Assignment 9 – Epic Superhero Showdown

Goals

- Use database for persistent data
- Use singletons to manage database
- Use MVC to store data and interaction with fragments

Required naming convention *(replace # with the current assignment number)*

- **Application Name**
  - *A#*
- **Company Domain**
  - *Firstname.lastname.itp341*
- **Package Name**
  - *Itp341.lastname.firstname.A#*

Overview / Goal

- You are building a superhero battle simulator. Each superhero has two superpowers and an initial health of 5 points. The user selects two superheroes to fight, and they battle until one of their health values falls below 0.
- During every round of the battle, a power from each hero is randomly chosen and the winning power of that round is determined using a lookup table (stored in a database table). When the battle is finished, the results (win / lose / tie) are written back to the database.
- The database comes pre-loaded with a table of superheroes and a table of powers.
- There is also an option to add a new superhero to the database
- There is <u>no requirement</u> to allow the user to modify existing heroes in the database
- There is <u>no requirement</u> to allow the user to add new / modify existing powers to the database

Requirements

- Create new Android Application Project
  - Min SDK: API 22
  - Follow default prompt, but make sure to choose **Empty Activity**.
- UI
  - Two activities + fragments
    - **`MainBattle`** Activity / Fragment

        - **AddHero** Activity / Fragment
    - **MainBattle** (figures 1-3)
        - Four **TextViews** with labels
        - **ListView** that show the rankings
        - **Button** to launch **AddHero** activity
        - Two **spinners** that load lists of existing heroes
        - **Button** to simulate hero fighting
        - **ScrollView** with a **TextView** inside to display results
    - **AddHero** (figures 4-5)
        - **EditText** for name
        - Two **TextViews** for "Power" label
        - Two **Spinners** to display the available powers
        - **Button**
- Model – **Hero** class
    - Instance variables
        - **_id : long**
        - **name : String**
        - **power1 : String**
        - **power2 : String**
        - **health : int**
        - **numWins : int**
        - **numLosses : int**
        - **numTies : int**
    - Methods
        - Overloaded constructor
        - **toString**
        - Getters / setters
        - **isAlive**
            - input: none
            - output: **boolean**
            - Returns **true** if **health** => 0; **false** otherwise
- Singleton
    - In addition to the standard methods in a singleton class, it is suggested to have the following:
    - **getUniquePowers**
        - input: none
        - output: either **cursor** or **ArrayList<Hero>**
        - Queries the **power** table and returns all the powers in the database

        - Hint: there is an version of the **`query`** method for SQLite that allows for selecting **`DISTINCT`** rows
    - **`addHero`**
        - input: **`hero`**
        - output: void
        - Take a **`hero`** object and writes the values to the **`heroes`** table
    - **`getRankings`**
        - input: none
        - output: either **`cursor`** or **`ArrayList<Hero>`**
        - Retrieves all the records in the **`heroes`** table sorted in descending order by **`num_wins`**
    - **`getHeroes`**
        - input: none
        - output: either **`cursor`** or **`ArrayList<Hero>`**
        - Retrieves all the records in the **`heroes`** table sorted in ascending order by **`name`**
    - **`getPowerResult`**
        - input: two stings, **`power1`** and **`power2`**
        - output: **`int`** representing which power won the round
            - 1 = **`power1`** wins
            - -1 = **`power2`** wins
            - 0 = tie
        - Queries the **`powers`** table using the two inputs and return the result
        - Use this method to determine the winner of a given round
    - **`addBattleResult`**
        - input: **`Hero`** object, and **`int result`** (which represents which hero won the battle)
        - output: none
        - Based on the value of **`result`**, update the row in the **`heroes`** table to increase the number of wins, or the number of losses, or the number of ties for that hero
            - 1 = **`power1`** wins
            - -1 = **`power2`** wins
            - 0 = tie
- **`MainListBattle`** Activity / Fragment (Figures 1-3)
    - When the activity starts:
        - Obtain all the rankings data from the **`heroes`** table (who has the most wins) and load the **`listView`** at the top of the screen (figure 1)

        - Obtain the names of all the heroes from the **`heroes`** and load the hero **`spinners`** (figure 2)
    - If the user clicks **add,** launch the **`AddHero`** activity / fragment
    - If the user clicks **`fight`**
        - Simulate fighting between the two selected (via **`spinners`**) heroes
        - Display the text output of the battle in the **`TextView`**
        - Update the **`heroes`** database table and update the rankings **`ListView`** (figure 3)
    - *What happens during a battle?*
        - Each hero starts with 5 health points
        - Hero1 and hero2 will "fight" multiple rounds
        - In each round, a power is randomly selected from hero1 and hero2
        - Using the **`powers`** table, determine who wins that round
        - The loser (or both players in the event of a tie) lose one health point
        - The heroes keep "fighting" until one (or both) of their health is equal to zero
- **`AddHero`** Activity / Fragment (figures 4-5)
    - This screen should first obtain all the powers from **`powers`** table and load those values into the two **`spinners`** (make sure there are no duplicate powers)
    - When the user clicks **`save`**, the new hero should be added to the **`heroes`** table and the user should be taken back to the main screen
- Database class – **`DBHelper`**
    - This class is provided for you entirely so copy it in your project's appropriate **`src`** folder (be sure to update the **package** line at the top
    - The purpose of this class is to pre-load the database file **`heroes.db`** (with **`heroes`** and **`powers`** tables)
    - If the database does not exist, the class will automatically copy the pre-built file into your emulator.
- Database class – **`DB Schema`**
    - This class is provided for you entirely so copy it in your project's appropriate **`src`** folder
    - The purpose of this class is to help make it easier to reference columns in the database
- Database file – **`heroes.db`**
    - The initial database file is included so copy it in your project's **`assets`** folder
    - The asset folder is in **`app/src/main/assets`**
    - See the Excel file **`Database Tables.xlsx`** for a visual representation of the data

