Object Oriented Software Design
— Object Oriented Software Design

B99902006 王柏軒 B99902012 陳立展 B99902075 賴君濠

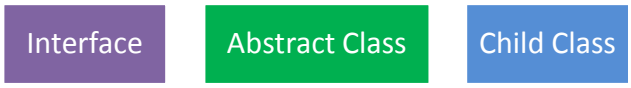
I. Responsibility division

王柏軒：architecture, map, itemlist, menu, fadeinout, gravity...

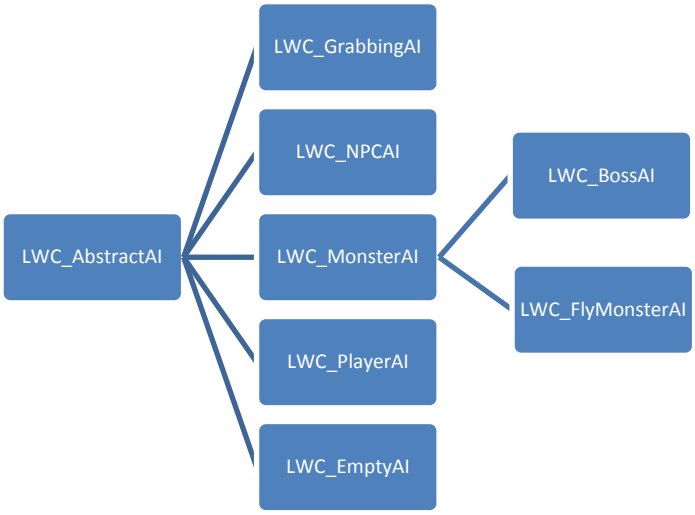
陳立展：NPC, FlyMonster, opencinema

賴君濠：Monster, Boss, their attack skills

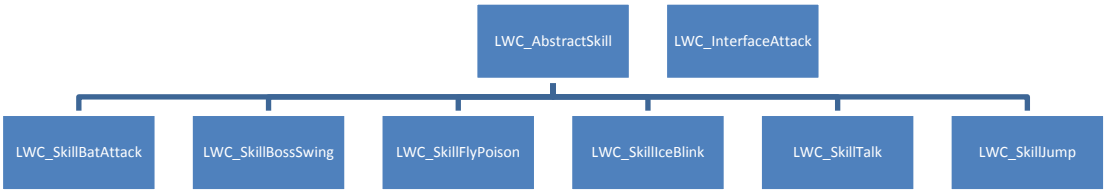
II. The relations between the classes



