Programming Process & Technical Design Document

Apa itu Technical Design Document? Apa fungsinya?

Site: DILo Game Academy
Course: Game Programming Studi Independen
Book: Programming Process & Technical Design Document
Printed by: 408 Dewa Sinar Surya
Date: Monday, 4 October 2021, 7:57 PM

Table of contents

1. Tahap 1 - Planning

Pada materi kali ini, kita akan mempelajari seperti apa role programmer pada saat fase planning game hingga menjadi dokumen yang bisa dijadikan sebagai patokan buat ngoding

Chapter 1: Planning

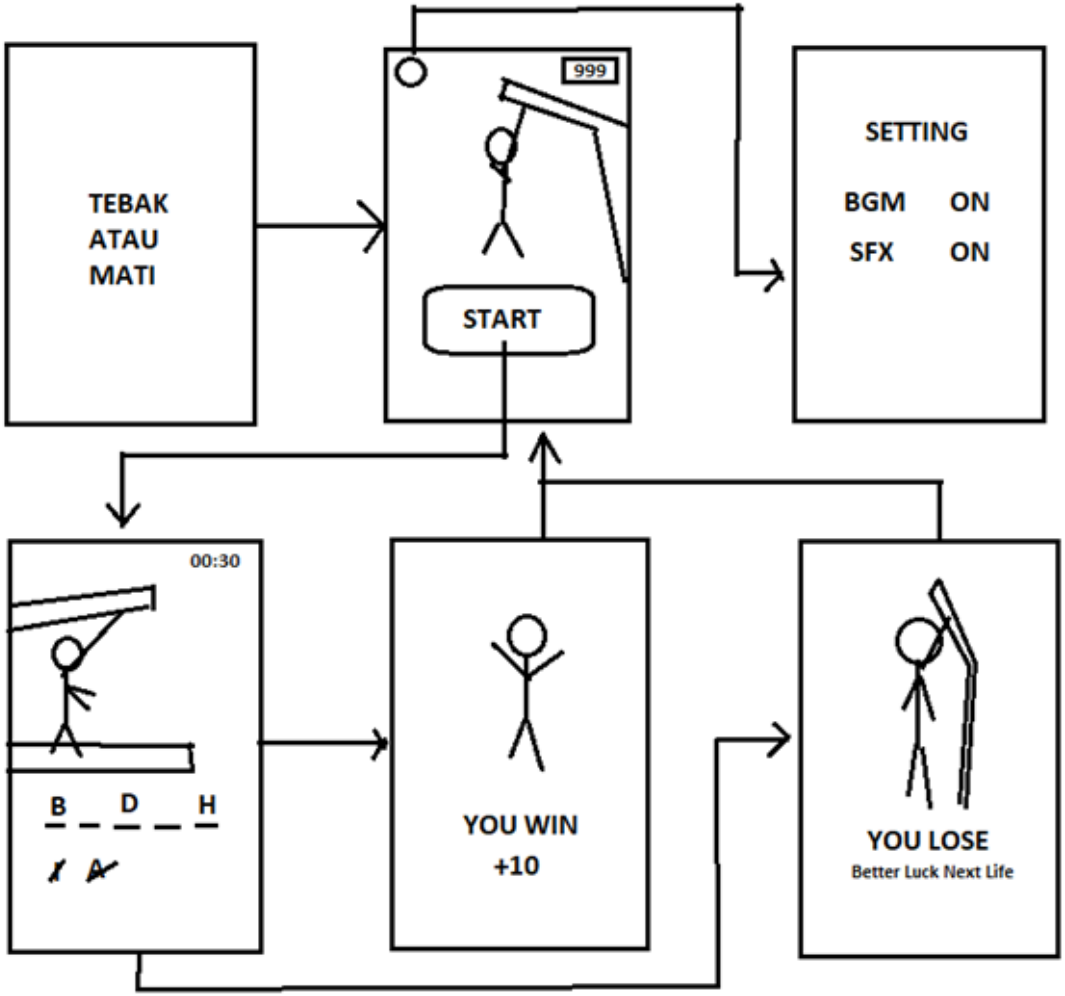
Biasanya di awal kita menjadi programmer untuk membuat game baru, kita akan bertemu dengan tim yang sudah ditentukan oleh produser. Kemudian tim akan melakukan diskusi atau meeting untuk menentukan game seperti apa yang akan dibuat. Biasanya meeting ini akan berjalan seperti ini:

- Perkenalan
- Penyampaian ide awal
- Brainstorming

Pada kasus ini sebenarnya kalian tidak perlu melakukan apapun, tapi lebih baik jika kalian mencatat fitur-fitur yang kemungkinan akan diimplementasi pada saat brainstorming.

Setelah meeting awal ini Game Designer akan ditugaskan untuk membuat mock up untuk game flow yang akan dibuat secara general. Biasanya mereka akan membuatnya dalam bentuk wireframe. Kemudian wireframe ini akan diberikan pada programmer baik dengan meeting lanjutan atau secara langsung.

