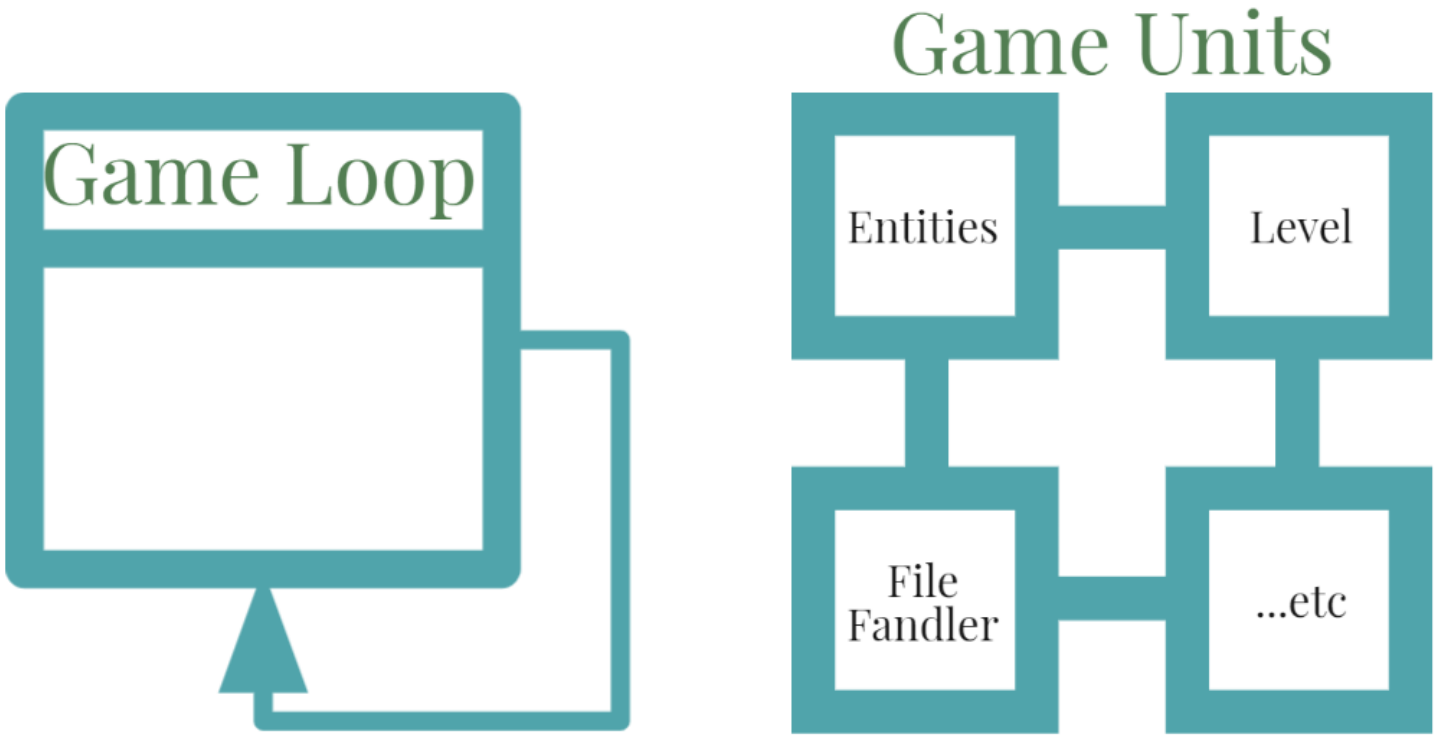
