

Draw It o Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc115077317)

[**Table of Contents 2**](#_Toc115077318)

[**Document Revision History 2**](#_Toc115077319)

[**Executive Summary 3**](#_Toc115077320)

[**Requirements 3**](#_Toc115077321)

[**Design Constraints 3**](#_Toc115077322)

[**System Architecture View 3**](#_Toc115077323)

[**Domain Model 3**](#_Toc115077324)

[**Evaluation 4**](#_Toc115077325)

[**Recommendations 5**](#_Toc115077326)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 01/28/24 | Matthew Allen |  |

**Instructions**

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

## [Executive Summary](#_sbfa50wo7nsh)

Draw It or Lose it is a game that requires there to be only one instance of a specific game at a time. Within that game, there can only be one instance of a specific team, and only instance of each player on that team. To achieve this goal, there needs to be a static class that handles instances of each game. A static class means there can only be one instance of the class in memory at a time, and therefore, each class that calls information for that class will access the same instance. With this static class, the next game, team, and player number will be tracked, allowing only one instance, identified by a unique number, will be available.

## Requirements

## [Design Constraints](#_2et92p0)

Network latency will affect the transfer of data over the internet. This could affect the game performance and user experience, because a poor network connection could lead to data transfer errors and delays in information loading. The system will need to handle network issues gracefully.

Since the user can play this game on any browser, the use of cross browser compatible libraries is required. Also, because different browsers have different security requirements, it is necessary to meet the basic standards for the most common ones.

## [System Architecture View](#_ilbxbyevv6b6)

Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

## [Domain Model](#_8h2ehzxfam4o)

Entity is a parent class that holds common properties and methods for an “entity” of the Draw it or Lose It game. The Game class, Team class, and Player class as inherit from the Entity class because they all require an ID and name, along with getters and setters for those properties. The Team class uses the Players class, and the Game class uses the Team class. The GameService class holds a list of Game class instances and handles the incrementation of the game, team, and player ID numbers. ProgramDriver holds the main function and uses the SingleTest class to test for proper functionality.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

Using your experience to evaluate the characteristics, advantages, and weaknesses of each operating platform (Linux, Mac, and Windows) as well as mobile devices, consider the requirements outlined below and articulate your findings for each. As you complete the table, keep in mind your client’s requirements and look at the situation holistically, as it all has to work together.

In each cell, remove the bracketed prompt and write your own paragraph response covering the indicated information.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac offers a user-friendly interface, a Unix-based operating system, and compatibility with many web tools and frameworks. However, Mac is also more expensive, less customizable, and less widely used than other platforms. | Linux offers stability, security, and compatibility with many open-source tools and languages. However, Linux has a steeper learning curve, less user-friendliness, and fewer technical support options than other platforms. | Windows offers a user-friendly interface, compatibility with Microsoft technologies, and integration with various web tools and frameworks. However, Windows has higher cost, lower security, and less flexibility than other platforms. | Mobile devices are portable and ubiquitous devices that can access web-based software applications through browsers or native apps. However, mobile devices have limited resources, such as memory, storage, battery, and network. |
| **Client Side** | Cost: May require special licenses that are more costly than normal.  Time: Complying with apple standards and guidelines may add time to the SDLC.  Experience: Limits amount of users because they are less popular than other platforms. | Cost: Linux is often open source, thus the cost is low.  Time: Linux has a steep learning curve and thus will be more time consuming.  Experience: linux is unpopular and community maintained. It would limit the user pool. | Cost: Windows platforms aremore affordable than Mac  Time: There are many resources available for windows users  Experience: The most popular OS and it’s relatively intuitive. | Cost: Can be expensive for the user to have a phone capable of running the software  Time: can be time consuming because of the variety of mobile apps  Experience: the experience will vary greatly due to the variety of phones. |
| **Development Tools** | Languages: Swift, objective C  IDE: Xcode, CLion | Languages: C/C++, Python, Java  IDEs: VS Code, Eclipse, NetBeans | Languages: C#, Java  IDEs: Visual Studios, VS Code | Langues: Swift, Objective C, Java  IDEs: Xcode, android studio |

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

1. **Operating Platform**: Windows. It is the most popular and can be written in java, which is a portable language.
2. **Operating Systems Architectures**: The Windows architecture is a layered client-server design that consists of two main components, user mode and kernel mode. User mode is a restricted area of the operating system that is for running applications and drivers. Kernel mode is the heart of the operating system where the core components run.
3. **Storage Management**: Oracle Cloud is a cloud based storage system that is compatible with Windows and will allow the software to scale.
4. **Memory Management**: It will use garbage collection, memory pools, and memory profiling.
5. **Distributed Systems and Networks**: Web services, such as HTTP, soap, and XML, can be accessed by different platforms and languages, and are scalable.
6. **Security**: To protect user information on and between various platforms, the software can use encryption, authentication, and authorization techniques, such as HTTPS, SSL/TLS, OAuth, and SAML.