated JBoss community project: Aerogear

EAT - Testing Infinite Software Project Versions

Summary of idea:

The innovative part of EAT is creating the test once and testing with any version of the tested software. It may be firstly applied for the JBoss Servers, but, in general, a similar structure, can be used for creating tests about any software with multiple versions or for multiple software programs that have a part of the testsuite in common. EAT is a project under the statement.

Possible tasks for this project :

Go through the EAT workshop, extending the existing AT testsuites, creating a new testsuite using the AT Structures, improving the dependency AT analyzer (dynamic testing), etc

Github repo: https://github.com/EAT-JBCOMMUNITY/EAT

Contact / potential mentors: Panagiotis Sotiropoulos (psotirop@redhat.com)

Associated JBoss community project: EAT

Knative - Event sources for container registries, pipelines and builds

Summary of idea:

Implement multiple Knative eventing sources container registries, pipelines and builds.

- An event source that is able to generate cloud events when a new image is uploaded to a registry
- Another event source that is able to generate cloud events on build success/failure as well as on pipeline success/failure and on individual steps of a pipeline.

Image registry mentioned is DockerHub.

Pipeline and builds are done by Tekton.

Possible tasks for this project :

- Implement the sources
- Implement unit tests, e2e test
- Implement performance tests
- Write documentation
- Prepare demos

Required knowledge

- Kubernetes
- Knative

- Containers and tooling (e.g. Docker)
- Golang

NOTE: all the skills listed above are big things. We don't expect students to be experts in any of these. However, we expect students to have experimented with these technologies and have experience in majority of them.

Github repo: https://github.com/knative/eventing and https://github.com/knative/eventing-contrib

Contact / potential mentors: Ali Ok (aliok@redhat.com)

Associated JBoss community project:

- Knative (upstream project) mailing list: https://groups.google.com/forum/#!forum/knative-users
- Tekton (upstream project); community information: https://github.com/tektoncd/community

Knative - Alternative Knative Broker Implementation based on Apache Kafka

Summary of idea:

Based on the Knative proposal: Alternative Broker Proposal (you need to join knative-dev user group to see the document)

Current Broker implementation in Knative Eventing is implemented using channels. We would like to have an alternative broker, without using channels, that is using Apache Kafka instead.

The goal is to implement a Kafka native broker since the default channel based broker has a few unnecessary network hops between HTTP and Kafka protocols.

Possible tasks for this project :

- Design and implementation of a Kafka native broker implementation
- Conformance tests for broker to match expected API
- Trigger/filter query for broker specific event routing and demo'ing
- Performance tests to compare the design against channel based broker
- Write documentation

Required knowledge

- Kubernetes
- Knative
- Apache Kafka
- Golang (must)

NOTE: all the skills listed above are big things. We don't expect students to be experts in any of these. However, we expect students to have experimented with these technologies and have experience in majority of them.

Github repo: https://github.com/knative/eventing and https://github.com/knative/eventing-contrib

Contact / potential mentors: Ali Ok (aliok@redhat.com)

Associated JBoss community project:

- Knative (upstream project) mailing list: https://groups.google.com/forum/#!forum/knative-users

Project Starfish - Open anything anywhere in any IDE/editor

Summary of idea:

Implement a client side app that supports urlhandlers (i.e. ide://clone-url?url=https://github... , ide://open-file?, ide://open-debugger?port=..&project=url, etc.)

Then integrate these various actions to perform and setup in vscode, eclipse, intellij, emacs, vi, etc.

Make it work across Linux, OSX and Windows so it can be used from anywhere.

Possible make browser extensions to enable it on various websites like github, gitlab, etc.

This project will need some level of research:

- How to setup url handlers on all three major platforms
- Understand the basic functionallity of at least three IDEs/Editors open/clone features to show it will work
- Explore extendability of browers or IDE's as needed

Possible tasks for this project:

- Implement cross-platform app

- Add tests
- Write documentation
- Prepare demos

Knowledge prerequisite:

- Open choice on language, but most likely Java, or Go based
- Basic understanding of IDE and/or Browser extensions
- Access to more than one of the Operating Systems, virtual machines okey.

Github repo:

Since this is a new idea there are no direct current github repo for it - for now created https://github.com/maxandersen/starfish for having a place of discussion/conversion.

If you want to explore/work in this area and show your

experience/interest you can look into contributing to projects like https://github.com/maxandersen/jbang or any vscode-* repo under https://github.com/redhat-developer/ or for more advanced

https://github.com/quarkusio. Basically any project that relates to developer tools, desktop tools, scripts or browser extensions will be useful experience.

Slides:

An outline of the idea with some screenshots/overview can be found here - feel free to comment.

