

Draw It or Lose It

# **CS 230 Project Software Design Template**

Version 3.0

## 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)4

[**Recommendations**](#_m8aleynsvzvc)7

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0               2.0  3.0 | 05/20/2022               06/05/2022  06/09/2022 | Kasey Myers | Critical information is added under the Executive Summary, Design Constraints, Domain Model, Evaluation and Recommendations sections. The executive summary outlines the software design problem and poses a solution to the problem. The design constraints section identifies other technical implementations such as programming language(s) that will be used, OS that will be used, etc. The domain model section presents a UML diagram and describes the diagram in terms of relations and principles. In the evaluation section, critical details and information are added to the development requirements/OS chart. Lastly, the recommendations section will have my recommendations for each heading listed and detailed. Made no additional revisions. Completed document on 05/20/2022.  No additional revisions. |

## [Executive Summary](#_sbfa50wo7nsh)

Successful application developer The Gaming Room is looking to expand by creating a web-based game that serves multiple platforms based on their hit Android gaming application Draw It or Lose It. CTS will assist in this expansion by analyzing the requirements, evaluating the operating system options, and making recommendations based upon the analysis and evaluation to find the appropriate operating system required for the task at hand.

## [Design Constraints](#_2et92p0)

* The program should utilize the Java programming language and debugging should be completed upon completion of code.
* The program should utilize the API framework.
* The program should use in memory data structures.
* The program should be completed in no more than two weeks’ time.
* Singleton feature should be used.
* Testing should be generated and ran throughout the coding process.

## [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 below UML Diagram, we see that there are several classes inside the Gaming Room application Draw It or Lose It, and between those classes, there are three types of relationships present. First, we see that the ProgramDriver class uses the SingletonTester class. We can also clearly see that the Game, Team, and Player classes inherit from the Entity class. Finally, we have the zero or more relationship present between GameService and Game, Game and Team, and Team and Player classes. The two object-oriented programming principles displayed in the diagram are inheritance, as explained above, and encapsulation. Encapsulation is present in the restricted access to methods within the application.

**"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.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | * Excellent GUI (graphical user interface) * Ease of use including accessibility options * Server configuration ability * expensive | * Difficult to use, especially without a GUI * GUI options available although not as user friendly * Utilizes the command shell for server configuration * inexpensive | * expensive * Like Mac, Windows offers an excellent GUI * Command prompt * Wealth of software options | * Inexpensive * Options: iOS, Android, WordPress, etc. * Varying specifications among users * Creating a game that can cross platforms is difficult |
| **Client Side** | * Time: average * Skills required: medium. Prior knowledge necessary * Cost: expensive | * Time: maximum * Skills required: maximum. Proficiency in Linux required * Cost: minimal | * Time: minimal * Skills required: minimal. Easy to learn. * Cost: medium (more than Linux, less than Mac) | * Time: maximum * Skills required: maximum. Skills needed for multiple mobile platforms. * Cost: medium |
| **Development Tools** | * IDEs: Python, JavaScript, CSS, etc. * Dev. Tool: Ecplise, VS(VisualStudio), CodeBlocks, and multiple other online developing tools | * IDEs: HTML, Ruby on Rails, Python, Java, etc. * Dev Tools: nodejs, VisualStudio, Github, etc. | * IDEs: C languages (C++, C#, etc), Python, Java, HTML, etc. * Dev. Tools: There are many including but not limited to Eclipse, VisualStudio, CodeBlocks, etc. | * IDEs: php, Python, HTML, etc. * Dev. Tools: command prompt, VisualStudio, Github, etc. |

## 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**: The operating system that I recommend The Gaming Room use for its application Draw It or Lose It is Windows OS. I recommend Windows due to its wide selection of available software, ease of use due to its simple GUI, and wealth of abilities for developers.
2. **Operating Systems Architectures**: As mentioned about, the simplified GUI offers users the ease of use needed regardless of skills or familiarity. Windows is also compatible with a wealth of programming languages making the OS a prime choice for developers. For server configuration, Windows offers an easy-to-use Command Prompt power shell. And finally, as mentioned above, the wide variety of software made available on Windows surpasses that of the other OS options.
3. **Storage Management**: The common theme with Windows is its ease-of-use. Windows keeps its configuration settings simple making memory management effortless. Windows also offers cloud storage in the form of OneDrive that allows for easy storage and recall across multiple devices.
4. **Memory Management**: Windows is fitted with both virtual and physical memory space. Typically it allows for up to four gigabytes.
5. **Distributed Systems and Networks**: A distributed system offers excellent coordination and simple communication between nodes. However, these systems can have issues such as congestion, issues when one component fails, connection issues, and lagging.
6. **Security:** Windows offers an array of user account settings that are meant to help secure incoming and outgoing user data. These user accounts are also assigned with roles, such as administrator, furthering the securing the ability to make changes to the system without the appropriate assigned role. Lastly, Windows is designed with built in spyware and VPN capabilities protecting the device from malicious intent.