Dari sini kita akan memulai membuat catatan terkait fitur2 yang ada pada desain mock up ini. Sebagai contoh kita mendapatkan plan untuk membuat game tebak kata. Kita diberi contoh game HangMan, akan tetapi pada game ini ditambahkan sistem waktu yang akan berkurang drastis jika salah menambahkan huruf.



Dari wireframe ini kita harus bisa mendeskripsikan game flow dan juga fitur-fitur yang dibutuhkannya. Jika dilakukan saat meeting ada baiknya menanyakan fitur-fitur yang ada pada game nya terlebih dahulu supaya Game Designer dapat mendeskripsikan secara rinci fitur-fitur yang diinginkan agar tidak terjadi miskomunikasi. Contohnya pada wireframe tadi kita mendapatkan informasi fitur sebagai berikut.

- Audio
- Bisa nyala mati
- BGM dan SFX dipisah

- Timer
- Sistem penalty saat salah huruf
- Ada event untuk waktu habis (untuk lose)
- Berjalan dengan animasi orang
- Point
- Bertambah saat menang
- Ditampilkan di main menu
- Sistem Game Hangman
- Ada event untuk salah huruf (untuk penalty)
- Ada event untuk semua terisi (untuk win)
- Ada lose dan win saat game over

Setelah membuat notes pastikan kalau notes yang dicatat sesuai dengan plan dari Game Designer, kita juga bisa mengutarakan pendapat yang lebih baik seperti menu bgm dan sfx disatukan saja karena dirasa lebih mudah misalnya. Atau kita juga bisa mengusulkan ide fitur baru, yaitu sistem skoring yang disesuaikan dengan performa kita di game, tidak constant +10 point misalnya.

Pada meeting ini juga biasanya programmer akan ditanya terkait kesanggupan dan tingkat kesulitan implementasinya. Fitur2 ini mungkin akan dipotong atau ditambahkan sesuai dengan tingkat kesulitan yang kita jelaskan tadi.

INGAT! Jangan sekali-sekali kita memberikan informasi yang kurang jelas disini. Kita sebagai programmer adalah seorang yang paling tahu kodingan fitur tersebut seperti apa, sedangkan tim kita tidak tahu apa-apa tentang ini. Jika memang hal ini adalah hal yang baru, maka katakanlah bahwa kita memang belum tahu bagaimana membuatnya dan mintalah waktu untuk mencari tahu terlebih dahulu bila perlu.

