# Verbesserung des 1. SEW-Tests

## Aufgabe 1: Import CSV

### readTeamsAndDriversFromFile

Ich habe die Daten direkt nach dem Splitten zu speichern, anstatt dass ich das String-Array an die persistTeamAndDrivers Methode weitergebe und dort die Daten erst speicher. Außerdem habe ich bei Driver versucht mit `map(a → new Driver(a[1],new Team(a[0])))` die Daten zu speichern. Das funktioniert aber nicht, weil ich statt `new Team(a[0])` eine Array-Position statt einem Team hätte angeben müssen.

```
    private void readTeamsAndDriversFromFile(String teamFileName) throws IOException {
        URL url = Thread.currentThread().getContextClassLoader()
                .getResource(teamFileName);
        try (Stream<String> stream = Files.lines(Paths.get(url.getPath()))) {
            stream
                    .skip(1)
                    .map(lines -> lines.split(";"))
                    .map(a -> new Team(a[0]))
                    .forEach(em::merge);

            //            stream
//                    .skip(1)
//                    .map(lines -> lines.split(";"))
//                    .map(a -> new Driver(a[1],new Team(a[0])))
//                    .forEach(em::merge);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

#### Verbesserung:

```
    private void readTeamsAndDriversFromFile(String teamFileName) throws IOException {
        URL url = Thread.currentThread().getContextClassLoader()
                .getResource(teamFileName);
        try (Stream<String> stream = Files.lines(Paths.get(url.getPath()),
StandardCharsets.UTF_8)) {
            stream
                    .skip(1)
                    .map(lines -> lines.split(";"))
                    .forEach(this::persistTeamAndDrivers);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

Hier wird zuerst mit der `@NamedQuery( name = "Team.findByName", query = "select t from Team t where t.name = :NAME" )` geschaut, ob es das Team schon gibt, wenn nicht wird das Team gespeichert.

```
    private void persistTeamAndDrivers(String[] line) {
        Team team = null;
        try {
            team = em.createNamedQuery("Team.findByName", Team.class)
                    .setParameter("NAME", line[0])
                    .getSingleResult();
        } catch (NoResultException ex) {
            team = new Team(line[0]);
            em.persist(team);
        }
        em.persist(new Driver(line[1], team));
        em.persist(new Driver(line[2], team));
    }
```

# Aufgabe 2: Import REST

## readResultsFromEndpoint

```
    public void readResultsFromEndpoint() {
        this.client = ClientBuilder.newClient();
        this.target = client.target(RESULTS_ENDPOINT);

        Response response = this.target.request(MediaType.TEXT_PLAIN).get();
        JsonArray payload = response.readEntity(JsonArray.class);

        persistResult(payload);
    }
```

### Verbesserung:

MediaType vom Response muss natürlich ein JSON sein, da payload ein JsonArray ist.

```
        Response response = this.target.request(MediaType.APPLICATION_JSON).get();
```

Das Speichern der Daten hätte so ausschauen können:

```
    @Transactional
    void persistResult(JsonArray resultsJson) {
        for (JsonValue jsonValue : resultsJson) {
            JsonObject resultJson = jsonValue.asJsonObject();
            Driver driver = em
                    .createNamedQuery("Driver.findDriverByName", Driver.class)
                    .setParameter("NAME", resultJson.getString("driverFullName"))
                    .getSingleResult();

            Race race = em
                    .createQuery("select r from Race r where r.id = :ID", Race.class)
                    .setParameter("ID", Long.valueOf(resultJson.getInt("raceNo")))
                    .getSingleResult();

            em.persist(new Result(
                    race,
                    resultJson.getInt("position"),
                    driver
            ));
        }
    }
```

# Aufgabe 3: Gesamtpunkte eines Fahrers

Die NamedQuery kann nicht funktionieren, weil alle Spalten summiert werden. Ich hätte sum(r.points) verwenden sollen, damit die Punkte des Fahrers summiert werden. Außerdem habe ich den Fehler gemacht, dass nur die Punkte gezählt werden, wenn der Fahrer der 1. Platz beim Rennen war.

```
@NamedQuery(
                name="Result.findDriverPointsById",
                query="select sum(r) from Result r where r.driver = :DRIVER " +
                        "and r.position=1"
        )
```

Außerdem hätte ich bei Result einfach mit dem Fahrernamen nach den Punkten suchen können, anstatt mit der FahrerId. Bei der Query "Result.findDriverPointsById" bekomme ich die Punkte des Fahrers zurück, deshalb Hätte ich statt Result ein Long gebraucht. Beim Return hätte ich ein neues

Json mit dem Fahrername und seinen Punkten bauen und zurückgeben können.

```
    @GET
    @Path("points")
    @Produces(MediaType.APPLICATION_JSON)
    public JsonObject getPointsSumOfDriver(
            @QueryParam("name") String name
    ) {
        TypedQuery<Driver> query1 = em
                .createNamedQuery("Driver.findDriverByName", Driver.class)
                .setParameter("NAME", name);
        Driver d = query1.getSingleResult();
        Long driverId = d.getId();
        TypedQuery<Result> query2 = em
                .createNamedQuery("Result.findDriverPointsById", Result.class)
                .setParameter("DRIVER", driverId);
        Result r = query2.getSingleResult();

        return null;
    }
```

## Verbesserung:

```
        @NamedQuery(
                name="Result.sumPointsForDriver",
                query="select sum(r.points) from Result r where r.driver.name = :NAME
"
        )
```

```
    @GET
    @Path("points")
    @Produces(MediaType.APPLICATION_JSON)
    public JsonObject getPointsSumOfDriver(
            @QueryParam("name") String name
    ) {
        TypedQuery<Driver> query1 = em
                .createNamedQuery("Driver.findDriverByName", Driver.class)
                .setParameter("NAME", name);
        Driver d = query1.getSingleResult();
        String driverName = d.getName();
        Long points = em
                .createNamedQuery("Result.sumPointsForDriver", Long.class)
                .setParameter("NAME", name)
                .getSingleResult();

        return Json
                .createObjectBuilder()
                .add("driver", driverName)
                .add("points", points)
                .build();
    }
```