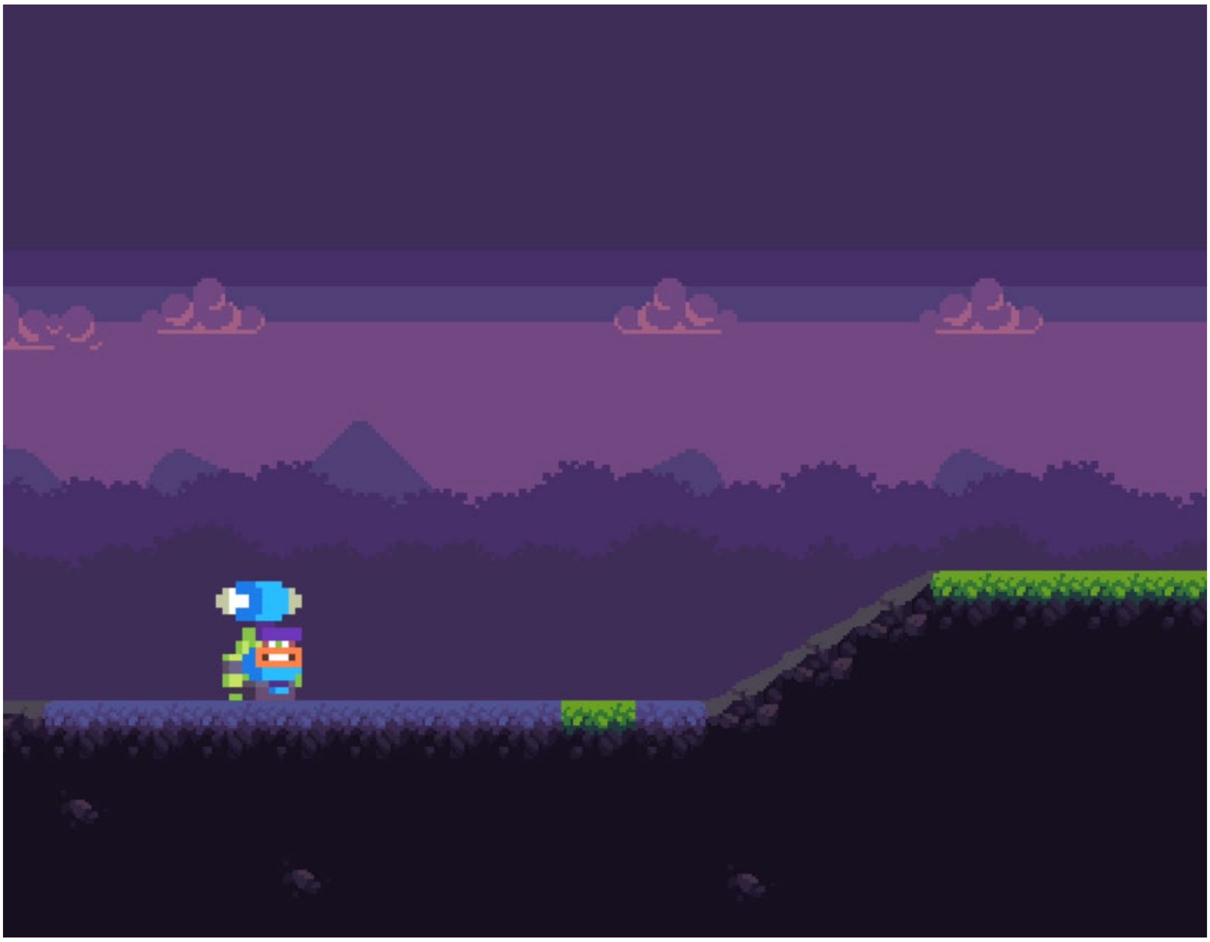
Design Report for Poised Land Adventure

Name: Hoang Phuc Huynh
Student ID: 103176852

Summary of Program

Poised Land Adventure is like its name, an adventure/ platformer game where the player traversing through the poised land to reach the key that can open the crystal gate. The crystal gate is the only way to escape a stage, but beware, there is variety of monster that is prevent player from reach the key. The player could use arrow to destroy those creatures.



Required Roles

Describe each of the classes and interfaces you will create using the following table (one per record).

Table 1: Character <<abstract>>

Responsibility	Type Details	Notes
charX	double	X coordinate
charY	double	Y coordinate
charBitmap	Bitmap	Character's Bitmap
charHealth	int	health
isOnGround	bool	Check if the character is on the ground or not
isFlip	bool	Check if the character is flip
IsAttacked	bool	Check if the character have attacked
triggerBox	Rectangle	A triggerbox for character to instantiate objects
charWeaponObject	InstantiateObjects	Handle for instantiate character weapon
hitManager	HitManager	Handle taking damage
CollisionUpdate()	void	Check if the character hit the terrain or not
Update()	void	Update the character
Draw()	void	Draw the character
Fall()	void	Check if the charcter cross an Y border, if it is then the character health is 0.