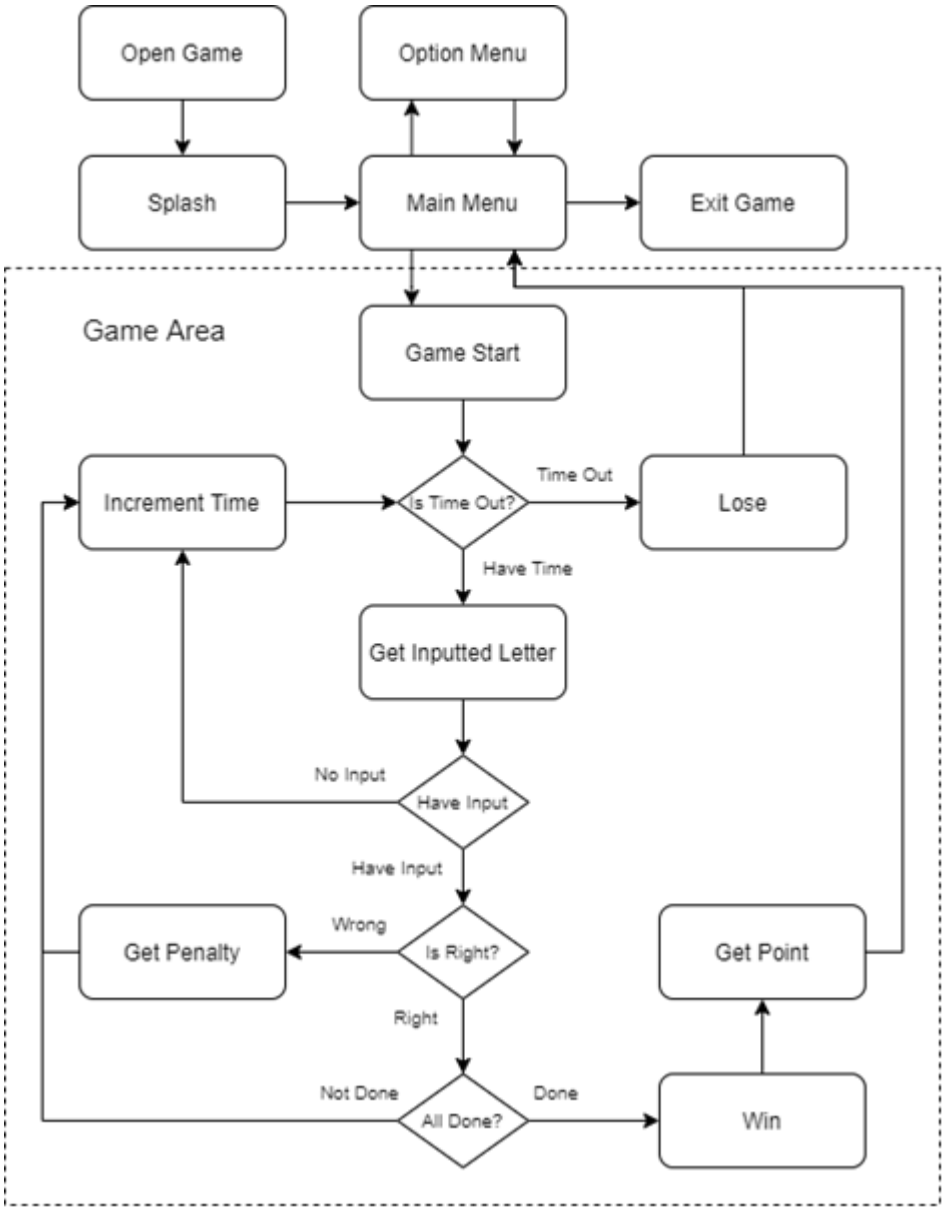
2. Tahap 2 - Menulis Dokumen

Pembuatan dokumen biasanya dilakukan sebelum game mulai dibuat, akan tetapi terkadang lebih mudah dilakukan saat kita membuat game nya atau bahkan saat prototype game nya sudah selesai. Terkadang pembuatan game juga memiliki deadline yang sangat padat sehingga pembuatan dokumen ini tidak dilakukan. Walaupun seperti itu, kita tetap perlu mengenal dokumen-dokumen yang dibutuhkan untuk pembuatan sebuah game ini agar kita dapat menjelaskan rencana dan dapat mengatur jadwal dengan baik.

Pada dasarnya dokumen untuk pembuatan game sama seperti pada pembuatan aplikasi-aplikasi lain. Disini hanya akan dibahas beberapa yang biasa digunakan oleh game programmer

Diagram Flow Chart

Flow chart ini sebenarnya tidak terlalu penting, tapi bisa lebih memudahkan programmer dalam membuat fitur yang dibutuhkan karena biasanya wireframe flow yang dibuat oleh desainer kurang lengkap untuk menjelaskan flow game yang sesungguhnya. Disini kita hanya perlu membuat flow saja untuk bagaimana game nya berjalan, tidak perlu terlalu detail. Sebagai contoh dari wireframe tadi kita akan membuat flow game dari mulai awal game sampai akhir game seperti ini

