Module 2 Assignments Help -- UML Diagram and Project One Milestone Game App

Posted Jan 13, 2025 11:03 AM

All -- Just to help you out a bit on the assignments, please find below some explanations/hints as to how you should proceed.

In the UML diagram, few fields and methods are missing -- you need to fill in the blanks:

GameService

-games: List<Game>

-nextGameId: long

???

???

???

+ addGame(name: String):
Game

+ getGame(index: int): Game

+ getGameCount(): int_

Regarding the Java Game App assignment (Project One Milestone Game App), I am sharing below partially completed GameService class that use singleton and iterator patterns. The rest of the tasks in the assignment would be similar or use similar concepts. I hope you would be able to complete the rest of the assignment on your own leveraging these 2 examples.

<u>Singleton Example (In orange color below):</u>

```
public class GameService {


	// A list of the active games

	private static List<Game> games = new ArrayList<Game>();

	// Holds the next game identifier

	private static long nextGameId = 1;


	// -> Add missing pieces to turn this class a singleton

	// create an object

	private static GameService instance = new GameService();

	// private constructor so no objects can be created elsewhere

	private GameService() {

	}

	// get the only object available

	public static GameService getInstance() {
```

```
                return instance;

        }
```

.

.

.

.

.

.

__**Iterator Example (In orange color below):**__

```
public Game addGame(String name) {

                // a local game instance

                Game game = null;
```

```
Iterator<Game> it = games.iterator();

// creates iterator of type Game

// to iterate through the 'games' ArrayList

while (it.hasNext()) { // while it has NOT reached the end of the list

        if (it.next().getName().equals(name)) { // if the game's name equals the name parameter

                game = it.next(); // local game instance assigned with found instance

        }

}


// if not found, make a new game instance and add to list of games

if (game == null) {

        game = new Game(nextGameId++, name);

        games.add(game);

}


// return the new/existing game instance to the caller
```

```
                return game;
        }
}
```

Hope this helps!

Thanks,

Suhash