Table 2: ObjectToInstan <<Enum>>

Value	Notes
Arrow	change the object to instantiate to arrow
Tongue	change the object to instantiate to arrow

Table 3: InstantiateableObject <<Abstract>>

Responsibility	Type Details	Notes
objX	double	X coordinate
objY	double	Y coordinate
isFlip	bool	Check if the object is flip
IsDestroy	bool	Check if the object is allow to be destroy
objBitmap	bitmap	Object Bitmap
liveTime	int	The live time of the object
index	int	The index to count toward the liveTime
Update()	void	Update the object
Draw()	void	Draw the object

Table 4: Arrow , inheritance from InstantiateableObject

Responsibility	Type Details	Notes
Arrow(double arrowX, double arrowY, bool isFlip)	constructor	
Shoot()	void	Index the object X coordinate
Update()	void	Update the arrow
Draw()	void	Draw the arrow

Table 5: Tongue , inheritance from InstantiateableObject

Responsibility	Type Details	Notes
Tongue(double tongueX, double tongueY, bool isFlip)	constructor	
Update()	void	Update the tongue
Draw()	void	Draw the tongue

Table 6: InstantiateObject

Responsibility	Type Details	Notes
List< InstantiateableObject> objects	List	Contain the list of object to instantiate
pX	double	X coordinate
pY	double	Y coordinate
selectedObjectt	ObjectToInstan	Determine the object to instantiate
InstantiateObject()	constructor	
Instantiate(bool isFlip)	void	Instantiate the object to instantiate
SetObjectToInstan(ObjectToInstan obj)	void	Set the selectedObject to the object to instantiate
Update(double posX, double posY)	void	Update the object’s position with the character contain this class
Draw()	void	Draw the tongue

Table 7: Player, inheritance from Character

Responsibility	Type Details	Notes
ammunition	int	Number of arrow that player has
horizontalInput	int	Determine whether the player press to go left or right
playerSpeed	int	The speed of player

Pushback()	void	Push back the player if he takes damage
FlipPlayer()	void	Flip the player
Player(Bitmap terrain)	constructor	
Update()	void	Update the player
Draw()	void	Draw the player

Table 8: EnemyFrog, inheritance from Character

Responsibility	**Type Details**	**Notes**
player	Player	Takes the player object to process output
EnemyFrog(double frogX, double frogY, Bitmap terrain, bool isFlip, Player player)	constructor	
ScanForPlayer()	bool	Check if the triggerBox collide with the player Bitmap
Pushback()	void	Push back the player if he takes damage
FlipPlayer()	void	Flip the player
Update()	void	Update the player
Draw()	void	Draw the player

Table 9: HitManager

Responsibility	**Type Details**	**Notes**
damageableObjects	List<InstantiateObjects>	List of instantiateObject that will deal damage to the character
damageableType	List<ObjectToInstan>	List of Type that will deal damage to the character
currentDamage	int	The current damage that will be subtract to the character's health.
AddDamageableObjects(InstantiateObjects objs)	void	Add instantiateObject
AddDamageableType(ObjectToInstan type)	void	Add ObjectToInstan

IsHit()	bool	Check if the character have been hit by each of the damageable objects
DealDamage()	int	Deal damage according to the instantiateableObject that the character has been hit

Table 10: IPickupAble <<interfaces>>

Responsibility	Type Details	Notes
IsPickup()	bool	Check if the object has been picked up
Draw()	void	Draw the object
Update()	void	Update the object