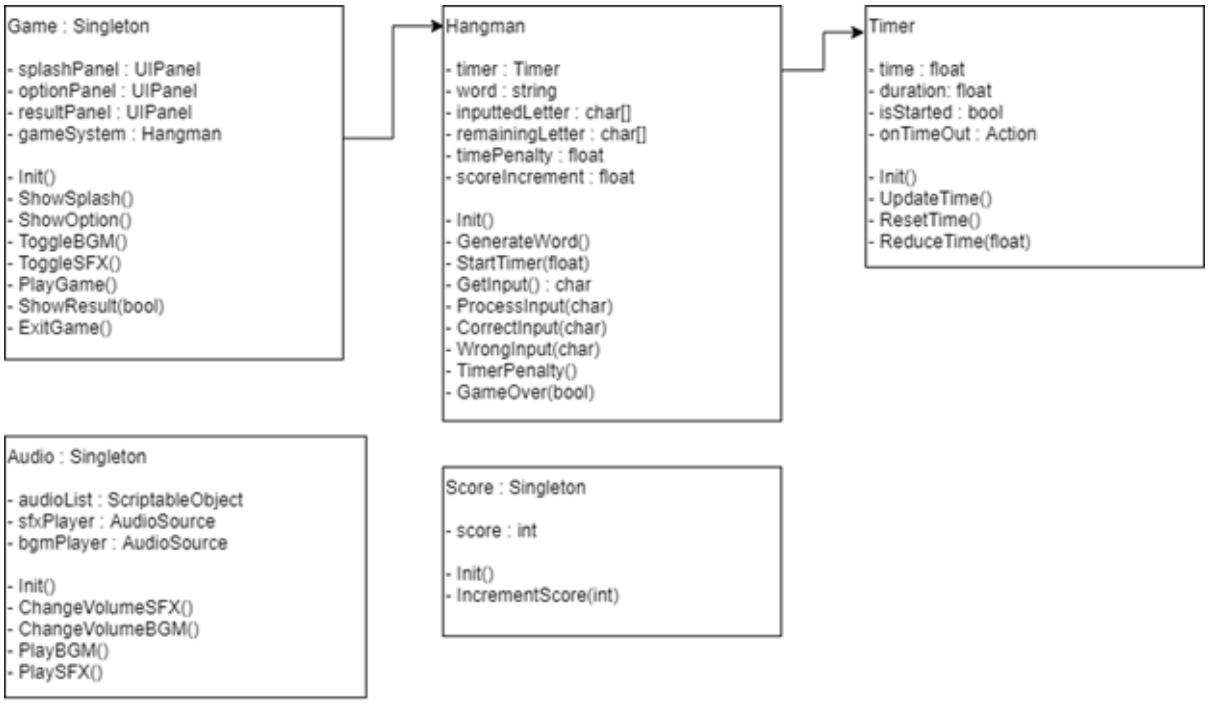


Pada saat kita buka game nya, kita akan diperlihatkan splash screen dengan judul game kita, setelah itu kita masuk ke main menu. Di main menu kita memiliki menu untuk membuka option, memulai game, atau keluar dari game. Kemudian pada option menu seperti pada wireframe tadi kita memiliki opsi untuk mengubah-ubah opsi suara, disini kita tidak akan deskripsikan secara detail tentang menu-menu pada option menu karena seharusnya cukup lumayan jelas.

Kemudian pada game nya ada baiknya kita buat flow nya lebih detail agar memudahkan kita untuk menetapkan flow utama pada game yang nantinya tidak boleh diubah dan bahkan mungkin tidak bisa diubah secara sistem. Flow game dari contoh wireframe tadi kita jabarkan dengan percabangannya juga. Pada game yang cukup besar kita mungkin tidak perlu menulis sangat detail seperti ini, tetapi menulis secara detail pada fitur masing-masing supaya lebih mudah ditulis.

Diagram UML

Kemudian kita juga memerlukan diagram untuk menjelaskan sistem secara langsung, yaitu diagram UML, pada diagram ini kita akan membuat hubungan antar class ataupun object yang akan dibuat. Selain untuk memudahkan pembuatan sistem, diagram ini juga dibuat untuk melimitasi kemampuan class dan memutuskan apakah class tersebut perlu dipecah atau tidak jika terlalu banyak fitur yang dia miliki. Sebagai contoh dari notes kita saat diskusi tadi, kita bisa membuat wireframe seperti berikut.



Pada game ini kita akan memiliki 1 class yang akan mengontrol flow nya menu-menu yang ada karena tidak terlalu rumit yang nantinya bisa kita pecah jika memang perlu, kemudian kita juga memiliki satu class yang akan mengatur audio, kemudian ada satu class yang akan mengatur poin yang kita dapatkan saat kita memenangkan game. Lalu pada game hangmannya sendiri kita memiliki dua class, satu untuk mengatur jalannya game, dan satu lagi untuk mengatur waktu game nya.

Terlihat pada diagram UML tadi kita memiliki 3 class Singleton. Singleton adalah class yang bisa diambil oleh class lain tanpa harus memiliki koneksi, jadi pada dasarnya semua class bisa mengakses class tersebut. Hal ini adalah salah satu alasan yang membuat penggunaan diagram UML jarang digunakan untuk keseluruhan game. Akan tetapi UML tetap bisa digunakan untuk menjelaskan class-class yang memang digunakan pada satu

fitur yang kompleks, misalnya pada sistem game hangman pada game ini yang membutuhkan class Time. Walaupun hubungan antar classnya tidak bisa dijelaskan dengan tepat, fungsi dan variabel yang kita tulis pada diagram UML ini akan cukup membantu kita pada proses pengerjaan game.

3. Tahap 3 - Prototyping

Pembuatan prototype akan membantu kalian untuk memulai pembuatan game. Pada dasarnya prototype sebuah game tidak perlu game jadi, contohnya pada game Tap Titan, prototype nya akan dimulai dengan sebuah game dimana user akan men tap layar, kemudian player akan mendapatkan skor. Proses ini memang jarang dilakukan pada pembuatan game kecil dan lebih akan digunakan untuk pembuatan game besar sebagai rangka atau gambaran besar game yang akan dibuat.

Untuk pembuatan prototype ini biasanya dilakukan agar memudahkan kita untuk membuat diagram-diagram di atas, juga karena mungkin ada hal yang belum terpikirkan atau ada hal-hal yang memang terjadi karena limitasi dari engine atau bahasa yang kita pakai dan baru bisa diketahui saat kita membuat prototype ini. Proses ini biasa dilakukan saat kita baru pertama kali membuat game tersebut, atau kita sedang melakukan riset suatu game atau fitur tertentu.

Proses prototype bisa langsung dilakukan dengan game engine yang akan dipakai seperti Unity, Unreal Engine, dll. Untuk assetnya, kita bisa menggunakan asset dummy saja, atau kita ambil dari asset store, atau bisa juga meminta bantuan artist untuk membuatkan asset dummy nya, seperti karakter orang kotak-kotak dengan animasi seadanya untuk membuat prototype pergerakan karakter.

