CAS CS 411

Software Engineering

Lab 2 - DAOs and Databases

Goal

The goal of this lab is to build upon lab 2 and include a data access object (DAO) for the SQLite database.

For the front-end, we will be using vanilla Javascript. For the back-end, we will be using Python/Flask.

Feel free to use different technologies if you desire. However, these lab instructions will be for the technologies outlined above.

Data Access Object (DAO)

Until now, the model has been responsible for storing the lists used for `cards`, `state`, and `matched`. However, in many real-world applications, we will want the model to store information in a database.

In this lab, we will be storing two of our lists (`cards` and `matched`) in a database. The DAO's responsibility is to:

1. Provide an interface (through getters and setters) so the model can easily access `cards`, `state`, and `matched`.
2. Implement the details for reading/writing `cards` and `matched` to a database.

We have provided the abstract DAO class in `concentration_daos/concentration_dao.py`. Take a look at this class to understand the high level goal.

Implementation

Part 1: Dummy DAO

Before thinking about any database details, lets create a dummy DAO. This dummy DAO does nothing special except store our three lists.

The dummy DAO class can be found in `concentration_daos/dummy_concentration_dao.py`

Your tasks are to:

1. Implement the getter/setter for cards.
2. Implement the getter'setter for matched.

Hint, these will be identical to the getter/setter for cards in the abstract DAO class.

Part 2: Add Dummy DAO to Model

We will use this dummy DAO to transition the model logic to rely on DAOs. We have provided a new model class for you. Please look at how the dummy DAO is used in the constructor (for now, ignore the `dao_identifier` argument to the constructor).

Your tasks are to:

1. Fill in the rest of the model with your code from lab 2 using the DAO to access the model's data.

After completion, your concentration game should run the same as lab 2. Please test to make sure this is the case.

SQLite DAO

Now we will implement a DAO for a SQLite database. This DAO can be found in `concentration_daos/sqlite_concentration_dao.py`.

Take a look at the constructor first. We create a connection to an SQLite database that we create locally. We then create two tables in that database to store `cards` and `matched`.

Additionally, we implemented the getter and setter for `cards`.

Implementation

Part 3: Finish SQLite DAO

You need to finish the implementation for the getter and setter for the `matched` list. This will be almost identical to that of cards but with one caveat:

SQLite automatically stores booleans as a binary integer value `0` (`False`) and `1` (`True`). When reading the list in the matched getter, you must convert integer values back to boolean values.

Part 4: DAO Selection

To put everything together, we will allow the model to run either the dummy DAO or the SQLite DAO. This is where the dao_identifier string in its constructor is used.

Your tasks are to:

1. Use the dummy DAO when `dao_identifier=None`.

2. Use the SQLite DAO when `dao_identifier='sqlite'`.

3. In your controller, design an easy way to set the `dao_identifier` of the model.