Homework 6: MVC Model Associations

For this homework, you will further refine your team's web app by adding associations between some of your previously created model classes and by updating your seed data and actions/views to use these associations. Additionally, you will continue to practice with the version-control system, Git.

You will do this homework as a team; however, each member of your team will be responsible for the completion of a particular task. Each team member will choose one task from the list below to complete. All team members must do a different task. If your team has only three members, then ignore Task 4.

The Tasks

Similar to previous homeworks, there will be four tasks (Tasks 1 through 4); however, this time the tasks are more inter-related. Each task will have the same five parts:

1. Add validations and unit tests to the model class that has none.
2. Add two model associations.
3. Add seed data that uses the associations.
4. Update views/controllers to make use of the associations.
5. Update the home page to link to your newly created pages.

Part 1: Add validations and unit tests

Figure 1 (below) depicts a class diagram with the previously created classes and their associated tasks. In Homework 4, you created validations and unit tests for one the model classes for your task. In this homework, you must add validations and unit tests for the other model class.

In particular, add at least one validation per attribute of the class. Also, write at least one unit test for each attribute that checks that the attribute's validation can catch an invalid value. The choice of validations and tests is largely up to you, but choose something sensible (i.e., not too weird) and have at least 3 types of validation overall (i.e., two attributes may have the same type of validation).

Part 2: Add two model associations

Implement two one-to-many associations, as shown in Figure 1 (see below). You must create the associations labeled with your task number. Your Rails model associations must match the class diagram exactly—including the role names on the association ends.

Part 3: Add seed data

Add seed data as follows. You must have at least three records from the "one" side of each of your associations, and each of those records must be associated with at least two records from the "many" side of the association. Thus, for a given association, there will need to be a minimum of 9 records (3 from the one side + 2 + 2 + 2 from the many side).

In creating the seed data, you may need to coordinate with your teammates to come up with sensible data given their constraints. To keep the total number of seed objects manageable, it is OK for you to "share" model objects with teammates in the seeds file. That is, you may count the same model object toward the objects required for multiple associations.

Part 4: Update views/controllers

Using your newly created associations, you must update views/controllers as per the task-specific requirements below:

Task 1:

- Update the Planet *show* page such that it includes a table listing all the Continent objects that the Planet object has. The table should be inserted below the usual Planet *show* info and be styled like the table in the Continent *index* page.

- Update the Continent *index* page such that a new column is added to the table. The column should be titled "Homeworld", and it should display the *name* attribute of the Planet object to which the Continent belongs.

- The Continent *new*/*create* and *edit*/*update* pages should now include a dropdown field (`collection_select`) that allows the user to select the Planet object to which the Continent object will belong. The text for each item in the dropdown should be the *name* attribute of the Planet. In the *edit* form, the dropdown field should select the Continent's current Planet by default.

Task 2:

- Update the Country *show* page such that it includes a table listing all the City objects that the Country object has. The table should be inserted below the usual Country *show* info and be styled like the table in the City *index* page.

- Update the City *index* page such that a new column is added to the table. The column should be titled "Country", and it should display the *name* attribute of the Country object to which the City belongs.

- The City *new*/*create* and *edit*/*update* pages should now include a dropdown field (`collection_select`) that allows the user to select the Country object to which the City object will belong. The text for each item in the dropdown should be the *name* attribute of the Country. In the *edit* form, the dropdown field should select the City's current Country by default.

Task 3:

- Update the Owner *show* page such that it includes a table listing all the Building objects that the Owner object has. The table should be inserted below the usual Owner *show* info and be styled like the table in the Building *index* page.

- Update the Building *index* page such that a new column is added to the table. The column should be titled "Manager", and it should display the *first_name* and *last_name* attributes of the Owner object to which the Building belongs.

- The Building *new*/*create* and *edit*/*update* pages should now include a dropdown field (`collection_select`) that allows the user to select the Owner object to which the Building object will belong. The text for each item in the dropdown should be the *last_name* attribute of the Owner. In the *edit* form, the dropdown field should select the Building's current Owner by default.

Task 4:

- Update the Park *show* page such that it includes a table listing all the Statue objects that the Park object has. The table should be inserted below the usual Park *show* info and be styled like the table in the Statue *index* page.

- Update the Statue *index* page such that a new column is added to the table. The column should be titled "Park", and it should display the *name* attribute of the Park object to which the Statue belongs.

- The Statue *new*/*create* and *edit*/*update* pages should now include a dropdown field (`collection_select`) that allows the user to select the Park object to which the Statue object will belong. The text for each item in the dropdown should be the *name* attribute of the Park. In the *edit* form, the dropdown field should select the Statue's current Park by default.

Part 5: Update the home page

Update your project's home page so that it now lists your name (no hyperlink) followed by the following two links:

Task 1:

- "Planets" linking to Planet *index* page.
- "Continents" linking to Continent *index* page.