Setelah membuat prototype, biasanya kalian bisa langsung mendiskusikannya dengan tim terkait keberhasilan atau kegagalannya, juga kesulitan-kesulitan serta limitasi yang muncul pada prototype ini. Setelah itu biasanya tim akan mulai memakai prototype ini sebagai basis dari game yang akan dibuat. Saatnya membuat game yang hebat!!

4. Apa itu Technical Design Document?

TDD merupakan dokumentasi dari spesifikasi/*blueprint* dari produk/*software* yang akan dibuat. Dokumen ini mendeskripsikan solusi dari masalah-masalah teknis yang ada dalam project, dan memberikan arahan kepada tim developer mengenai arsitektur *software* yang akan dibuat. Isi dari TDD dapat mencakup keseluruhan project atau dibagi per modul besar dalam project, tergantung kesepakatan pihak yang terkait. Dalam game development, umumnya TDD mencakup sebuah game yang akan dikerjakan.

Dokumen ini akan menjadi dasar dari program yang akan dibuat oleh tim, karena merumuskan apa saja yang akan dibuat serta bagaimana cara mengimplementasikannya, dan juga standarisasi *environment* yang akan digunakan selama proses development.

- **Standarisasi tools dan teknologi yang digunakan**
  Dalam TDD perlu dituliskan tools yang akan digunakan sebagai kesepakatan untuk seluruh tim. Misalkan menggunakan Unity versi tertentu karena ada beberapa fitur yang tidak terdapat dalam versi lainnya, agar tim dapat bekerja dengan lancar diperlukan kesepakatan untuk menggunakan versi yang tercantum dalam dokumen.

- **Coding style**
  Bagi para programmer juga diperlukan untuk menyamakan cara menulis code program. Deskripsi lengkap untuk standarisasi coding style dituliskan dalam dokumen terpisah, akan tetapi dalam TDD sebaiknya dituliskan file yang merujuk pada dokumen tersebut. Misalkan, produk yang dibuat merupakan permintaan client, dan pada perusahaan client tersebut mereka memiliki standarisasi coding style sendiri, maka dalam TDD perlu ditambahkan *coding style* yang digunakan.

- **Process standard**
  Proses-proses yang perlu dilakukan oleh developer semasa development, misalkan jika pada produk dibutuhkan game analytics, developer perlu menuliskan proses yang dibutuhkan untuk dapat menggunakan analytics tersebut.

5. Apa Fungsi Technical Design Document?

Technical Design Document (TDD) dalam game development memiliki fungsi untuk:

- **Alat komunikasi**
  Bagi tim, dengan adanya TDD, maka akan lebih mudah untuk mengetahui *scope*/lingkup project yang akan dikerjakan, serta prosedur-prosedur yang digunakan untuk mengerjakan project. TDD juga mempermudah anggota baru untuk memahami project.

- **Standarisasi**
  Seperti dijelaskan pada bagian sebelumnya, TDD berisi standar dari *environment* pada saat development project. Standar ini merupakan hal yang sudah disepakati oleh seluruh anggota tim (dan client, bila ada).

- **Game Flow**
  TDD mendeskripsikan flow dari game yang akan dibuat sesuai dengan spesifikasi yang dibutuhkan. Ini membantu developer sebagai acuan ketika mengerjakan project.

- **Organizing Solutions and Ideas**
  Seringkali bagi developer untuk menuliskan TDD setelah proses development selesai, padahal hal terpenting dari TDD adalah untuk menggali segala solusi yang terdapat dan mengolah serta mendisukusikan solusi yang ada terlebih dahulu dengan tim sebelum memulai proses development, agar tidak membuang-buang waktu ketika masa development nantinya. Bayangkan jika developer langsung membuat/men-develop program tanpa mendesain produk terlebih dahulu. Ketika developer sudah mengimplementasikan suatu metode, kemudian pada saat melanjutkan development metode tersebut ternyata tidak dapat digunakan untuk masalah lain, maka developer harus merombak ulang pekerjaan yang sudah dikerjakan sebelumnya dan menggantinya lagi. Hal ini sering terjadi karena tidak ada persiapan sebelum melakukan development. Oleh karena itu dalam TDD perlu dirumuskan masalah-masalah yang akan dihadapi beserta solusi-solusi yang dapat digunakan. Kemudian solusi-solusi tersebut didiskusikan dengan tim untuk memperoleh solusi terbaik.

- **Tracking**
  Dokumentasi TDD tidak berhenti pada saat proses development dimulai. Dalam tahap development, sangat mungkin terjadi perubahan dalam spesifikasi project, sehingga dapat terjadi perubahan dalam TDD juga.

