NVS Script

Version: 1.0

Alexander Walliser

RestServer

1. RestConfig

```
at.htl.projectName.rest.RestConfig
```

```
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("api")
public class RestConfig extends Application {

}
```

2. Endpoint

```
at.htl.projectName.rest.Endpoint
```

```
import javax.ws.rs.GET;
import javax.ws.rs.POST;

import javax.ws.rs.Path;
import javax.ws.rs.core.Response;

@Path("projectName")
public class Endpoint {

    public Endpoint() {
    }

    @GET
    public Response Get(){}

    @POST
    public Response Post(){}
}
```

3. InitBean

```
at.htl.projectName.business.InitBean
```

```
import javax.enterprise.context.ApplicationScoped;
import javax.enterprise.context.Initialized;

import javax.enterprise.event.Observes;
import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;
import javax.transaction.Transactional;

@ApplicationScoped
public class InitBean {

    @PersistenceContext
    EntityManager em;

    public InitBean() {
    }

    @Transactional
    private void Init(@Observes @Initialized(ApplicationScoped.class) Object init)
    {
    }
}
```

4. Read File

```
private <T> void readCsv(String fileName, Function<String[], T> creatObject){
        URL url = Thread.currentThread().getContextClassLoader()
                .getResource(fileName);
        try (Stream<String> stream = Files.lines(Paths.get(url.getPath()),
StandardCharsets.UTF_8)) {
            stream.skip(1).map(line -> line.split(";"))
                    .map(creatObject)
                    .forEach(e -> em.merge(e));
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    private Function<String[], myObject> creatMyObject = d -> new MyObject(d[0]);
```

5. Sample RestServer Pom

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>at.thl</groupId>
    <artifactId>["projectName"]</artifactId>
    <version>1.0-SNAPSHOT</version>
    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
        <jakarta.jakartaee-api.version>8.0.0</jakarta.jakartaee-api.version>
    </properties>
    <packaging>war</packaging>

    <dependencies>
        <dependency>
            <groupId>jakarta.platform</groupId>
            <artifactId>jakarta.jakartaee-api</artifactId>
            <version>${jakarta.jakartaee-api.version}</version>
            <scope>provided</scope>
        </dependency>
    </dependencies>
    <build>
        <finalName>["projectName"]</finalName>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <configuration>
                    <source>8</source>
                    <target>8</target>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

6. project.iml

```
<?xml version="1.0" encoding="UTF-8"?>
<module type="JAVA_MODULE" version="4">
  <component name="FacetManager">
    <facet type="jpa" name="JPA">
      <configuration>
        <setting name="validation-enabled" value="true" />
        <datasource-mapping>
          <factory-entry name="myPU" />
        </datasource-mapping>
        <naming-strategy-map />
        <deploymentDescriptor name="persistence.xml"
url="file://$MODULE_DIR$/src/main/resources/META-INF/persistence.xml" />
      </configuration>
    </facet>
  </component>
</module>
```

7. persistence.xml

   in resources → META-INF

```
 persistence.xml
```

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence" version="2.1">
    <persistence-unit name="myPU" transaction-type="JTA">
        <jta-data-source>java:jboss/datasources/dbDS</jta-data-source>
        <properties>
            <!-- When 'could not initialize proxy - no Session' Error appears -->
            <property name="hibernate.enable_lazy_load_no_trans" value="true"/>
            <!--
            <property name="hibernate.show_sql" value="true"/>
            <property name="hibernate.format_sql" value="true"/>
            <property name="hibernate.transaction.flush_before_completion"
value="true"/>
            <property name="hibernate.dialect"
value="org.hibernate.dialect.DerbyDialect"/>
            -->
            <property name="javax.persistence.schema-generation.database.action"
value="drop-and-create"/>
            <!--
            <property name="javax.persistence.schema-generation.scripts.action"
value="drop-and-create"/>
            <property name="javax.persistence.schema-generation.scripts.create-
target" value="sampleCreate.ddl"/>
            <property name="javax.persistence.schema-generation.scripts.drop-
target" value="sampleDrop.ddl"/>
            <property name="javax.persistence.sql-load-script-source"
value="insert.sql"/>
            -->
        </properties>
    </persistence-unit>
</persistence>
```

8. Db

```
/opt/db-derby-10.14.2.0-bin/bin/startNetworkServer -noSecurityManager
```

9. Sample Get

```
    @Path("entity/{id}")
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Entity findEntityById(@PathParam("id") Long id){
        return em.createNamedQuery("Entity.findById", Entity.class).setParameter(
"ID",id).getSingleResult();
    }

    @Path("entity")
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Response findEntityById(@QueryParam("id") Long id){
        return Response
                .status(Response.Status.OK)
                .header("SomeText","Hallo")
                .entity(em.createNamedQuery("Entity.findById", Entity.class)
                        .setParameter("ID",id).getSingleResult())
                .build();
    }
```

10. Sample Post

```
    @Path("addentity")
    @POST
    @Consumes(MediaType.APPLICATION_JSON)
    @Transactional
    public Response createEntity(final @Context UriInfo uriInfo, JsonValue
jsonValue){
        JsonArray jsonArray = (jsonValue.getValueType() == JsonValue.ValueType
.ARRAY) ? jsonValue.asJsonArray()
                : Json.createArrayBuilder().add(jsonValue).build();
        for (JsonValue value : jsonArray) {
            System.out.println(value.toString());
            em.merge(
                    /*With Gson: new
Gson().fromJson(jsonValue.toString(),Entity.class)*/
                    createEntityFromJson(jsonValue)
                    );
        }
        //return Response.ok().build();
        URI uri = uriInfo.getAbsolutePathBuilder().path("/ entity").build();
        return Response
                .created(uri)
                .header("operation", "object created")
                .build();
    }

    private Entity createEntityFromJson(JsonValue value){
        try {
            return new Entity(value.asJsonObject().getString("name"),
                    createSomeOtherEntity(value.asJsonObject().getJsonObject(
"otherEntity")),
                    value.asJsonObject().getJsonArray("entityCollection").stream()
                            .map(o -> createCollectionEntity(o.asJsonObject()))
.collect(Collectors.toList())
                    );
        }catch (Exception e){
            return null;
        }
    }
```

11. Sample Entity

