

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 | 03/16/21  03/31/21 | Keith Ellison | This document is meant to take a hands-on comprehensive look at our options for porting “Draw it or Lose 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)

The Gaming Room is seeking a developer that can produce a network version of their “Draw it or Lose it” app(currently only on Android) that can be accessed by other existing platforms. Essentially, they want a port of the existing game that’ll only allow one instance at a time to run.

## [Design Constraints](#_2et92p0)

The original can be ported to Java, assuming it’s not already developed in it. Only allowing one instance at a time is a pretty common limitation for video game apps, but I see it as our priority, it’s the step we need to keep in mind, while the others are game related, and can be addressed as an element of normal development.

## [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 “ProgramDriver” box contains our main loop and is the start of the program, that’s where we’ll place our Singleton Pattern’s code so that this will be the only instance of the program that can run at a given time. The Entity class defines an individual character through individual methods that are spun off into whole classes of their own to streamline the code, “Entity” will consolidate the variables from it’s child classes.



## [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** | Most game releases don’t make it to mac or take years in doing so, this platform is a low priority for development behind all but Linux. That said, it can be expected to develop a small but likely loyal user base that will stick with the game for years. | If we must develop for this platform, then the user base can be expected to be small, and of them, not many will stick with the game long, not many people use Linux as a gaming platform. Apart from that, Linux development would require specialist skills that not many potential applicants could be expected to have. | As the platform in widest circulation among these four, most developers will understand it’s architecture and we can expect the largest influx of people from this user base. | I expect this to be the most active platform during the early life of our game. We would physical need enough servers to handle the early onset of players, though a large monetary investment would hurt us long term once this audience settles and drops off, leaving us with excess server space. |
| **Client Side** | The Mac audience will always be small, but a well ported and supported version of the game may have some life to it. | Again Linux is the lowest priority for development, a flashy UI wouldn’t matter here, and likely neither would content. I feel there is nothing that can be done to build or maintain interest on this platform. | Similar to mobile, Windows users will expect a flashy UI with minimal bugs. A similar logic to keeping the game updated will keep the user base active, furthermore, if we allow it, user created content can keep the PC version alive into perpituity. | Here our biggest concern should be aesthetics, with the largest base of casual users, we need an interface that is appealing and through updates, stays appealing long term, that and content are the priorities here. |
| **Development Tools** | Apple’s Swift can be used to develop/port the game here and on IOS. Swift being open source may make porting code between this and the Windows version easier. | C++ or Python could be used, keeping us in line with a C++ development on the PC version, any Java portions from other versions can be written here in Python. | On Windows we will use Python and C++, since it would help with the potential version for Linux. | For mobile, we can probably do the entire game in Java for Android, or Python with Swift for IOS, port the whole thing to PC where extra functionality can be added. |

## 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**: As mentioned/referenced throughout, I feel that with it’s user base, near ubiquity in the sector, and public confidence, that Windows PC is the best operating platform to develop for outside of the console and mobile markets.
2. **Operating Systems Architectures**: With Microsoft Windows there is support for both x86 and x64 platforms within their architecture in an already established, already trusted, consumer friendly package, and finally, expanding out of the console and mobile spaces must include a Windows development strategy as any other platform trails Windows in all key criteria.
3. **Storage Management**: The gameplay is based on photos/images, easing our storage requirements. Even considering user profiles and other requisite information, for our storage needs I recommend Microsoft Windows Azure; They offer “Iaas”, infrastructure as a service, that coupled with our familiarity with the architecture would allow incremental outsourcing over the years as the company’s priorities shift to other projects.
4. **Memory Management**: Windows based desktop will most easily handle gameplay here, only needing 2GB of system memory, thus running smoothly on essentially any modern system. Obviously server architecture will fare even better with the mobile/network version of the game, I anticipate no issues on any platforms for a game like this; with it easily being stored within the RAM of again, any modern system.
5. **Distributed Systems and Networks**: Cross play is an emerging most wanted feature among consumers; to meet this need Microsoft Azure could be further coded on our end to support this, the game being on Xbox, is a Microsoft product which should give us a leg up in understanding how to connect the two, leaving the only hurdle as the Sony Playstation.
6. **Security**: Malicious users wouldn’t have much/anything to gain from attempting illegal mods to the game’s code, but that withstanding, our security needs would be offloaded to Windows Azure if we choose that platform; they protect the integrity and security of data, apps, and our infrastructure with in-built solutions for the aforementioned concerns, this is security for the networked mobile versions and the others iterations would need to connect to our server and pass certain security checks within Azure to be allowed access to the game and content updates.