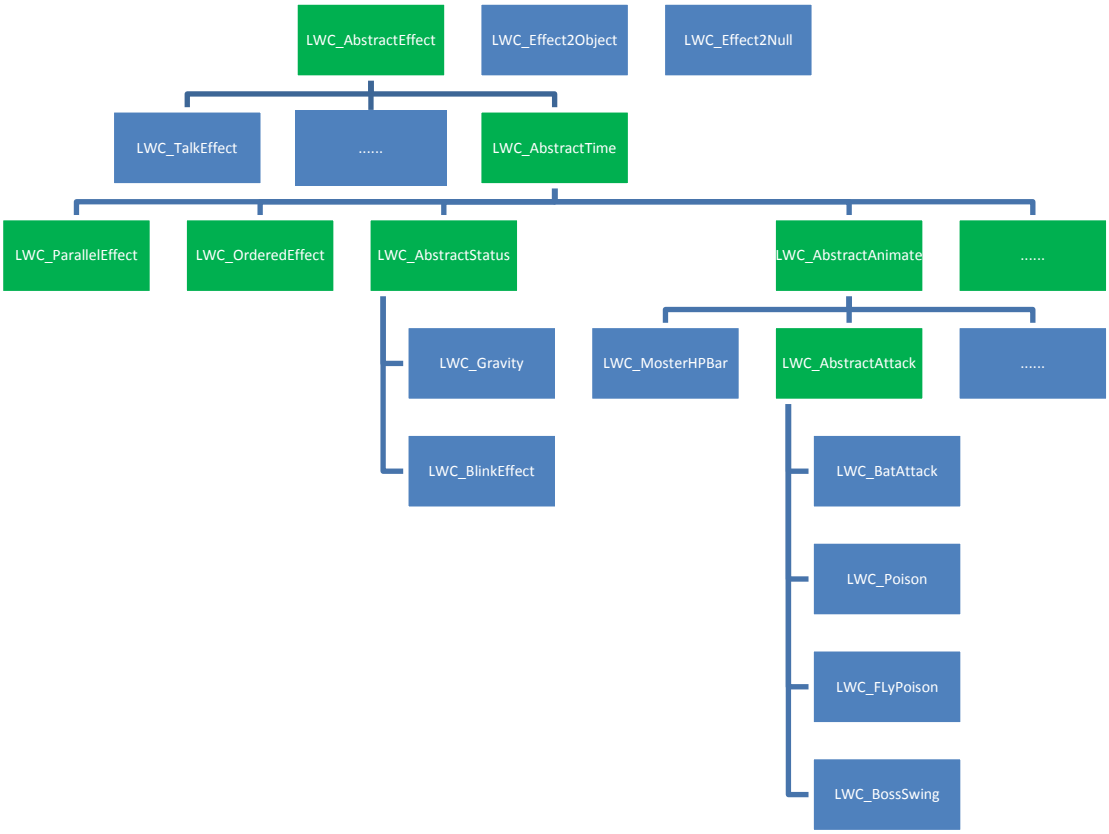
AI



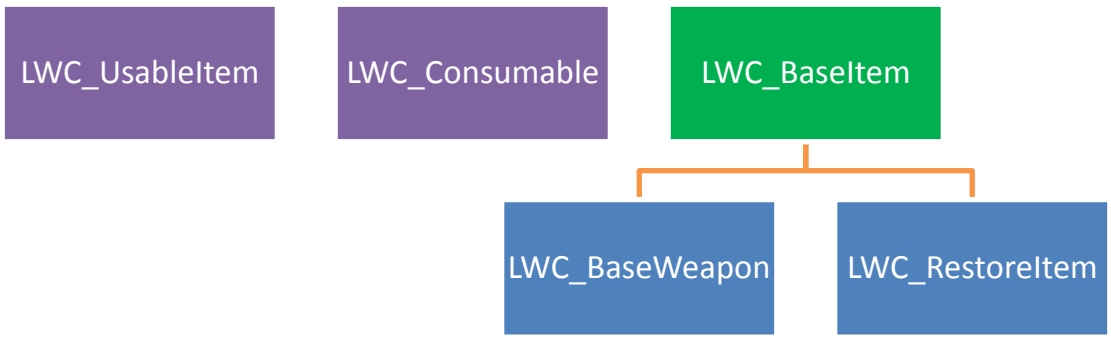
Skill



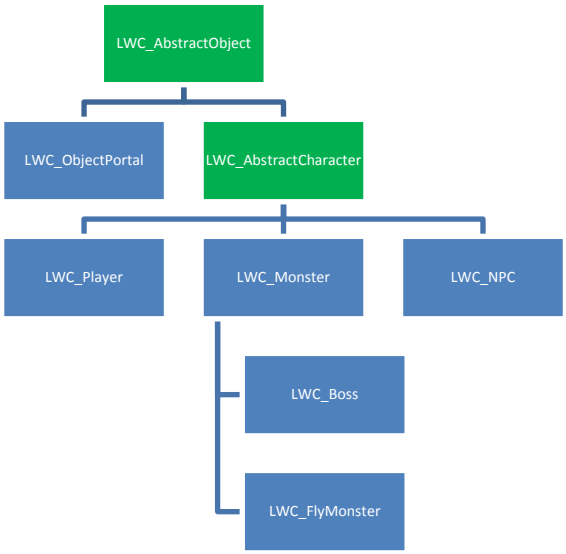
Effect


LWC_AbstractEffect
LWC_Effect2Object
LWC_Effect2Null
LWC_TalkEffect
.....
LWC_AbstractTime
LWC_ParallelEffect
LWC_OrderedEffect
LWC_AbstractStatus
LWC_AbstractAnimate
.....
LWC_Gravity
LWC_MosterHPBar
LWC_AbstractAttack
.....
LWC_BlinkEffect
LWC_BatAttack
LWC_Poison
LWC_FLyPoison
LWC_BossSwing