- **heroes** table
  - Columns
    - **_id**
    - **name**
    - **power1**
    - **power2**
    - **num_wins**
    - **num_losses**
    - **num_ties**
- **powers** table
  - Columns
    - **_id**
    - **own_power**
    - **opposing_power**
    - **winning_power**
  - The table below is another way to visualize which power wins in a dual

	F	**G**	**I**	**L**	**A**	**S**
F		G	F	L	A	F
G	G		I	G	G	S
I	F	I		L	I	I
L	L	G	L		A	S
A	A	G	I	A		A
S	F	S	I	S	A	

Power Definitions:

F: Flight
G: Gadgets
I: Superhuman intelligence
L: Laser shooting eyes
A: Adamantium claws
S: Superhuman strength

- Extra Credit
  - Create a custom adapter for the rankings **listview**
    - Figures 6-8 show a custom **adapter** and custom row layout. Use any layout you like, but each row should show wins, losses, ties, name, and powers
  - Create **AsyncTasks** for all databases operations
  - Create custom **CursorWrappers** for abstracting the database cursors from the fragment code

Deliverables

1. A compressed file containing your app. Follow the guidelines for full credit.
   Here are the instructions for submission
   a) Navigate to your project folder.
   b) Include the *entire* folder in a zip file

c) Rename the zip file so it follows this convention: *A#.lastname.firstname*
d) Upload zip file to Blackboard site for our course

Note: Test app on Nexus 5 AVD or with Genymotion's Nexus 5

Grading

Item	Points
UI for fragments	10
Model class	5
Singleton class	10
MainBattle load rankings	5
MainBattle display possible heroes	5
MainBattle battle functions successfully	15
MainBattle saves results	5
AddHero loads unique powers	5
AddHero saves new hero / adds to DB	5
Total	**65**

Sample Output

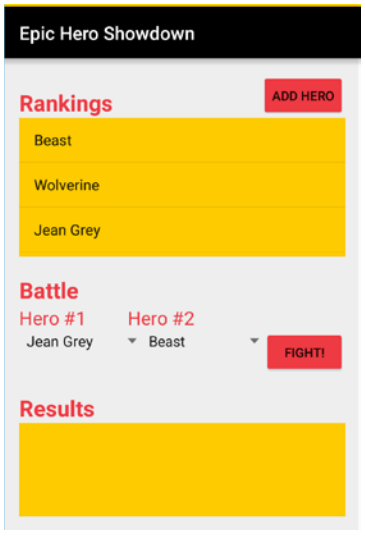


Figure 1

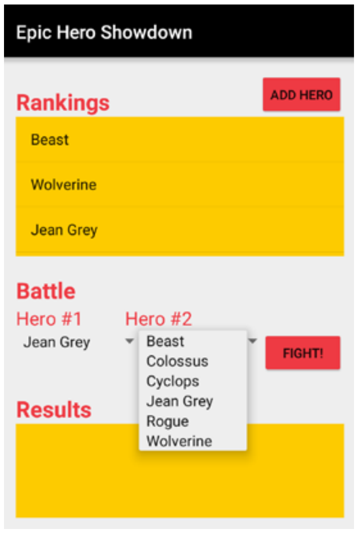
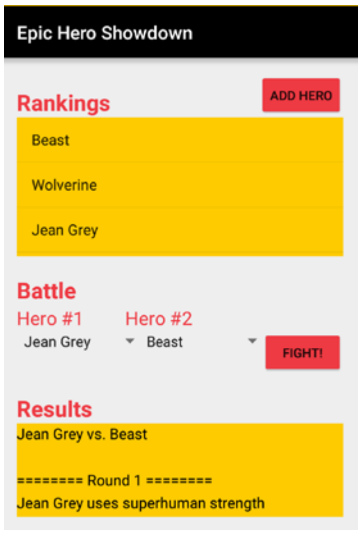