Skill level: Beginner/Intermediate

Contact(s) / potential mentors(s): Max Rydahl Andersen (manderse@redhat.com)

Associated JBoss community project(s): Quarkus, vscode extensions, JBoss Tools

Improve Kotlin support in Quarkus

Summary of idea:

Help improve the existing Kotlin support in Quarkus. Although Kotlin is a supported language in Quarkus there are a few rough edges that need improvement, especially with regards to the GraalVM native image support.

This project will require some level of research:

- Understanding the bytecode the Kotlin compiler produces
- Understanding the limitations of GraalVM's native images and what techniques Quarkus uses to overcome them

Possible tasks for this project:

- Improve the interaction of Kotlin Data Classes with Json libraries (like Jackson) when working with GraalVM
- Explore the use Kotlin Data Classes as Hibernate and / or Mongo Panache entities

Knowledge prerequisite:

- Java and Kotlin programming language
- Basic understanding of Java build tools

Github repo: https://github.com/quarkusio/quarkus

Skill level: Intermediate

Contact(s) / potential mentors(s): Georgios Andrianakis (gandrian@redhat.com)

Associated JBoss community project(s): Quarkus

Quarkus - Improve Gradle support

Summary of the Idea:

Quarkus is a Kubernetes Native Java framework tailored for GraalVM and HotSpot, crafted from best-of-breed Java libraries and standards.

Currently in Quarkus, support of the maven build system is mature. Gradle is another build system. Gradle is an open-source build-automation system that introduces a Groovy and Kotlin-based domain-specific language (DSL) and a more declarative description of dependencies allowing for more repeatable and faster incremental builds

Possible tasks:

- Support multi-module projects (https://github.com/quarkusio/quarkus/issues/5722)
- Support for Gradle Tests (https://github.com/quarkusio/quarkus/issues/5101)

Required Knowledge

- Java
- Gradle build system

Skill Level: Intermediate

NOTE: all the skills listed above are big things. We don't expect students to be experts in any of these. However, we expect students to have experimented with these technologies and have experience in the majority of them.

Github Repo : https://github.com/quarkusio/quarkus

Associated JBoss community project(s): Quarkus

Contact/Potential Mentors : Max Rydahl Andersen (manderse@redhat.com), Georgios Andrianakis (gandrian@redhat.com), Rohan Maity(rmaity@redhat.com)

Extend Apache Tomcat Kubernetes operator

Summary of idea:

Extend the current tomcat operator . Functionality and planning should be discussed between mentors and students

Knowledge prerequisite: Go, Kubernetes, Java, Tomcat

Github repo: https://github.com/web-servers/tomcat-operator

Skill level: Intermediate

Contact(s) / potential mentors(s): Petros Prokopiou (pprokopi@redhat.com) George Zaronikas (gzaronik@redhat.com)

Associated JBoss community project(s): JWS / Apache Tomcat

HTTP/3 Tomcat connector

Summary of idea:

Create and demo HTTP/3 connector for Tomcat using java http/3 QUiC libraries. Look to https://github.com/quicwg/base-drafts/wiki/Implementations for existing implementations. The idea is to evaluate the existing libraries and find one that can be use to create an HTTP/3 connector. If no libraries are OK for the project we might have to write the HTTP/3 code we need for the prototype.

We don't plan to use a native library like Quiche for the moment but that might be a solution to get prove of concept quickly. Note the IETF has published a new draft (https://quicwg.org/base-drafts/draft-ietf-quic-http.html) of the protocol.

Knowledge prerequisite: Tomcat, Java, HTTP protocols (HTTP)

Github repo: N/A for the moment, should be created from scratch

Skill level: Intermediate

Contact(s) / potential mentors(s): Jean-Frederic Clere (jfclere@redhat.com), George Zaronikas (gzaronik@redhat.com)

Associated JBoss community project(s): JWS / Apache Tomcat

Add Kubernetes dynamic proxy configuration to Apache httpd

Summary of idea:

The purpose of this project is for a student to create a custom Kubernetes dynamic proxy configuration for Apache httpd. mod_proxy_balancer allows some dynamic creation of work, the idea is to use the kubernetes APU to discover the nodes where an application is running and the port on which the service is exposed in kubernetes.

kubectl get nodes gives the list of nodes (all nodes)

kubectl get services display the service port (and the VirtualHost).

To create a worker for mod_proxy_balancer we need the node (name or IP) and the port, for the VirtualHost that is linked to webapp name.

The idea is to create the VirtualHost and workers dynamically using the information from the kubernetes API.

Knowledge prerequisite: Kubernetes, httpd, mod_proxy.