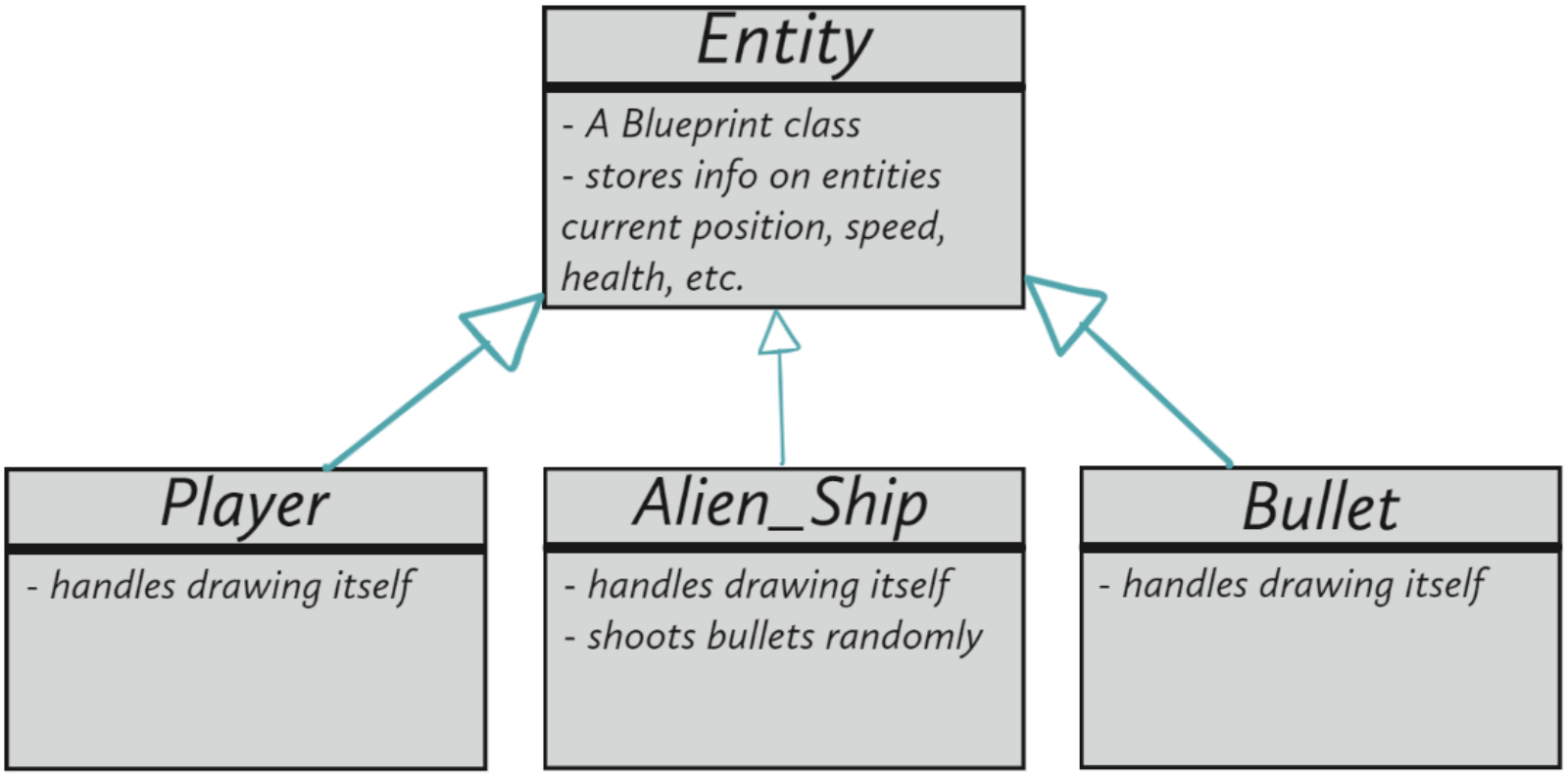

