

Draw it or 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 | 03/16/23 | Oliver Zinser | Creating a plan for, Draw it |

**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)

Create a program that allows us to create games, teams and players. These will have unique names (in their respective groups).

## Requirements

* *A* design pattern that allows us to assure that only one instance of a game is created.
* We want each game to be able to create team, and each teams create players.

## [Design Constraints](#_2et92p0)

* The software must be developed using Java programming language.
* The game needs to be able to run while also waiting for user input.
* The game needs to be accessible by people using different platforms.

## [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)

In the top-left we have the ProgramDriver and the SingltonTester classes. The ProgramDriver class holds the main method and use it to run the rest of the program. The SingltonTester class is used ONLY by the main method (in the ProgramDriver) to test if the program is working probably.

The next section starts with the Entity class, this holds some commonly (in this program) used variables and methods and is extended by the Game, Team, and Player classes. The GameService class is a singleton style class, so it can’t be initialized more than once, it holds an array of Game objects, and each one can be accessed by a name, id, or index. In the Game class it holds a list of Team objects, Team follows the same pattern except with Player objects.

**"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** | MacOS has the disadvantage of not being specifically designed to host websites, although it certainly works, and has the benefits of being extremely secure | Linux is arguably the best choice to host a web-based serve, as it’s very open source (depending on the distribution) and relatively secure. | Windows is a good choice for hosting webservers (it’s what I always use) as it has great support and lots of tutorials but is not particularly secure | BAD IDEA always, if you need a portable server use a raspberry pi, not made to host servers in the slightest (especially IOS) and will make it very hard |
| **Client Side** | Good choice all around, all new macs/MacBooks have extremely good performance, so you know that you have more power than let's say a mobile device.  Less support though for things like games, but if this is a HTML web game then no problem | Different Linux distributions may cause unexpected problems even when using something like a browser, plus the fact that Linux isn’t as popular as other OS makes Linux not a very good candidate to focus attention on | Is a great platform to develop games/applications for as it has lot’s of support and backwards compatibility. Although you won’t know what the specs of the windows computer are (could be extremely old) but still always important to support windows devices | It's always a good idea to support mobile device but because of the lack of input choices (no keyboards or precise mouse movement) making it compatible or fair can be difficult. Everyone has one though so it’s important to provide support. |
| **Development Tools** | Can be annoying as usually, you need a MacBook to create an application for a MacBook, but lots of support with programs like XCode or IntelliJ. | Linux is definitely the most annoying software to learn from the beginning. The reason it is so difficult is because of the multiple different distribution. Linux provides a wide range of powerful development tools that can help developers create high-quality software efficiently. However, some of these tools can be complex to use and may require some learning curve. | Windows has lots of support for different programming languages and tons of tools to help with the creation of applications, I generally use Visual Studio 2023, VS Code, and IntelliJ | Mobile, while doesn’t have as many supported languages, still has good options such as Java or Kotlin (I’ve used both). |

## 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**:

After careful consideration we recommend Windows. This platform has the following advantages:

* + - Support. Windows has a long history of support and has a large community of developers with experience using it.
    - Microsoft office. Windows devices work well with Microsoft suit of products which make working as a team easier.
    - Ease of use. Windows 10 (or Windows 11) are easy to use and most issues are simple to debug, this will make production time faster and more efficient.

These are the reasons why we chose windows over other popular environments such as MacOS or Linux.

1. **Operating Systems Architectures**:

Windows has two separate modes, User mode, and Kernel mode. User mode processes are mostly user-facing, and this is what the user is interacting with most of the time (if you are a more casual user; all the time). Kernel mode is much lower-level, and it deals with memory management, inputs and outputs, networking, and managing hardware. Windows uses a directory structure to hold data. Windows also has support for multiprocessing.

1. **Storage Management**: We recommend using cloud storage. Since we will want to potentially scale this program up later, to house more images (or higher resolution) having a storage solution that is easy to scale is a good choice. That along with cloud storage being so low-cost makes this a great choice for “Draw it or Lose it”.
2. **Memory Management**: The chosen operating system (Windows 10) uses various memory management techniques to make sure that there’s optimal performance for software. The system uses demand paging to load only the required parts of the software into the memory when it’s needed. Windows 10 also uses disc paging to allocate additional virtual memory from the hard drive, this provides a larger memory pool for the software to work with. Additionally, Windows 10's memory compression feature compresses data in memory to free up more space for other processes. These techniques work together to ensure that Draw It or Lose It will run smoothly and efficiently on Windows 10.
3. **Distributed Systems and Networks**: For Draw It or Lose It to communicate effectively between platforms, a distributed software architecture and network connectivity is needed. Technologies like web services, message queues, or RPC can enable communication between machines. The network architecture must make sure that there is a high availability and fault tolerance to avoid system failures. On Windows 10, options for implementing distributed systems and networks include WCF and WinRM, enabling communication across platforms and reliable communication between Draw It or Lose Its components.
4. **Security**:

To make sure that your users have peace of mind and to protect them the following criteria need to be met:

* + Login info such as username and password cannot be stored as plain text on the serv-side OR client-side, this is to ensure that if the user or yourselves have compromised any information that could be leaked wouldn’t contribute to damage to your users.
  + Passwords stored on your servers must be stored as a hash, this prevents anyone including employees from seeing your user’s info.
  + Employees need to be prepared and know best practices in how to use the systems provided. This is to make sure that a hacker couldn’t use human error to gain access to your systems and carry out ransomware attacks etc.