Project Documentation For M11 Vehicles and DB PartC

Author: Mason Fraser
Date: 08-Nov-19

Subject

Table of Contents

1. Project Overview

Implement JDBC Database into existing 3D Vehicles, save their state in a database so it can be reloaded from previous state.

2. Project Requirements

Continue with your Vehicles assignment...and modify it to the following:

1) Complete outstanding Vehicles updates from the midterm, using proper OOD with an abstract (base) class.
2) Use a database to store data (How many Vehicles , of which type, where, and how fast). The Vehicles at the start come from the database, from the completed run of last time the program ended.

3) Use a sequence diagram to show relevant movement of data in the system.

3. Database Implementation

SQLite has been implemented to allow for a save state, the system will save on a button press (Numpad_7), which places all the items on screen into an array and send it of in a insert query to the database, on next boot, everything saved on screen will return in its last-saved position.

3.1. Persistent Database

Used a simple persistent database application provided by Karl Meissner.

3.1.1. Saving the Data

```
4.  if (window.isKeyPressed(GLFW_KEY_KP_7)) {
        try {
            Thread.sleep(100);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        em.getTransaction().begin();

        Query q2 = em.createQuery("delete FROM ObjDbConverter obj",
    ObjDbConverter.class);
        q2.executeUpdate();

        for (GameItem items: gameItems) {
            ObjDbConverter obj = new ObjDbConverter(items.getPosition().x,
    items.getPosition().y, items.getPosition().z, items.getThisIs());
            em.persist(obj);
        }
        em.getTransaction().commit();

        TypedQuery<ObjDbConverter> query =
                em.createQuery("SELECT obj FROM ObjDbConverter obj",
    ObjDbConverter.class);
        List<ObjDbConverter> results = query.getResultList();
        for (ObjDbConverter p : results) {
            System.out.println(p);
        }
```

The way I handle this, is by creating a new object, a data container for all the important data that is contained within Vehicle objects, and with this new "Converter object", which gets stored into the ObjectDB as a separate thing, which later we pull from and recrate the objects.

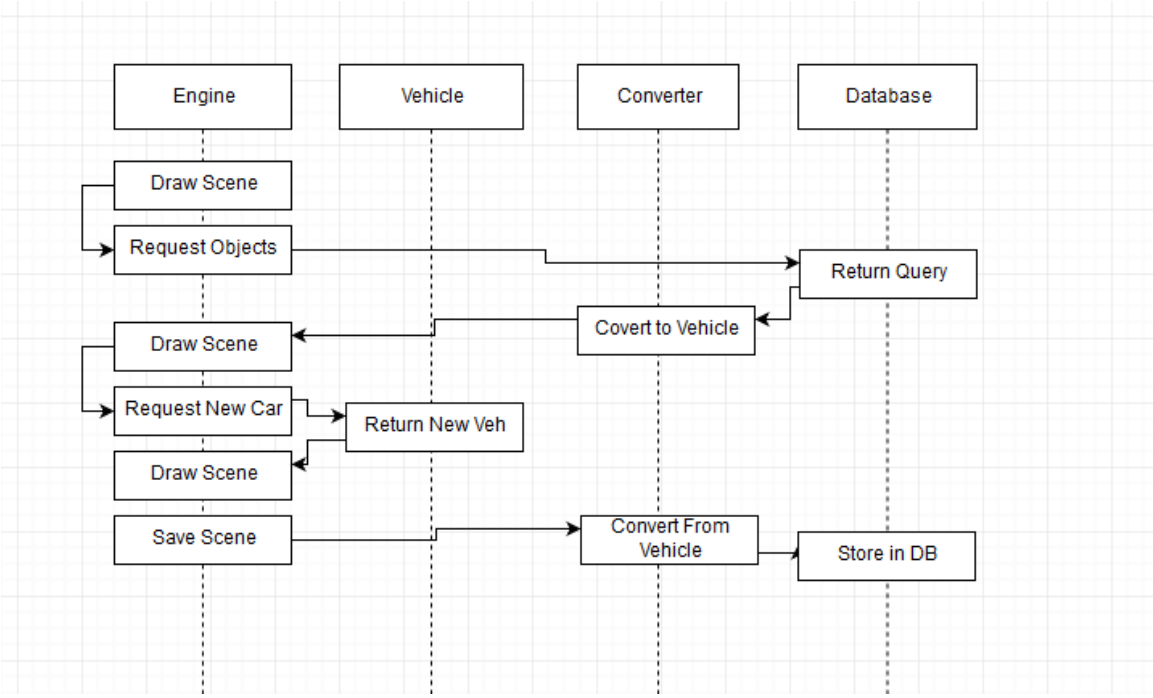
4.1.1. Loading Data

```
5.
          TypedQuery<ObjDbConverter> q2 = em.createQuery("SELECT obj FROM
    ObjDbConverter obj", ObjDbConverter.class);
          List<ObjDbConverter> results = q2.getResultList();

          for (ObjDbConverter obj: results) {
              if(obj.getVeh_type() != null){
                  if(obj.getVeh_type().equals("LandVehicle")) {
                      Vector3f vector = new Vector3f(obj.getX(), obj.getY(),
    obj.getZ());
                      Vehicle house = new LandVehicle(vector, 0, 0, 100);
                      house.setVelocity(0.002f, 0.000f, 0.003f);
                      house.setRotationVel(new Quaternionf(0.06f, 0.01f,
    0.03f, 0.0f));
                      house.setScale(0.150f);
                      scene.setGameItems(new GameItem[] {house});
                  }
                  if(obj.getVeh_type().equals("AirVehicle")) {
                      Vector3f vector = new Vector3f(obj.getX(), obj.getY(),
    obj.getZ());
                      Vehicle house = new AirVehicle(vector, 0, 100);
                      house.setVelocity(0.002f, 0.000f, 0.003f);
                      house.setRotationVel(new Quaternionf(0.06f, 0.01f,
    0.03f, 0.0f));
    //                    house.setScale(0.150f);
                      scene.setGameItems(new GameItem[] {house});
                  }

              }
          }
```

Here we handle checking what is in the database, and creating new objects based on what they are and adding them to the game scene, you can see I check what type of item I'm adding, if the object is not null (Null will always be terrain), then compare against what items are stored in the results of the query.

This then pulls everything from the database, and compares against the data. If the data type is a vehicle, which is stored as a string, it then creates a new vehicle object and stores it on screen.

5.1. Sequence Diagram



This diagram shows the sequence of data being passed around the database and game client.