6. Isi TDD

1. Content version history

Bagian ini merupakan bagian yang umum terdapat dalam design document untuk melihat perubahan apa saja yang terjadi dalam dokumentasi.
Contoh:

Content version history

Time	Ver	Author	Changes
09-09-2019	0.0	Agus	Initial Document
30-09-2019	0.1	Budi	Penambahan fitur: entity baru dengan skill akselerasi

2. Table of content

Daftar isi dokumen. Setiap bagian diharapkan memiliki hyperlink ke bagian yang dituju.
Contoh:

Table of Content

3. Introduction

Pendahuluan yang berisi deskripsi singkat mengenai produk yang akan dibuat, beserta tujuan serta lingkup/*scope* dari produk.
Contoh:

1 Introduction

Angry Birds merupakan sebuah casual puzzle game di mana *player* diharuskan untuk menembakkan burung yang ada dan menghancurkan *base* lawan. Game ini akan mengsimulasikan ilmu fisika (*physics*) di dalamnya, serta mengimplementasikan konsep *inheritance* dalam Object-Oriented Programming (OOP) .

1.1Purpose & Objective

Angry Birds ditargetkan untuk pemain casual, terutama penggemar game *puzzle* agar dapat dimainkan di sela-sela aktifitas para pemain.
Target utama dalam game ini adalah untuk menghancurkan semua musuh pada setiap level. Pemain akan menggunakan ketapel yang ada untuk melemparkan sejumlah burung ke arah *base* lawan untuk langsung mengenai target atau *obstacles* yang ada agar *obstacles* tersebut dapat menimpa lawan.

1.2Project Scope

Pada game ini akan diimplementasikan fitur-fitur berikut:

- *Gravity* pada objek-objek dalam game
- Melempar burung dari ketapel
- Menghancurkan musuh yang terkena burung
- Menggambarkan *trajectory* pada saat akan melempar burung
- Menggambarkan gerakan burung pada saat dilempar

4. Technical Overview

Deskripsi *requirement* teknis yang diperlukan pada proses development.
Contoh:

2 Technical Overview

2.1 Target System Requirements

Platform/OS: Android 4.1 API level 16

RAM: 512MB

2.2 Tools Used

Users	Tool	Use
Prog	Java SDK	**jdk_v1.8.0 and jre_v1.8.0**
Prog	*Unity*	**Ver 2018.3.6f1**

2.3 Engines & Middleware

Deskripsi mengenai engine yang digunakan dan middleware bila perlu

Game ini akan dibuat dengan menggunakan Unity2D game engine. Physics dalam game pun akan menggunakan fitur *Physics2D* dari Unity untuk mempermudah pengerjaan.

2.4 File Format

Assets	.png	Image used in game
Code script	.cs	Game script

2.5 Technical Code

2.5.1 Technical specification

Platform:	Android Mobile
Programming language:	C#

5. Technical Feature

Fitur-fitur yang dibuat dalam project perlu dijelaskan pada bagian ini. Selain penjelasan terhadap fitur, perlu juga dijelaskan *technical strategy* apa saja yang dapat digunakan untuk menyelesaikan fitur yang ada. Dari kumpulan *technical strategies* tersebut, kemudian jelaskan teknik yang dipilih dan alasannya.

Contoh:

3 Technical Feature

3.1 Gravity pada objek-objek dalam game

Penjelasan fitur dan problem yang dihadapi, dan solusinya

Pada game, objek-objek seperti burung, lawan dan obstacle akan terpengaruh gaya gravitasi, sehingga akan selalu jatuh ke bawah, ketika dimainkan.

3.1.1 Implementation 1 – Physics 2D Unity

Technical strategy untuk menyelesaikan fitur ini

Setiap game object akan diberikan komponen *Rigidbody2D*. Komponen ini akan mensimulasikan *Physics2d* pada Unity, termasuk dengan gaya gravitasi.

3.1.2 Solution 2

Technical strategy lainnya yang dapat digunakan untuk menyelesaikan fitur yang ada

Ex.
Membuat script yang akan mensimulasikan gravitasi dengan mengimplementasikan rumus fisika ke dalam script dan menambahkan script ke setiap game object.

3.1.3 Selected Implementation

Solusi yang dipilih dan alasannya

Physics2D dari Unity akan digunakan untuk mensimulasikan gravitasi pada setiap objek game yang membutuhkan. Dengan menggunakan *Rigidbody2D*, selain gravitasi, kita juga dapat mengatur komponen *physics* lainnya dari sebuah objek, misalnya daya gesek objek ketika bersentuhan dengan objek lain.

3.2 Trail

Pada game ini, kita akan menggambarkan trail dari burung saat dilemparkan.

3.2.1 Implementation 1 – Instantiate image object

Image dari trail akan diinisiasi dengan menginisiasi objek baru yang memiliki komponen image dari *trail* setiap *x* seconds.

3.2.2 Implementation 2 – Trail renderer

Menambahkan komponen TrailRenderer pada burung yang diterbangkan agar menampilkan *trail*.