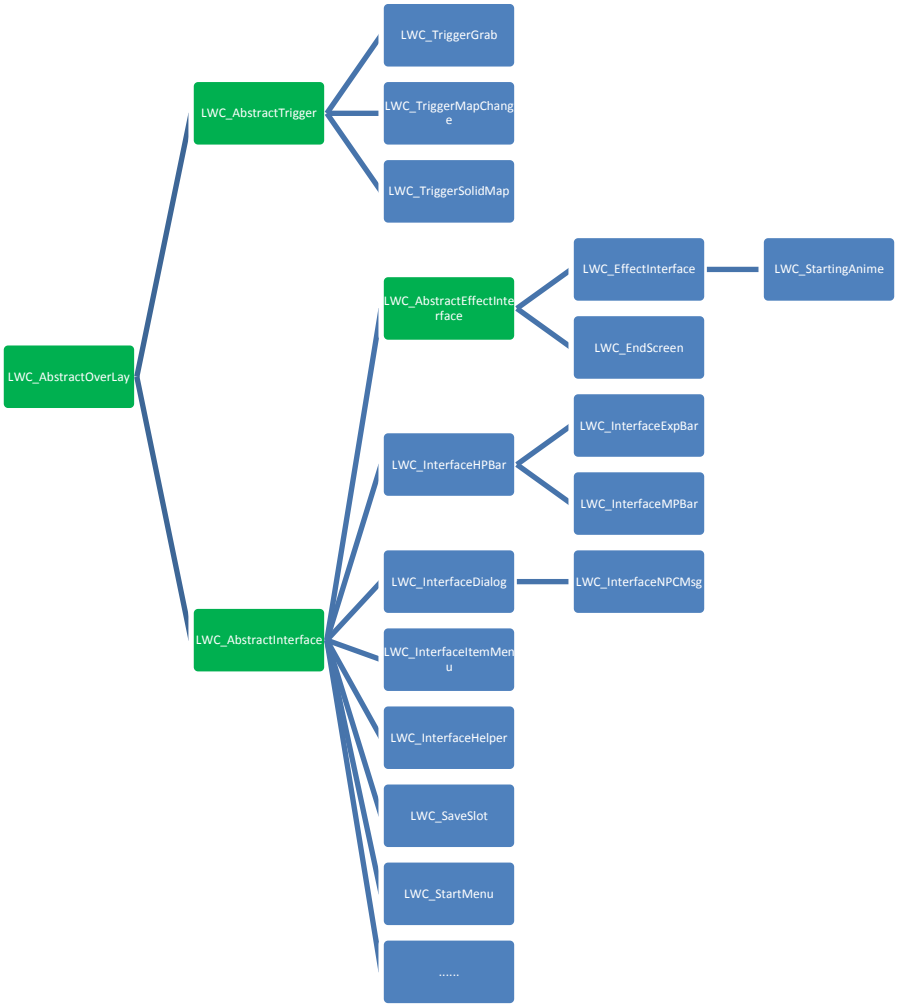
Item


LWC_UsableItem
LWC_Consumable
LWC_BaseItem
LWC_BaseWeapon
LWC_RestoreItem

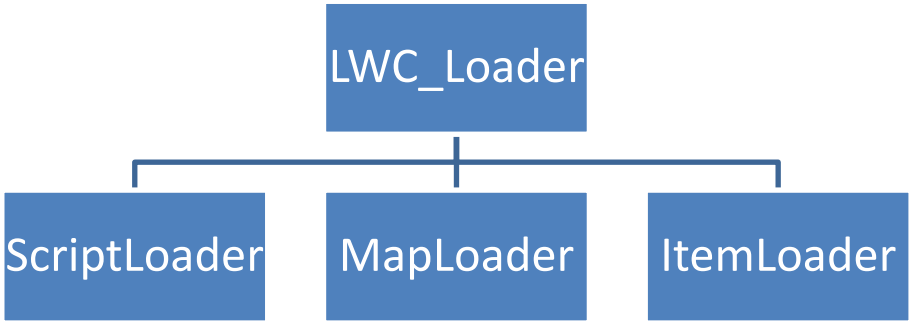
Object



Overlay



Loader



Has Relationship

Object

LWC_AbstractObject

- Has HashMap Register for saving the variables such as HP, MP and states.
- Has HashMap counter for saving the variables relating to time such CD and counter for actions.

LWC_AbstractCharacter

- Has LWC_AbstractAI, which controls the actions of the character including how to move and attack.
- Has LWC_AbstractWeapon, which enhance the power of the character

LWC_Player

- Has ArrayList itemlist, saving the items that the players now.

AI

LWC_AbstractAI

- Has the obj, that use this AI.

LWC_PlayerAI

- Has AttackList, which store the attack skill
- If the skill is instance of LWC_InterfaceAttack, and then we would add it into our attacklist.

LWC_MonsterAI

- Has detect_radius, that can use to detect player
- Has skill, that can use that skill, such as Poison skill

Effect

LWC_AbstractEffect

- The lowest level Effect
- Has a secret to distinguish which effect

LWC_AbstractTime

- Has a counter and a duration
- Because we need to use counter to calculate time effect

LWC_AbstractAnimate

- Has an image array that can show Animate

LWC_AbstractAttack

- Has a LWC_AbstractCharacter obj, that can know the user

LWC_AbstractStatus

- Has a LWC_AbstractObject owner to record bacause this effect only has one user

Loader

LWC_Loader

- Has an JSON object that help to read ".txt" file in json format

Item

LWC_BaseItem

- The lowest level Item
- Has a item name, description, and img

LWC_BaseWeapon

- Has a weapon damage

LWC_RestoreItem

- Has a retoreMP and restoreHP that record this item can restore how many