Task 2:

- "Countries" linking to Country *index* page.
- "Cities" linking to City *index* page.

Task 3:

- "Owners" linking to Owner *index* page.
- "Buildings" linking to Building *index* page.

Task 4:

- "Statues" linking to Statue *index* page.
- "Parks" linking to Park *index* page.

How to submit your team's work

Before you can submit, all team members must have merged their code into the master branch and pushed the updates to GitHub. If a team member does not complete his/her work on time, you may submit without his/her contribution.

To submit your team's work, you must "tag" the current commit in the master branch:

```
$ git tag -a hw6v1 -m 'Tagged Homework 6 submission (version 1)'
$ git push origin --tags
```

To grade your work, I will check out the appropriate tag, and run it on my machine.

Note that if for some reason you need to update your submission, simply repeat the tagging process, but increment the version number (e.g., hw6v2, hw6v3, hw6v4, etc.).

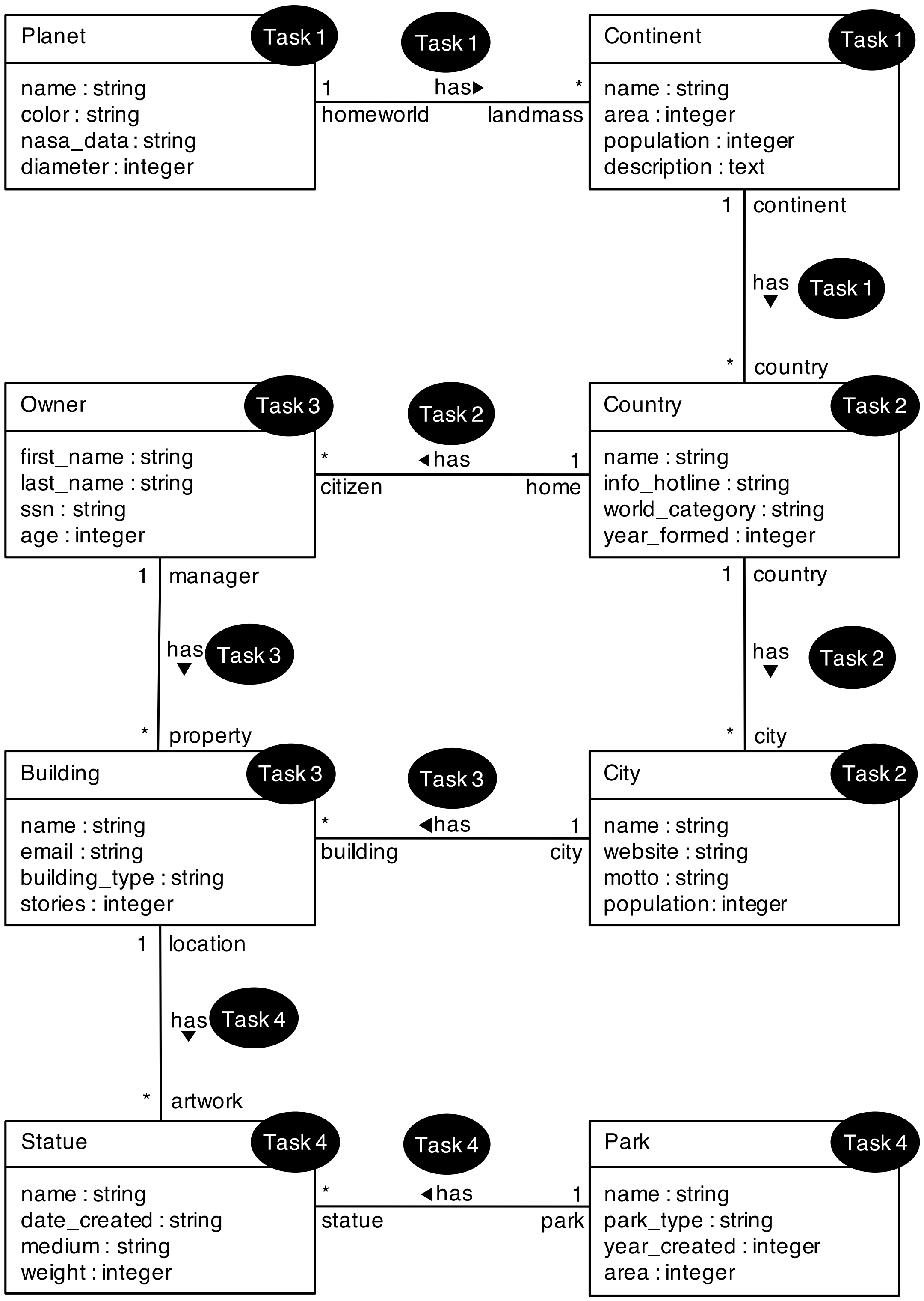

Figure 1. Model class diagram with task assignments.