Github repo: Should be added here https://github.com/web-servers

Skill level: Intermediate

Contact(s) / potential mentors(s): Jean-Frederic Clere (jfclere@redhat.com)

Associated JBoss community project(s): JBCS, Apache httpd

k3s vs k8s

Summary of idea:

The purpose of this project is for the students is to research, evaluate, compare k3s & k8s by building their own clusters and compare middleware and jboss offerings on these. Student should conduct experiments and investigations that will conclude in community blogposts. This project idea is open for discussion between students and potential mentors and prepare scenarios for a possible demos.

Knowledge prerequisite: Kubernetes

Skill level: Beginner

Contact(s) / potential mentors(s): Jean-Frederic Clere (jfclere@redhat.com)

Associated JBoss community project(s): Apache Tomcat, Wildfly, Apache httpd and more upon discussion.

Teiid Translators for AVRO, Paraquet

Summary of idea:

Teiid (https://teiid.io) is Data Virtualization system that allows reading data from various different data sources and data types. The idea of this project is to extend the support for Apache AVRO, Apache Paraquet files on HDFS or in any other store. The Serializers and deserializers for these are already defined https://docs.aws.amazon.com/athena/latest/ug/avro.html and https://docs.aws.amazon.com/athena/latest/ug/parquet.html

Teiid has a concept called a Translator, one need to write a new translator(s) for above where they need to use these SerDes from these links and convert the data.

Knowledge prerequisite: Java, Spring Boot

Github repo: https://github.com/teiid

Skill level: Intermediate

Contact(s) / potential mentors(s): Ramesh Reddy (rareddy@redhat.com), Steven Hawkins (shawkins@redhat.com)

Associated JBoss community project(s): Teiid

Intrusion Detection Using Elytron Security Events

Summary of idea:

Without intrusion detection, an attacker can attempt attacks many times until an attack is finally successful. Intrusion detection allows for these attacks to be identified before a successful attack is likely to occur. The purpose of this project is to add the ability to detect attacks using Elytron, the security framework used by the WildFly Application Server. In particular, Elytron already provides support for security events which can indicate things like a failed authentication attempt for a particular user. Can we leverage these events to detect things like multiple failed authentication attempts for a particular user? Once detected, what kind of action can we take (e.g., a server administrator could be notified, an account could be disabled, etc.)?

This project will need some level of research into the following areas before starting on the implementation:

- What kinds of attacks can be detected using Elytron security events?
- Should more security events be added to Elytron? Should more information be added to existing Elytron security events?
- What kinds of actions can be taken upon intrusion detection?

Possible tasks for this project:

- Identify a specific type of attack to focus on initially.

- Create a document that describes the attack you will be focusing on, how you plan to use security events to detect this attack, and what kind of action you plan to take upon detecting this attack.
- Implement the ability to detect this type of attack using security events.
- Implement the ability to take action upon detecting this type of attack.
- Implement appropriate test cases.
- Write documentation.
- Create a blog post that gives an overview of your project.
- Look into other attacks that could be detected.

Required knowledge:

- Experience with Java and a good understanding of object-oriented programming concepts
- Experience in data sciences, machine learning, or AI in general would be ideal

GitHub repo: https://github.com/wildfly-security/wildfly-elytron

Elytron website: https://wildfly-security.github.io/wildfly-elytron/

Elytron getting started guide: https://wildfly-security.github.io/wildfly-elytron/getting-started-for-developers

Skill level: Intermediate

Contact(s) / potential mentor(s): Darran Lofthouse (darran.lofthouse@redhat.com) and Farah Juma (fjuma@redhat.com)

Associated JBoss community project(s): Elytron, WildFly

Kubernetes Client - Improve tests

Summary of the Idea:

Kubernetes Client is a Java client for Kubernetes and OpenShift. The client provides access to the full Kubernetes & OpenShift REST APIs via a fluent DSL.

The current level of code coverage and tests for the client is, in general, quite good. However, in the recent months, we've added several extensions and updated some of the API versions, and the project requires an update of the current tests and the addition of newer ones to cover the missing points.

Possible tasks for this project:

- Provide more unit tests for the new auto-generated fluent DSL classes from new extensions and updated APIs.
- Improve current integration tests and our Mock Web Server implementation.
- Implement E2E tests using a real cluster (Minikube) > We have examples for this in some of the other projects our team manages.

Required knowledge:

- Java
- Some experience with Kubernetes or OpenShift

Skill Level: Intermediate

Github Repo: https://github.com/fabric8io/kubernetes-client

Contact(s) / potential mentor(s): Devang Gaur (dgaur@redhat.com), Rohan Kumar (rokumar@redhat.com) and Marc Nuri (marc.nuri@redhat.com)

Associated JBoss community project(s): fabric8io/kubernetes-client