LWC_UsableItem

- This is a interface that can use this item but not disappeared

LWC_ConsumableItem

- This is a interface that can use this item but would disappear

Skill:

LWC_AbstracSkill

- If the skill's CD is cool down and this skill's requirement is fulfilled, it can be trigger.

OverLay:

LWC_AbstractOverlay

- The lowest abstract class
- Has a RPG object

LWC_AbstractTrigger

- Has an image that records trigger points
- This is responsible for trigger event

LWC_AbstractInterface

- Has a counter for updating the interface

LWC_AbstractEffectInterface

- Has an effectlist that record time effect
- If the effect belongs to time effect, it would be add into the effectlist, and then wait to be update
- If not, make this effect act

III. The advantages of your design

We divided our game into three layer – LWC_Effect, LWC_Object, LWC_Overlay. The LWC_Effect is handling the underlying game stat change and controls LWC_Object in some way for animation or certain physical condition.

In the other hand, LWC_Object represents the real thing on the screen, like the animation, player, monster, etc. The LWC_Object is controlled either by LWC_Effect such as LWC_EffectAnimate or by LWC_AI for LWC_Character. Frankly saying, LWC_Object are designed only for showing, though it holds a method 'update'. And for LWC_Character which extends LWC_Object, the only thing it should concern about is how to show its images by its current state. For the AI held by LWC_Character, its main job is to deal with the moving strategy and action strategy, should it use a skill, where should it go, etc. For LWC_Object, it only needs to concern about where it location is but where the viewer actually sees, which provides an abstract environment on the game world.

