Use-Case Model

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 3/20/2016 | 1.0 | Initial Commit | Daian Dragos |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Use-Cases Identification 4

2. UML Use-Case Diagrams 4

Use-Case Model

# Use-Cases Identification

This document discusses the actors, scenarios and use cases.

**Use case: Start Scene**

**Level: summary level**

**Primary actor: Gamer**

**Main success scenario: Starts a level**

**Extensions: Scene loads but some objects give error(non blocking errors)**

**Use case: Test/Give Logic**

**Level: goal level**

**Primary actor: Gamer/Tester**

**Main success scenario: Give input to the logic part of the game**

**Extensions: Input device not available, nothing happens.**

**Use case: Add Entities**

**Level: summary level**

**Primary actor: Designer**

**Main success scenario: Start scene-> add different entities**

**Extensions: No objects already defined, programmer has to code them.**

**Use case: Edit Scene**

**Level: summary level**

**Primary actor: Programmer**

**Main success scenario: Add different assets in scene.**

**Extensions: -**

**Use case: Add logic**

**Level: goal level**

**Primary actor: Programmer**

**Main success scenario: Starts a level -> create entity -> add script component**

**Extensions : Logic has syntactic errors, no logic is added, no game breaking error is given.**

**Use case: Add component/Change Component**

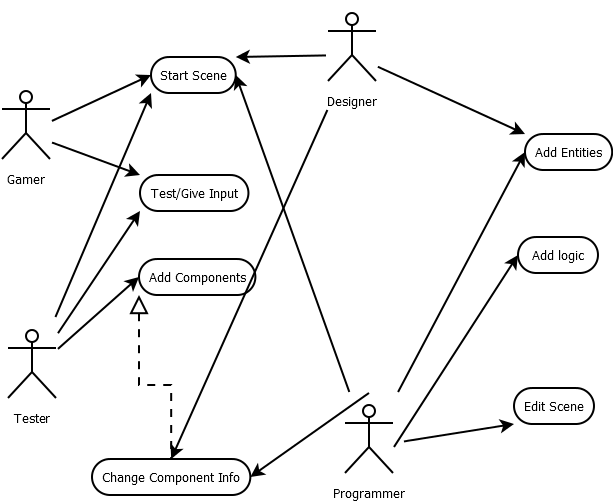
**Level: summary level**

**Primary actor: Programmer**

**Main success scenario: Add data to be saved or logic to be saved**

**Extensions: -**

# UML Use-Case Diagrams



As you may see above, the gamer can only start the game and send input to it.

The tester can add components and also test the game.

The designer can as well test as well as add new entities, like designing a level.

The programmer can do all, which also includes adding logic(scripts) and editing scenes.