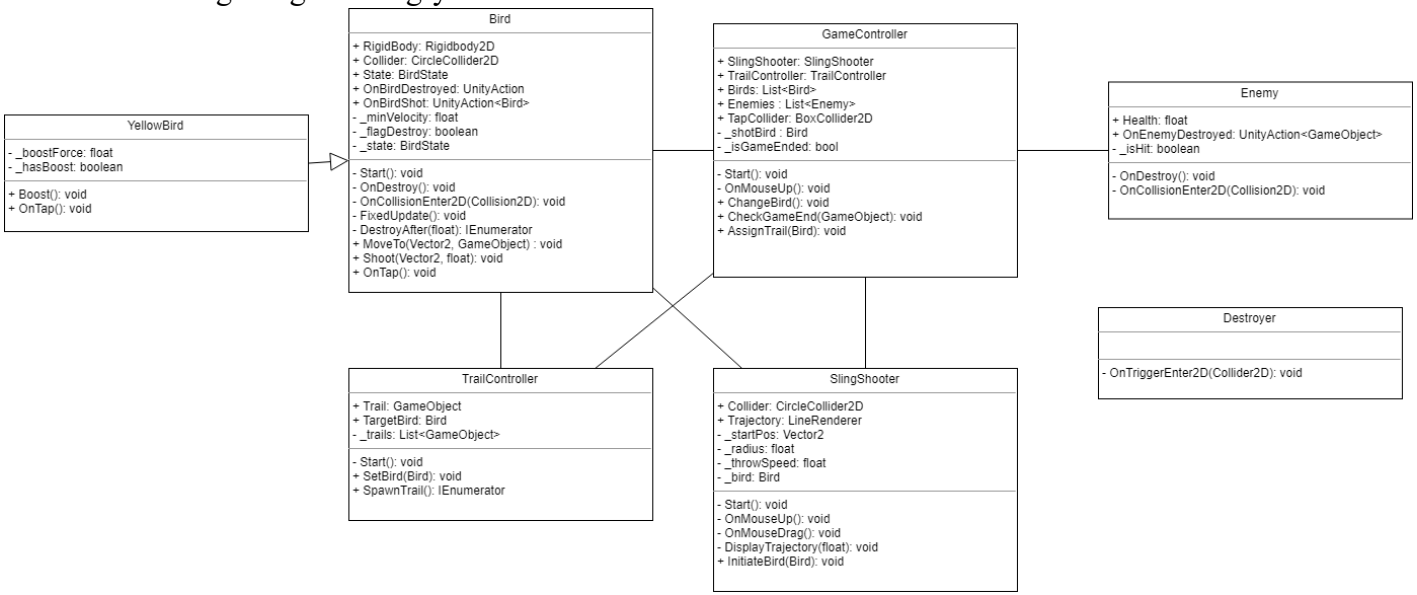
3.2.3 Selected Implementation

Inisiasi *gameobject* dengan image trail akan digunakan untuk menggambarkan trail dari burung. Hal ini dikarenakan trail yang diinginkan bersifat statis (tidak terus bergerak mengikuti burung). Selain itu ukuran trail yang dihasilkan oleh TrailRenderer lebih sulit untuk diubah sesuai dengan ukuran yang diinginkan.

