**Creating a Plan Of Approach**

# Creating my Plan Of Approach

So now I have to think about what I like to have in my game. There are some elements I like to see in my game. I wrote them down as bullet points:

* Time-Interval
  + Real-Time only
* Genre
  + Action
* Platform
  + Xbox 360
  + Windows-PC
* Goal
  + Entertainment
* Gameplay
  + The game is based on the multiplayer modus, arena style.
    - 4 Players Max
    - NPC’s Available (Non-Player Characters/Artificial Intelligence)
  + The players car is fully customizable.
  + The players car is bound to the players gamertag(Xbox Only)
  + The players car is bound to the players custom save-game (PC Only)
  + There is a currency system, to buy upgrades and customization items.
  + Upgrades:
    - Defensive Upgrades
    - Offensive Upgrades (Both range and direct)
    - Movement Upgrades
    - Looks Upgrades
  + The game has a levelling system, to unlock new arena’s

So I like to create this game as a Multiplayer-Game where you can battle up to 3 other players or computerized players. To create a feeling of achievement and to be able to set personal goals, thus making the game addictive, I will add a currency system where you can earn money for example based on amount of damage dealt, damage done to you and time. An upgrade system will be placed in the game as well, where the money gained can be used. The game will be in Real-Time interval, where both players play simultaneously in an attempt to defeat their contestant. To make the game more personal I like to give each player his own car, which he can bind with and what will create a feeling of ownership.

# Programming & Graphics tools

So I got my Ideas in place, but now I have to actually create a game. I have to use a programme to program my game in, test my game in, and design my game in(Graphically). As I said before I already have experience in creating games, both programming them and designing them.

I always used, and will use, the windows-language C#. There are different levels of programming levels, first you have ASSEMBLY, which is the language used to talk directly to the hardware of the system. After that you have the 2nd programming language level such as C, a language created to make it easier for humans to program in. I use a 3rd level of language named C#. You can program with this language in Notepad, a standard tool in Windows, but in this you can only type, not test or look up what mistakes you have made. To make this job easier to do, Windows has developed a program to develop programmes and debug them. This program is named “*Microsoft Visual C# 2010 Express”*. This is a free tool released by Windows.

But with this tool it is still hard do develop game effectively, since there is no graphical engine, and no methods (groups of code) to draw graphics to the screen. Another team of Windows has created the “*XNA Framework*”, with their most recent version XNA 4.0. This gives multiple classes to draw basic things to the screen and connect online and with storage devices. This is very helpful, but very basic as well. There is no code to check collisions in an effective way for example, which I will develop myself later in this paper. This framework is used for graphical and storage use only, the actual ‘Game’ still has to be developed fully by the user.

To draw things to the players screen, the game needs graphical images first, which in turn need to be created first. The program I use for this is “*Paint.NET*”, a free program developed by a small company. This tool is similar to the Windows program Paint, but with much more extra’s. I use a simple program like this since I can use it and create what I need. Because I myself am not a very creative person with skill to draw, I don’t need any sophisticated programmes like Photoshop.

# Creating Graphics

First I’d like to complete most of the graphics first, and further graphics (such as offensive/defensive items + animations) later. I have to create multiple arena’s for different levels and I want about 10 arenas in my game. I have to draw a basic car as well which is coloured white, so I can overlay it with colours later. There also has to be a menu, and that is what I will create primary to the main game. The rest I will developed mixed with the process of programming.

# Programming Steps

In this subsection I will globally sum up the steps I want to take to achieve certain goals. This section will not include every step, because I know that when I am busy programming, I will bump into thing I had not yet foreseen.

Step 0:

I have to import my limited code already generated for previous projects.. This will save much time and nothing very important is included. I will include my “Sprite” class only, which is a group of code that I myself wrote that handles graphics, so that the graphics file is immediately ready to be drawn to the screen.

Step 1:

I have to program the screen-engine. This engine must only update and draw what is needed in the window of the player at a particular moment. I need different states in the form of: Main Menu, Load Menu, Save Menu, Playing, Playing.Pause(State in a state), Options Menu. This way I can assign different buttons and images to different screen, and the Screen manager can make the program only draw what is necessary. This way I will spare CPU. This Manager will become clearer later on.

Step 2:

I have to create the menu’s and buttons. No engine for playing is required yet. I want to make the menu to a state where it is almost finished, so I don’t have to edit this anymore. The button’s won’t get any actions assigned yet, except for the ones carrying the player to a different screen. The Screen Manager is already ready, so I can control this.

Step 3:

In this step I will make a controllable car. This car can rotate to the angle the driver is controlling it. I have to make the driving look natural. I will create a trial of smoke as well, just to make it look nice. Another graphical addition the car will have is a trail when it brakes.

Step 4:

I have to make the car customizable, through colouring. I will create an upgrade menu as well, but I will think later of the upgrades that it should contain, so I will leave this screen unfinished. The car class will need room for later code for upgrades as well.

Step 5:

I will make the game compatible for up to 4 players simultaneously, where when pressing the play button, each player should sign in manually with different controllers for setting the different players. There should also be an option to add NPC’s, in different difficulties. The AI department will not be created yet, first I want to get the game finished for real players only, this is due to the difficulty of reproducing human reactions. I should also create code where 4 players can drive in the arena at the same time.

Step 6:

I have to write collision code for 4 cars at the same time. This will proof a difficult challenge, mathematically and in the aspect of bugs. This will also include giving momentum to a car that gets hit, without changing the direction of the nose.

Step 7:

Here I will create code to determine the damage by the player that drove into another player. This depends on the hardness of the car’s outside and angle for example, by both of the players. I will have to create an algorithm which keeps in mind all the variables this depends on.

Step 8:

I have to create any option of respawning and repair I will put into the game, determined by global standards and personal upgrades.

Step 9:

Now it is time for the code of offensive items, with the attacks further in the development decided. Damage has to be set for each particular item, and effect per item. The upgrades have to be decided upon as well, which will be imported into the game.

Step 10:

Code for a finished game has to be created, as well a screen (popup?) for when each player earns his money. After this players will be asked for a rematch or a return to main menu. This will close the circle of never-ending gaming experience.

Step 11:

I will have to create the AI engine for computer-controlled cars, for this I have to do some personal research, but the book I read at the moment covers this subject. Difficulty levels have to be imported as well, but I think this is a matter of precision of the ‘virtual’ player by just one variable in percentages (such as: easy 0.3, medium 0.5, hard 0.7, near impossible 0.9).’

Step 12 & 13:

Create codes for saving personal statistics and scores and finalizing the product by removing bugs

Step 14:

After play testing remove the bugs found by the testers.

# Upgrades & Offensive Items

I will cover this subject later, because I now like to start drawing and programming, only when I have finished until Step 8 I start thinking about this, because then I have an idea on what I can and cannot do, with my resources, knowledge and time.

**Comparison**