```
    @NamedQueries(
        value = {
                @NamedQuery(
                        name = "Entity.findById",
                        query = "select e from Entity a where e.id = :ID"
                )
        }
)

@Entity
public class Entity implements Serializable {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private  Long id;
    private String name;

    public Entity() {
    }
}
```

12. Relations
    a. 1:n uni

```
    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name="entity_id")
    private List<Entity> entities;
```

    b. 1:n uni

```
    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name="entity_id")
    private List<Entity> entities;

    ...

    //With with association table
    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name="entity_id")
    private List<Entity> entities;

    ...

    @ManyToOne
    private Entity entity;
```

    c. 1:n bi

```
//Entity
@ManyToOne
private OtherEntity otherEntity;

//OtherEntity
@OneToMany(mappedBy="otherEntity" ,cascade = CascadeType.ALL)
@JoinColumn(name="entity_id")
private List<Entity> entities;

...

//With with association table
//Entity
@ManyToOne
private OtherEntity otherEntity;

//OtherEntity
@OneToMany(cascade = CascadeType.ALL)
private List<Entity> entities;
```

d. n:m

```
//Entity
@OneToMany(mappedBy = "id.entity", cascade = CascadeType.ALL)
private List<EntityItem> entityItmes;

//EntityItem
@Embeddable
class EntityItem implements Serializable {
    @ManyToOne
    @JoinColumn(name = "entity_id")
    private Entity entity;

    @ManyToOne
    @JoinColumn(name = "otherEntity_id")
    private OtherEntity otherEntity;
    }
```

e. Inheritance

```
    @Inheritance(strategy=InheritanceType.SINGLE_TABLE) //All data in a single
Table

    @Inheritance(strategy=InheritanceType.TABLE_PER_CLASS) //Table for every
Subclass

    @Inheritance(strategy=InheritanceType.JOINED) // Same Fields grouped in a
table
```

13. Select in Tuples

```
TypedQuery<Tuple> q = em.createQuery("select e.id, o.id  from Entity e join
e.otherEntity o ",Tuple.class);

        List<Tuple> result = q.getResultList();

        result.forEach(t -> System.out.printf("%d, %d", t.get(0,Long.class),t.get(
1,Long.class));
```

Rest Client

1. Test

```
at.htl.projectName.test.projectNameIT;
```

```
    import static org.hamcrest.CoreMatchers.is;
    import static org.hamcrest.MatcherAssert.assertThat;

    public class DemoEndpointIT {
        private Client client;
        private WebTarget target;

        @BeforeEach
        public void initClient(){
            client = ClientBuilder.newClient();
            target = client.target("http://localhost:8080/projcet/api/project/");
        }

        @Test
        public void test(){
            Response response = this.target.request(MediaType.APPLICATION_JSON).
get();
            assertThat(response.getStatus(),is(200));
            String st = response.readEntity(String.class);
            System.out.println(st);
             //Gson: Entity entity = new Gson().fromJson(st, Entity.class);
            JsonValue jsonValue = Json.createReader(new StringReader(st))
.readObject();
            JsonArray jsonArray = (jsonValue.getValueType() == JsonValue.ValueType
.ARRAY) ? jsonValue.asJsonArray()
                    : Json.createArrayBuilder().add(jsonValue).build();
            List<Entity> entity = jsonArray.stream().map(j -> createEntity(j))
.collect(Collectors.toList());

            client.target("http://localhost:8080/projcet/api/project/").request
(MediaType.APPLICATION_JSON)
            .post(Entity.json(new Gson().toJson(entity)));
        }
    }
```

2. Gson

```
    <dependency>
        <groupId>com.google.code.gson</groupId>
        <artifactId>gson</artifactId>
        <version>2.8.5</version>
    </dependency>
```

3. pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>at.htl</groupId>
    <artifactId>demotest</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <junit.jupiter.vesion>5.5.2</junit.jupiter.vesion>
        <org.hacrest.version>1.3</org.hacrest.version>
        <jakarta.jakartaee-api.version>8.0.0</jakarta.jakartaee-api.version>
        <org.glassfish.version>1.1.4</org.glassfish.version>
        <org.jboss.resteasy.version>4.3.0.Final</org.jboss.resteasy.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter</artifactId>
            <version>${junit.jupiter.vesion}</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.hamcrest</groupId>
            <artifactId>hamcrest-all</artifactId>
            <version>${org.hacrest.version}</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>jakarta.platform</groupId>
            <artifactId>jakarta.jakartaee-api</artifactId>
            <version>${jakarta.jakartaee-api.version}</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.jboss.resteasy</groupId>
            <artifactId>resteasy-client</artifactId>
            <version>${org.jboss.resteasy.version}</version>
        </dependency>
        <dependency>
            <groupId>org.glassfish</groupId>
            <artifactId>javax.json</artifactId>
            <version>${org.glassfish.version}</version>
        </dependency>

    </dependencies>
```

```
</project>
```