Eva Hildenbrand 4AHITM

Testverbesserung

Aufgabe 1: Import CSV

- InitBean.java:
  - Alter Code:
    - Die folgende Codezeile wurde sowohl in der readTeamsAndDriversFromFile(TEAM_FILE_NAME)-Methode, als auch in der readRacesFromFile(RACES_FILE_NAME)-Methode abgeändert

```
    BufferedReader br = new BufferedReader(new
InputStreamReader(getClass().getResourceAsStream("/" + fileName)));
```

- Probleme mit dem Code:
  - Oft kann es passieren wenn kein Charset (UTF-8) angegeben wird, dass beim Ausführen des Codes ein Fehler, so wie zum Beispiel eine StartException kommt und das Programm so auch nicht kompillierbar ist.
- Korrigierter Code:
  - Diesen Fehler habe ich in der readTeamsAndDriversFromFile(TEAM_FILE_NAME) und der readRacesFromFile(RACES_FILE_NAME)-Methode behoben.

```
    BufferedReader br = new BufferedReader(new
InputStreamReader(getClass().getResourceAsStream("/" + teamFileName),
StandardCharsets.UTF_8));
```

Aufgabe 2: Import Rest

ResultsRestClient.java

Definieren des aktuellen Rennfahrers:

- Alter Code

```
    Driver currentDriver = query.getSingleResult();
    System.out.println(currentDriver);
```

- Probleme mit dem alten Code:
  - Mit diesem Code bekomme ich immer einen "no entity found for query" - Error
- Lösung des Problems ⇒ korrigierter Code

- Das if-Statement wird dazu gebraucht, um zu überprüfen, ob der aktuelle Fahrer auch in der Datenbank gespeichert ist. Ist er ein, oder auch mehrmals in der Datenbank gespeichert, so wird der erste Eintrag in der Liste verwendet

```
    List<Driver> drivers = query.getResultList();
    if (drivers.size() >= 1){
        Driver currentDriver = drivers.get(0);
        System.out.println(currentDriver);
        ...
    }
```

Definition des aktuellen Rennens:

- Alter Code:

```
    //String raceId = object.get("raceNo").toString();
    //int raceNumber = Integer.parseInt(raceId);
    //Race currentRace = this.em.createNamedQuery("Race.findById",
Race.class).setParameter("ID", raceNumber).getSingleResult();
```

- Probleme mit dem alten Code:
  - Der Code konnte nicht funktionieren, da er auskommentiert war.
  - Der definierte String ist kein "richtiger" String und funktioniert beim Übergeben nicht.
  - Der String wird zwar mit Integer.parseInt(raceId) in einen int umgewandelt, jedoch ist die ID des Rennens ein long und kein int und so kann die query nicht funktionieren.
- Neuer Code:
  - Es gibt die Funktion .getInt und so kann ich gleich einen int vom JsonObject lesen und diesen dann als long kasten.

```
    int raceId = object.getInt("raceNo");
    long raceNumber = (long) raceId;
    Race currentRace = this.em
        .createNamedQuery("Race.findById", Race.class)
        .setParameter("ID", raceNumber)
        .getSingleResult();
```

Definition der aktuellen Position, erstellen des Results und in DB speichern

- Alter Code

```
//            String currentPosition = object.get("driverFullName").toString();
//            int currentPos = Integer.parseInt(currentPosition);
//
//              Result currentResult = new Result();
//              currentResult.setRace(currentRace);
//              currentResult.setPosition(currentPos);
//              currentResult.setDriver(currentDriver);
//              currentResult.setPoints(currentPos);
//
//            this.em.persist(currentResult);
//            System.out.println(object);
```

- Probleme mit dem alten Code
  - Da der Code auskommentiert war, konnte er nicht funktionieren.
  - Der String namens currentPosition gibt den aktuellen Fahrer aus und kann daher auch nicht in einen int geparst werden.
  - Die aktuelle Position kann nicht als Position des aktuellen Results gesetzt werden, da ein int erwartet wird und auch schon ein Fehler beim Parsen geworfen wird.
  - Auch die Punkte können nicht im aktuellen Result gesetzt werden, da eben currentPos nicht verwendet werden kann. Außerdem sind es auch nicht die Punkte für den aktuellen Platz.
- Neuer Code

```
    int currentPosition = object.getInt("position");
    Result currentResult = new Result();
    currentResult.setRace(currentRace);
    currentResult.setPosition(currentPosition);
    currentResult.setDriver(currentDriver);
    int currentPoints = currentResult.pointsPerPosition[currentPosition];
    currentResult.setPoints(currentPoints);
    this.em.persist(currentResult);
    System.out.println(object);
```

Aufgabe 3: Gesamtpunkte eines Fahrers

- Result:
  - Erstellung einer NamedQuery, zum Auslesen der Gesamtpunkte eines bestimmten Fahrers.

```
    @NamedQueries({
            @NamedQuery(
                    name = "Result.getPointSumofDriver",
                    query = "select sum(r.points) from Result r where r.driver =
:DRIVER"
            )
    })
```

- ResultsEndpoint
  - Alter Code:

```
    @GET
    @Path("name")
    @Produces(MediaType.APPLICATION_JSON)
    public JsonObject getPointsSumOfDriver(@QueryParam("name") String name) {
        TypedQuery<Driver> query = em.createNamedQuery("Driver.findByName",
Driver.class).setParameter("NAME", name);

        Driver driver = query.getSingleResult();

        return null;
    }
```

