Testverbesserung Schauflinger

InitBean.java

alte readRacesFromFile

```
        try {
            BufferedReader br = new BufferedReader(new InputStreamReader(getClass()
.getResourceAsStream(racesFileName)));
            br.readLine();
            String line;
            while ((line = br.readLine()) != null){
                String [] row = line.split(";");
                System.out.println(row);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

neu

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

        try {
            br.readLine();
            String line;
            while ((line = br.readLine()) != null){
                String [] row = line.split(";");
                Race race = new Race(Long.parseLong(row[0]),
                        row[1],
                        LocalDate.parse(row[2], DateTimeFormatter.ofPattern(
"dd.MM.yyyy")));

                em.persist(race);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

alte readTeamsAndDriversFromFile

```
        try {
            BufferedReader br = new BufferedReader(new InputStreamReader(getClass()
.getResourceAsStream(teamFileName)));
            br.readLine();
            String line;
            while ((line = br.readLine()) != null){
                String [] row = line.split(";");
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

neu

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

        try {
            br.readLine();
            String line;
            while ((line = br.readLine()) != null){
                String [] row = line.split(";");
                persistTeamAndDrivers(row);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

persistTeamandDrivers

war nicht vorhanden

Driver.java

Annotation @NamedQueries vergessen

```
@NamedQueries({
        @NamedQuery(
                name = "Driver.findByName",
                query = "select d from Driver d where d.name like :NAME"
        ),
        @NamedQuery(
                name = "Driver.findAll",
                query = "select d from Driver d"
        )
})
```

Result.java

Annotation @NamedQueries vergessen

```
@NamedQueries({
        @NamedQuery(
                name = "Result.getPointsSumOfDriver",
                query = "select sum(r.points) from Result r where r.driver = (select
d.id from Driver d where d.name like :NAME)"
        ),
        @NamedQuery(
                name = "Result.getWinner",
                query = "select re.driver from Result re where re.position = 1 and
re.race = (select ra.id from Race ra where ra.country like :COUNTRY)"
        ),
        @NamedQuery(
                name = "Result.wonRaces",
                query = "select re.race from Result re where re.position = 1 and
re.driver in (select distinct d.id from Driver d where d.team = (select t.id from Team
t where t.name like :TEAMNAME))"
        ),
        @NamedQuery(
                name = "Result.allPoints",
                query = "select sum(r.points) from Result r where r.driver = :ID"
        ),
})
```

Team.java

Annotation @NamedQueries vergessen

```
@NamedQueries(
        @NamedQuery(
                name = "Team.findByName",
                query = "select t from Team t where t.name like :NAME"
        )
)
```

ResultEndpoint.java

Entitymanager hinzugefügt

```
@PersistenceContext
    EntityManager em;
```

Client und Webtarget hinzugefügt

```
public static final String url = "http://vm90.htl-leonding.ac.at/results";
private Client client = ClientBuilder.newClient();
private WebTarget target = client.target(url);
```

readResultsFromEndpoint

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

persistResult

```
        for(JsonValue jsonValue : resultsJson) {
            Long raceNo = Long.parseLong("" + jsonValue.asJsonObject().getInt("raceNo
"));
            int position = jsonValue.asJsonObject().getInt("position");
            String name = jsonValue.asJsonObject().getString("driverFullName");

            em.persist(new Result(em.find(Race.class, raceNo),
                    position,
                    em.createNamedQuery("Driver.findByName", Driver.class)
                            .setParameter("NAME", name)
                            .getSingleResult()));

        }
```

ResultsEndpoint.java

@Stateless wird nicht benötigt

```
    @GET
    public JsonObject getPointsSumOfDriver(@QueryParam("name") String name) {
        Long points = em.createNamedQuery("Result.getPointsSumOfDriver", Long.class)
                .setParameter("NAME", name)
                .getSingleResult();

        JsonObject jsonObject = Json.createObjectBuilder()
                .add("driver", name)
                .add("points", points)
                .build();

        return jsonObject;
    }


        @GET
    @Path("winner/{country}")
    @Produces(MediaType.APPLICATION_JSON)
    public Response findWinnerOfRace(@PathParam("country") String country) {

        Long driverId = em.createNamedQuery("Result.getWinner", Driver.class)
                        .setParameter("COUNTRY", country).
                        getSingleResult().
                        getId();

        Driver winner = em.find(Driver.class, driverId);

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

diese Klasse aus Zeitproblemen nicht gemacht

eigene Methoden:

```
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    @Path("raceswon")
    public List<Race> allRacesWonByTeam(@QueryParam("team") String team) {
        List<Race> wonRaces = em.createNamedQuery("Result.wonRaces", Race.class)
                .setParameter("TEAMNAME", team)
                .getResultList();

        return wonRaces;
    }

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    @Path("all")
    public List<String[]> allRacesWonByTeam() {
        List<Driver> drivers = em.createNamedQuery("Driver.findAll", Driver.class)
.getResultList();
        List<String[]> driverWithPoints = new LinkedList<>();

        for (Driver driver: drivers) {
            Long points = em.createNamedQuery("Result.allPoints", Long.class)
                    .setParameter("ID", driver)
                    .getSingleResult();
            driverWithPoints.add(new String[]{driver.toString(), "" + points});
        }

        return driverWithPoints;
    }
```