

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template**](#_l6ti7uoag22u)1

[**Table of Contents**](#_30j0zll)2

[**Document Revision History**](#_grjogdjh5fi8)2

[**Executive Summary**](#_sbfa50wo7nsh)3

[**Design Constraints**](#_2et92p0)3

[**System Architecture View**](#_ilbxbyevv6b6)3

[**Domain Model**](#_8h2ehzxfam4o)3

[**Evaluation**](#_2o15spng8stw)3

[**Recommendations**](#_m8aleynsvzvc)5

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 5/30/2022 | Max Freeman | Created new entity class, extended through player and team classes and added iterator pattern to them as well. |
| 2.0 | 06/17/2022 | Max Freeman | Updated recommendations to provide holistic roadmap for success. |

**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)

Wanting to use singleton method to ensure only one instance of game and only one instance of player name and team name. We are trying to constrain instances across the game platform to only unique values.

## [Design Constraints](#_2et92p0)

Must be implemented in the Java Runtime Environment, and modular so that maintenance and deployment of future features can be low-cost and time-sensitive for client.

## [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)

The entity class will extend through the Team and Player classes. The SingletonTester will be referenced ala carte by the main method to ensure only one instance of items from Game, Team and Player classes exist. Program Driver class will hold our one and only main method.

**"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 has considerable market share for web hosting, can be scaled and takes very flexible commands from the terminal. Drawback would be less deployment as a server solution than other providers. | Low-cost option and very secure. Reputation for preventing viruses from becoming problematic. According to colleagues and experience, preferred option is Linux for web hosting. Compatibility of supporting software can be an issue. | High degree of comfort with WindowsOS as it is most common OS/largest market share. Not lowest cost and can be notably vulnerable to bugs and security flaws. Must be more closely monitored from security standpoint. Nearly any software compatible. | Not a preferred option due to potential security vulnerabilities and variability of support for software, etc. Specifications for devices can be difficult to navigate. Portable solution that is a popular feature and lower-cost according to research of the marketplace. |
| **Client Side** | Time will likely be required to train employees for the environment. Cost is comparable to slightly less than Windows according to marketplace research. | Time will be required to train employees for the environment. Broad past use by employee population unlikely. Least costly option for software but most expensive in terms of time to train. | Little to no time likely required for training employees for environment. Functions and commands will be familiar to most. Cost for OS will be higher than other options according to marketplace research. | Very flexible, may or may not require extensive training (probably case-by-case decision based on exact solution chosen). Implementation will likely take longer this way than others but cost of OS may be lower. |
| **Development Tools** | Unable to identify any languages that cannot be developed in the MacOS environment. Notepad++ is great as well as other tools available. No notes otherwise. | Not able to confirm that all languages compatible in development, but major languages (HTML/CSS/JS/Java/Python/Ruby in the VS environment look sound, as well as PostGreSQL, MySQL, and PGAdmin, etc. Development shouldn’t be hindered when working in major languages/with major tools. | Compatible with all known IDE/server/programming tools that I researched. Compatibility is maximized with WindowsOS. | Primary research into Swift, which will work seamlessly. Compatibility in Java looks sound, but may or may not be device and specifications specific and so more research would be required before deployment. HTML/CSS/JS/Python were confirmed across nearly all solutions. |

## 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**: WindowsOS to maximize compatibility. Development will work best when programming team has maximum selection of solutions and no time lost to relearning processes in an unfamiliar OS.
2. **Operating Systems Architectures**: Entire suite of WindowsOS based tools available for everything from direct system file support for Windows apps to instant messaging/email/exchange server/AD and enterprise security and threat assessment tools to be enabled across all streamed users.
3. **Storage Management**: WindowsOS added StorageSense with its release of Windows10. You can also save to a cloud service of your choice of use OneNote, which integrates beautifully with Windows offering of SharePoint.
4. **Memory Management**: Specific to the Draw It or Lose It, multimedia data management will be crucial. WindowsOS is no better or worse at storing images outside the default “Pictures” or “Videos” folder for our specific use cases within the development and production environment of the application.
5. **Distributed Systems and Networks**: Multiple offerings on the WindowsOS for cross-platform development style. A specific solution identified on the marketplace was Develop 4, along with a slew of others that will make this simple to handle across distributed systems.
6. **Security**: WindowsOS allows for a seamless integration of SSO (single-sign-on) with built-in MFA (multi-factor authentication). Windows also provides their own MFA application where you can generate and store all your MFA codes (continuously changing on intervals for maximum security). Threat assessment and mitigation tools are also available. I recommend deployment of Mimecast with your WindowsOS solution to add a layer of protection to email servers against phishing. Employees will be able to self-service their Mimecast settings to allow bounced/rejected items through when they trust the sender, while employees will be directly refused access to links in emails with hallmark signs of phishing or fraud (IT can override this when we trust the source).