- Probleme mit dem alten Code:
  - Der Code war noch nicht fertig programmiert.
  - Die Methode gab null und kein Ergebnis zurück.
- Korrigierter Code:
  - Zuerst hole ich mir den eingegebenen Fahrer von der Datenbank.
  - Anschließend wird zu diesem Fahrer die Gesamtpunkteanzahl der Rennen ermittelt.
  - Dann wird ein JsonObjectBuilder erstellt, dem der Fahrer und die Punkteanzahl hinzugefügt werden.
  - Als Rückgabewert wird der JsonObjectBuilder gebaut.

```
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public JsonObject getPointsSumOfDriver(@QueryParam("name") String name) {
        TypedQuery<Driver> query = em.createNamedQuery("Driver.findByName",
Driver.class).setParameter("NAME", name);

        Driver driver = query.getSingleResult();

        long sumPoints = em
                .createNamedQuery("Result.getPointSumofDriver", Long.class)
                .setParameter("DRIVER", driver)
                .getSingleResult();

        JsonObjectBuilder jsonObjectBuilder = Json.createObjectBuilder();
        jsonObjectBuilder.add("driver", driver.getName());
        jsonObjectBuilder.add("points", sumPoints);

        return jsonObjectBuilder.build();
    }
```

Aufgabe 4: Sieger eines bestimmten Rennens

- Result.java:
  - Um den Sieger eines bestimmten Rennens zu bestimmen, benötige ich eine neue NamedQuery in meiner Result-Entity, welche den Fahrer des gesuchten Rennens an der 1.Position zurückgibt.

```
    @NamedQuery(
            name = "Result.getWinner",
            query = "select r.driver from Result r where r.race = :RACE and r.position
= 1"
    )
```

- ResultsEndpoint.java
  - Zuerst suche ich das Rennen aus der Datenbank, welches in dem übergebenem Land stattgefunden hat.
  - Danach ermittle ich mit der NamedQuery aus der Result-Class den Sieger des Rennens.
  - Als Rückgabewert gebe ich einen Response mit dem Sieger zurück.

```
    @GET
    @Path("winner/{country}")
    @Produces(MediaType.APPLICATION_JSON)
    public Response findWinnerOfRace(@PathParam("country") String country) {
        Race race = em
                .createNamedQuery("Race.findByCountry", Race.class)
                .setParameter("COUNTRY", country)
                .getSingleResult();

        Driver winner = em
                .createNamedQuery("Result.getWinner", Driver.class)
                .setParameter("RACE", race)
                .getSingleResult();

        return Response.ok(winner).build();
    }
```

Aufgabe 5: Liste der Rennen, die ein Team gewonnen hat

- Driver.java:
  - Um eine Liste aller gewonnen Rennen eines Teams zu erstellen, benötige ich zuerst alle Fahrer, welche für dieses Team fahren. Dazu habe ich eine NamedQuery erstellt, welche mir die Fahrer eines Teams aus der Datenbank ausliest.

```
    @NamedQuery(
            name = "Driver.findByTeam",
            query = "select d from Driver d where d.team = :TEAM"
    )
```

- Result.java:
  - Außerdem benötige ich aus der Results-Tabelle die gewonnen Rennen eines jeden Fahrers, der für das Team fährt. Auch dafür habe ich eine NamedQuery erstellt.

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

- ResultsEndpoint.java:
  - Zuerst hole ich mir das Team, welches an die Methode übergeben wird, aus der Datenbank und speichere es auf die Variable team.
  - Danach erstelle ich eine Liste mit allen Fahrern dieses Teams.

- Dann erstelle ich eine Liste für die gewonnen Rennen des Teams und eine für die gewonnen Rennen des aktuellen Fahrers.
- Danach gehe ich in einer forEach-Schleife die Fahrer des Teams durch und speichere ihre Siege. Diese speichere ich dann in die Liste der Teamsiege.
- Die Methode gibt die Liste aller Siege des Teams zurück.

```
@GET
@Path("raceswon")
@Produces(MediaType.APPLICATION_JSON)
public List<Race> racesWonByTeam(@QueryParam("team") String teamName){
    Team team = em
            .createNamedQuery("Team.findByName", Team.class)
            .setParameter("NAME", teamName)
            .getSingleResult();

    List<Driver> drivers = em
            .createNamedQuery("Driver.findByTeam", Driver.class)
            .setParameter("TEAM", team)
            .getResultList();

    List<Race> wonRaces = new LinkedList<>();
    List<Race> wonRaceOfDriver;

    for (Driver driver : drivers) {
        wonRaceOfDriver = em
                .createNamedQuery("Result.getWonRacesOfTeam", Race.class)
                .setParameter("DRIVER", driver)
                .getResultList();

        for (Race race : wonRaceOfDriver) {
            wonRaces.add(race);
        }

    }
    return wonRaces;
}
```

Aufgabe 6(für Spezialisten): Liste aller Fahrer mit ihren Punkten

- Result.java:
  - Um eine Liste aller Fahrer mit ihren Punkten auszugeben, benötige ich eine NamedQuery, welche mir den Fahrernamen und die Summe seiner Punkte ausgibt.

```
    @NamedQuery(
        name = "Result.getAllDriversWithPoints",
        query = "select r.driver.name, sum(r.points) from Result r group by
r.driver.name"
    )
```

- ResultsEndpoint.java:
  - Um eine Liste aller Fahrer mit ihren Punkten auszugeben, erstelle ich eine Liste von einem Array von Object, in welche ich das Ergebnis der NamedQuery schreibe.
  - Dieses Ergebnis wird schließlich ausgegeben.

```
    @GET
    @Path("all")
    @Produces(MediaType.APPLICATION_JSON)
    public List<Object[]> allDriversWithPoints(){
        List<Object[]> allDriversAndPoints = em
                .createNamedQuery("Result.getAllDriversWithPoints", Object[].class)
                .getResultList();

        return allDriversAndPoints;
    }
```