Space Invaders
Game

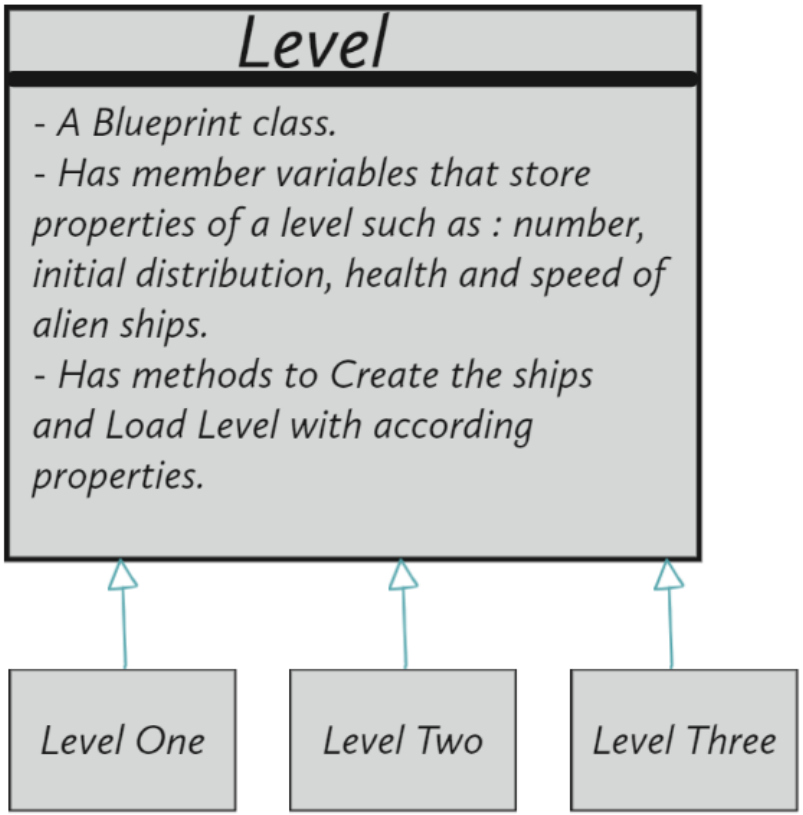
Games Usually have 2 main parts:



Game Units:


Entity
- A Blueprint class
- stores info on entities current position, speed, health, etc.
Player
- handles drawing itself
Alien_Ship
- handles drawing itself
- shoots bullets randomly
Bullet
- handles drawing itself

Game Units:



It's extendable, you just add new Level that inherits from Level and add whatever properties you want.

Game Units:

Ship Move Handler

- handles automatic movement of the alien ships

Collision Handler

- handles collisions between:
1- Player's bullet and aliens.
2- Aliens bullets and Player.
3- Player's ship and Alien's ship.

Sprites Handler

- Loads needed sprites whenever needed .

Game Units:

Credits

- Runs credits at the end of the game.

Score Handler

- Constantly calculates and updates player score.

Error Handler

- checks if all gfx files are present, and if not it doesnt run the game, instead it display a warning message.

File Handler

- Writes Player name and score in a .txt file.

Game Loop:

The Game loop has ONE main responsibility :
Handling user Input.
This means that it detects when the user presses a button, and accordingly change the game state.
For example, from “intro” to “gamePlay” to “pause” to “won” to “quit", etc.
and while in the “gamePlay” state, it moves the player’s ship and fires bullets according to his input.
The Game Loop "USES” all game units while running

External Tools:

I am using a single-file prototyping and game-engine framework created in C++ which is called The olcPixelGameEngine .

What does it do?

olcPixelGameEngine allows you to rapidly develop prototypes and games. It does this by creating a window, and rapidly drawing to that window. It is sensitive to keyboard and mouse input. The Screen is considered to be an 2D array of Pixels. Pixels have a defined width and height in real screen-pixels. Basic drawing tools for manipulating the Screen are provided.

By design, the olcPixelGameEngine requires no "boilerplate" effort from the user, i.e. the user can just focus on getting on with creating the fun parts of the application.

What doesn't it do?

olcPixelGameEngine does not provide any implementation of typical game resources. For example, it does not provide tools to handle asset loading, collision detection, vector mathematics. As it is an educational tool, it is expected the user will provide this functionality.

Source: https://github.com/OneLoneCoder/olcPixelGameEngine/wiki