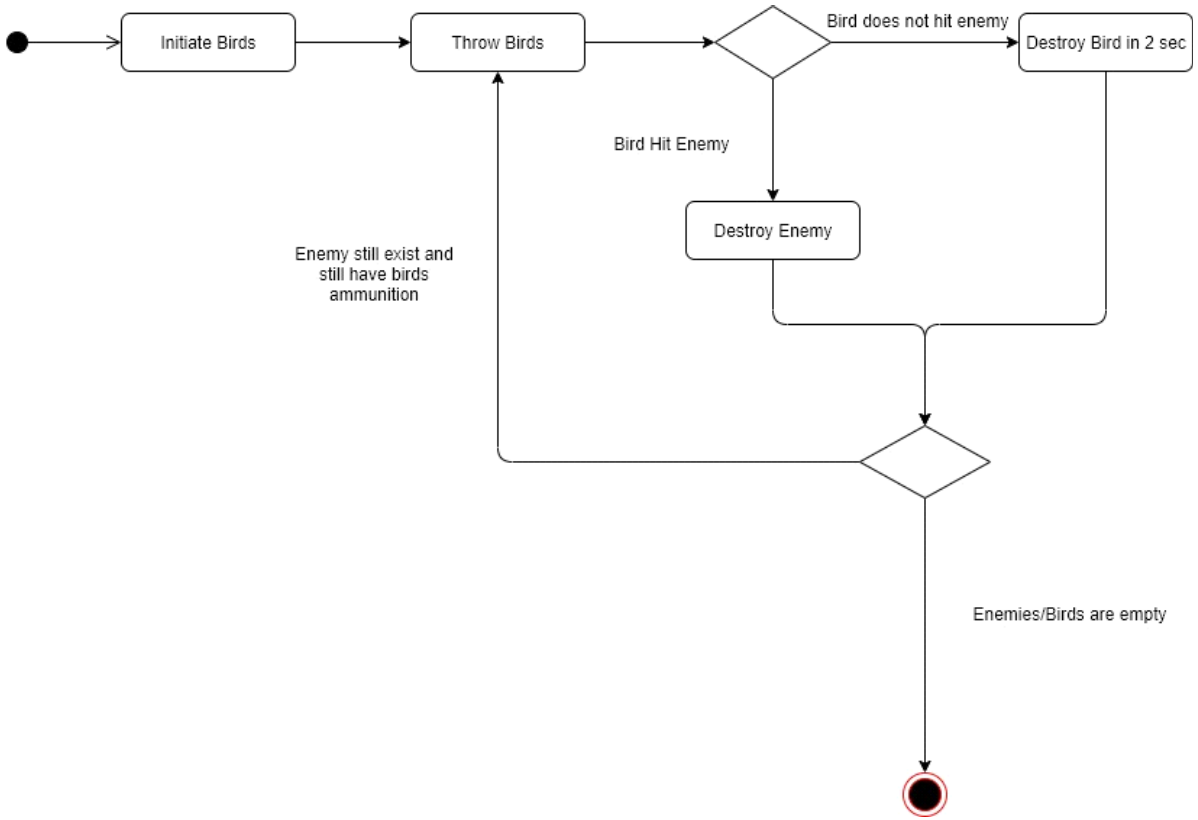
6. Technical Design

Bagian ini berisi dengan diagram-diagram UML (Unified Modeling Language).
Diagram-diagram tersebut dapat berupa class diagram, activity diagram, database diagram, dan diagram-diagram lainnya yang dapat menjelaskan struktur/flow dari project yang dikerjakan.

Class diagram merupakan tipe diagram yang paling umum digunakan untuk menggambarkan struktur dari sistem yang terdiri dari kumpulan *class*, beserta dengan *attribute* dan *method* yang dimilikinya, serta relasi antar *class*.
Contoh class diagram game Angry Birds:



Untuk dapat menjelaskan flow dari program dapat menggunakan activity diagram yang merupakan salah satu tipe diagram UML untuk menggambarkan *workflow* dari suatu aktifitas (*activity*) atau *actions*.
Contoh activity diagram Angry Birds:



7. Administrative Data

Jika pada project diperlukan server atau backend, maka pada bagian ini deskripsikan konfigurasi yang digunakan.
Contoh kerangka table untuk mendeskripsikan administrative data:

5 Administrative Data

5.1Server

Item	Value	Note
Host IP		
Domain name		
Username		
Password		

5.2Database

Item	Value	Note
Host IP		
Domain name		
Server username		
Sserver password		
DB username		
DB password		

5.3Other

Item	Value	Note
Google store account		
Apple developer id		
Username		
Password		

8. Reusable Codes

Apabila dalam project yang dibuat terdapat modul-modul yang dapat digunakan kembali dalam project lain, maka perlu dijelaskan modul-modul apa saja yang *reusable* dan jelaskan cara penggunaannya, bila suatu waktu modul-modul itu akan digunakan dalam project lain.
Contoh:

6 Reusable Codes

Dokumentasikan seluruh reusable code dalam project bila ada.

6.1Object Pools module

6.1.1 Description

Modul ini digunakan untuk mengimplementasikan konsep *object pooling*. Misalkan object pooling untuk game object meteor pada *Space Shooter* tutorial Unity. Modul ini dapat menginisiasi objek sejumlah N (bergantung pada nilai yang dimasukkan oleh developer), kemudian menentukan apakah pools akan terus bertambah jika objek yang dibutuhkan melebihi jumlah objek yang terdapat dalam pools. Setelah objek selesai digunakan, maka objek akan dikembalikan ke dalam pools agar dapat digunakan untuk kebutuhan yang mendatang.

Modul ini terdiri dari sebuah class bernama ObjectPools.cs.

6.1.2 How To Use

Bila terdapat sebuah class yang memerlukan object pool, tambahkan class ObjetcPools sebagai attribute ke dalam class tersebut. Kemudian panggil method GetObject dari class ObjectPools untuk mendapatkan GameObject yang bersangkutan. Ketika GameObject selesai digunakan, disable game object tersebut lalu panggil method ReturnObject dari class ObjectPools.

9. Other Notes

Bagian ini dapat digunakan untuk memberi catatan tambahan untuk project yang dikerjakan, apabila ada bagian yang tidak terdapat dalam point-point di atas.

7. Referensi Pembuatan Diagram

Dalam pemrograman, dikenal sebuah bahasa khusus yang dipergunakan untuk menggambarkan dan mendokumentasikan artifak perangkat lunak. Bahasa khusus tersebut dinamakan dengan UML (Unified Modelling Language).

Di contoh TDD sebelumnya, terdapat dua jenis diagram yang digunakan, yaitu class diagram dan activity diagram. Bagi yang belum familiar dengan konsep ini, berikut ini link untuk mempelajari lebih lanjut tentang class diagram dan activity diagram:

1. Class diagram: https://www.tutorialspoint.com/uml/uml_class_diagram.htm
2. Activity diagram: https://www.tutorialspoint.com/uml/uml_activity_diagram.htm