Figure 2

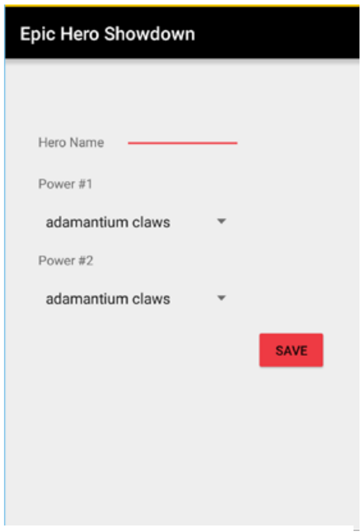


```
Jean Grey vs. Beast

======== Round 1 ========
Jean Grey uses superhuman strength
```

Figure 3

```
Jean Grey vs. Colossus

======== Round 1 ========
Jean Grey uses flight
Colossus uses superhuman strength
Jean Grey wins this round with flight

======== Round 2 ========
Jean Grey uses flight
Colossus uses superhuman strength
Jean Grey wins this round with flight

======== Round 3 ========
Jean Grey uses superhuman strength
Colossus uses superhuman strength
Jean Grey and Colossus tie this round

======== Round 4 ========
Jean Grey uses superhuman strength
Colossus uses superhuman strength
Jean Grey and Colossus tie this round

======== Round 5 ========
Jean Grey uses flight
Colossus uses superhuman strength
Jean Grey wins this round with flight

Jean Grey is the winner!
```

Sample Text Output

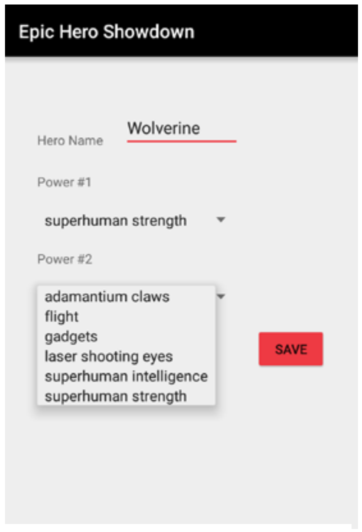


Hero Name

Power #1: adamantium claws

Power #2: adamantium claws

Figure 4

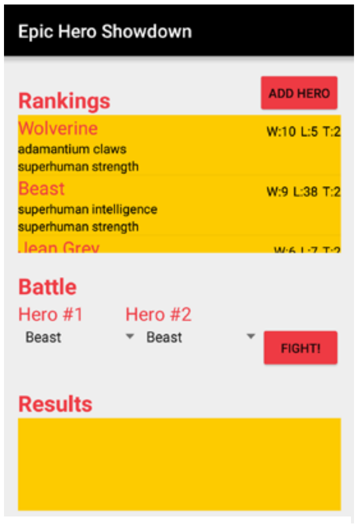


Hero Name: Wolverine

Power #1: superhuman strength

Power #2

Figure 5



Rankings

Wolverine W:10 L:5 T:2
adamantium claws
superhuman strength

Beast W:9 L:38 T:2
superhuman intelligence
superhuman strength

Jean Grey W:6 L:7 T:2

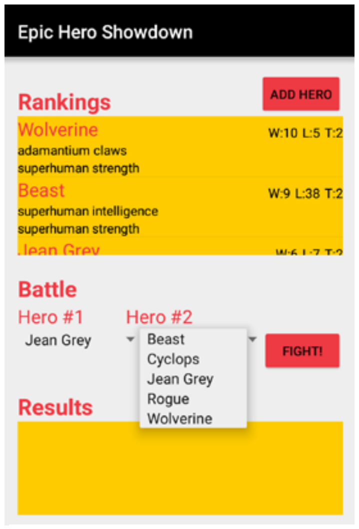
Battle

Hero #1: Beast

Hero #2: Beast

Results

Figure 6



Rankings

Wolverine W:10 L:5 T:2
adamantium claws
superhuman strength

Beast W:9 L:38 T:2
superhuman intelligence
superhuman strength

Jean Grey W:6 L:7 T:2

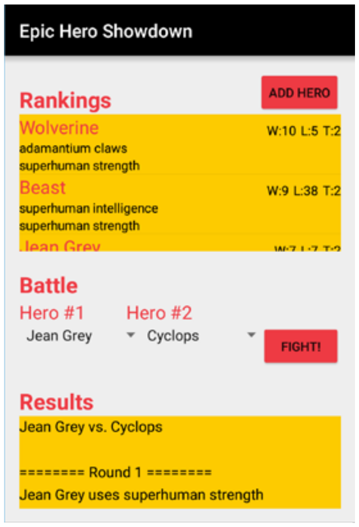
Battle

Hero #1: Jean Grey

Hero #2

Results

Figure 7



Rankings

Wolverine W:10 L:5 T:2
adamantium claws
superhuman strength

Beast W:9 L:38 T:2
superhuman intelligence
superhuman strength

Jean Grey W:7 L:7 T:2

Battle

Hero #1: Jean Grey

Hero #2: Cyclops

Results

```
Jean Grey vs. Cyclops

======== Round 1 ========
Jean Grey uses superhuman strength
```

Figure 8