As the "overlay" words meaning, LWC_Overlay mainly provides the Interface on

top of all the other things on screen or points the trigger point on map. It have a two main child, LWC_Interface and LWC_Trigger. LWC_Trigger provides the ability to choose what should to do when object enter certain area and what should do if none of them enter. LWC_Interface provides a canvas with the same axis as JFrame instead of the game background axis as LWC_Object. The Interface is mainly designed for the startmenu, hpbar, mpbar, expbar, itemlist, and which provides images on top of all the things.

And for the main manipulating class LWC_RPG. It extends the Jpanel and overwrite the paintComponent method for showing

For the other things, LWC_Loader is mainly designed to load and create new instance dynamically by human readable “txt”. We designed a MapLoader, ItemLoader, ScriptLoader, and MissionLoader though it is written as eventRecorder. LWC_Skill contains LWC_effect and usage requirement. Besides, LWC_Skill can set the skill’s CD time easily and gracefully by simply call a method. LWC_ImagePool loads the img and save them in register for avoiding duplicate load. LWC_Keys implements the key listener and manipulate it by release time and press time. LWC_MP3* use the certain library JLayer which decode and play the mp3 file. And LWC_Util provides the useful tools for the whole package.

1. The three layer design. One is mainly for manipulate, one is for showing, one is for the interface. Which is good to transform our game into other game type any other game type. For turn-by-turn RPG, the only thing we have to implement is let one effect notify the other effect when its turn complete. And this can be done by LWC_StatusEffect. For game type like shooting, since action game type as ours is very similar with shooting type, it can easily implements.
2. Dynamically add item/map/script/mission by simply changing the “txt” file without compilation.
3. One thread implement, easily debug and maintain.
4. Resizable available, can change the screen size at any time.
5. We cache all the images which faster our game to run.

IV. The disadvantages of your design

1. Our design is one -thread orienting, all the things is updated sequentially, so if some component updates too late, then it will block others to update.
2. We use a mask map to detect if the user can climb the rope. And the map is generated manually and time-consuming, so we should implement our own map editor to make the processes faster.

3. We cache all the images, which is memory-consuming.
4. We use the images from the internet, which is bound by the resources on the website.
5. The Effect class containing too many kinds of classes for different usage, such as animation, skills and utility effects, which make it a little chaos.

V. How to play your RPG

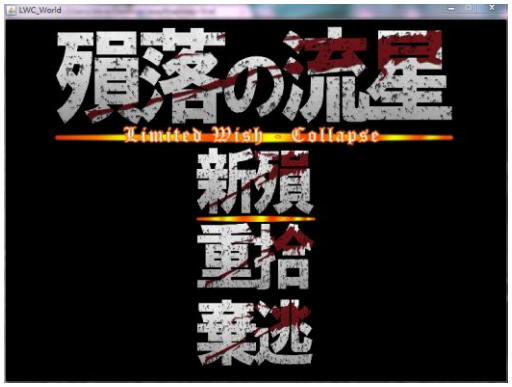
Plot:

Go straight ahead and save the little girl by killing 4 monster around, then go right to the next map for the portal to the boss.

Control:

Z: Talk Skill
F: Fight Skill
I: Open Item Menu
Ctrl: Change Skill
ESC: Main Menu
Space: Jump Skill
Move: Right and Left Arrow keys

Game Scenes:



Starting Menu

Starting animation

Talking to NPC

Fighting against monsters

The Main Menu

Dead Scene

VI. The 3-party library

1. Json-simple [code.google.com/p/json-simple]
2. JLayer-MP3 library [www.javazoom.net/javaplayer.html]