Table 11: KeyPickup, implement IPickupAble

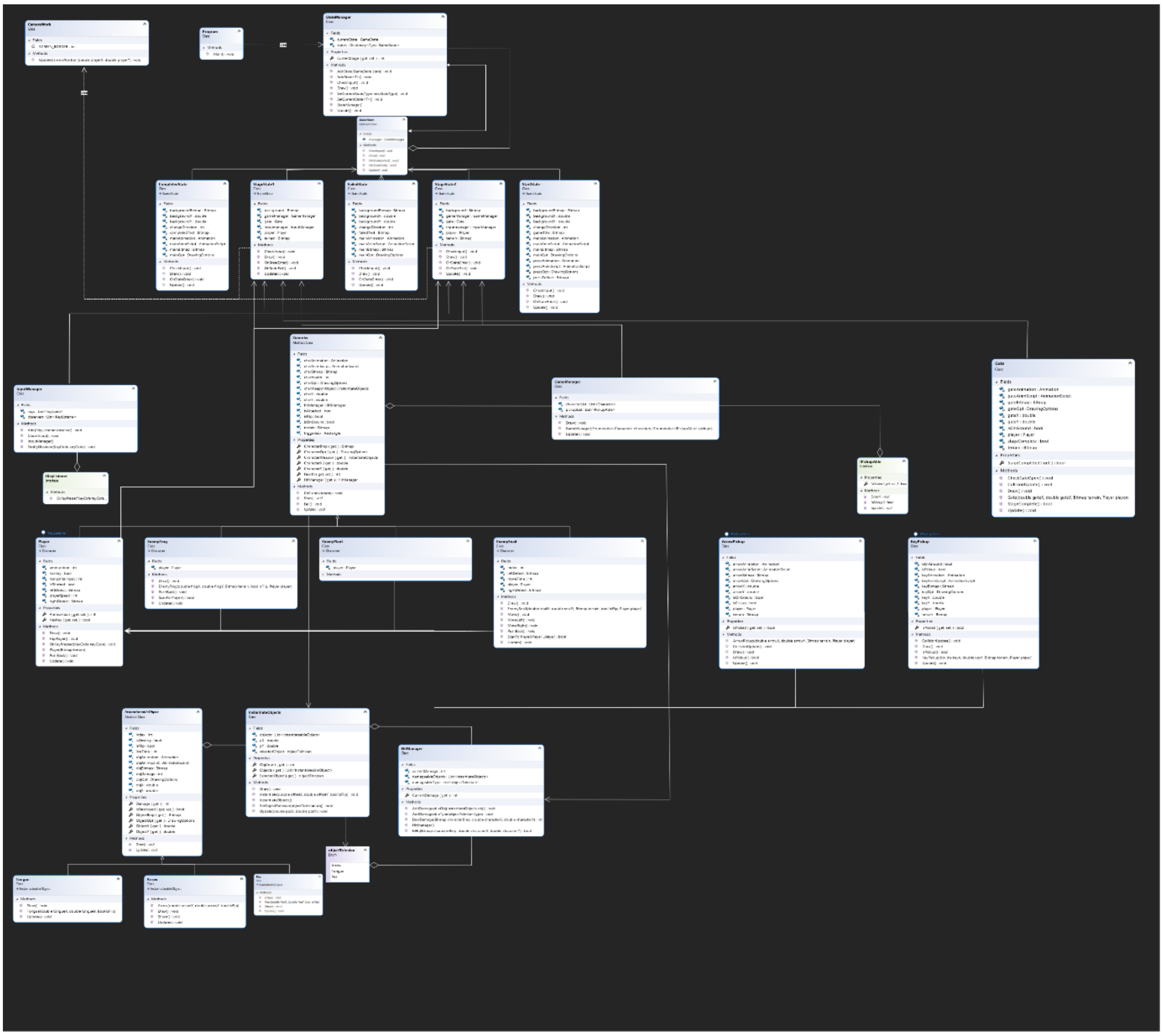
Responsibility	Type Details	Notes
isPickup	bool	Check if the object has been picked up
keyX	void	X coordinate
keyY	void	Y coordinate
player	Player	Take player object to process output
KeyPickup(double keyX, double keyY, Bitmap terrain, Player player)	contructor	

Table 12: KeyPickup, implement IPickupAble

Responsibility	Type Details	Notes
characterList	List<Character>	List of all character in the stage
pickupList	List<IPickupAble>	List of all pickup in the stage
GameManager(IEnumerable<Character> characters, IEnumerable<IPickupAble> pickups)	construtor	
Update()	void	Update the GameManager

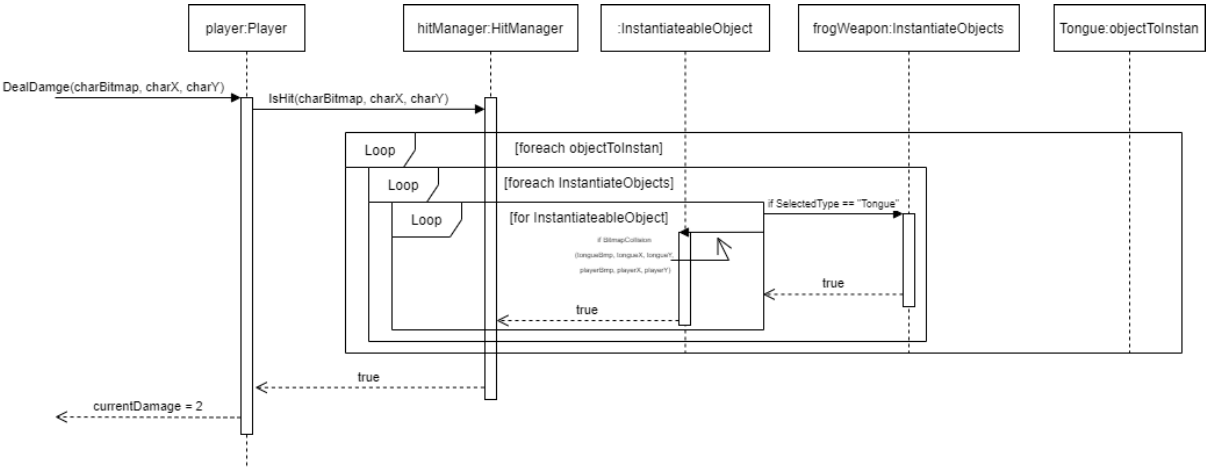
		which update all the entities in both lists
Draw()	void	Draw the GameManager which draw all the entities in both lists

UML Diagram

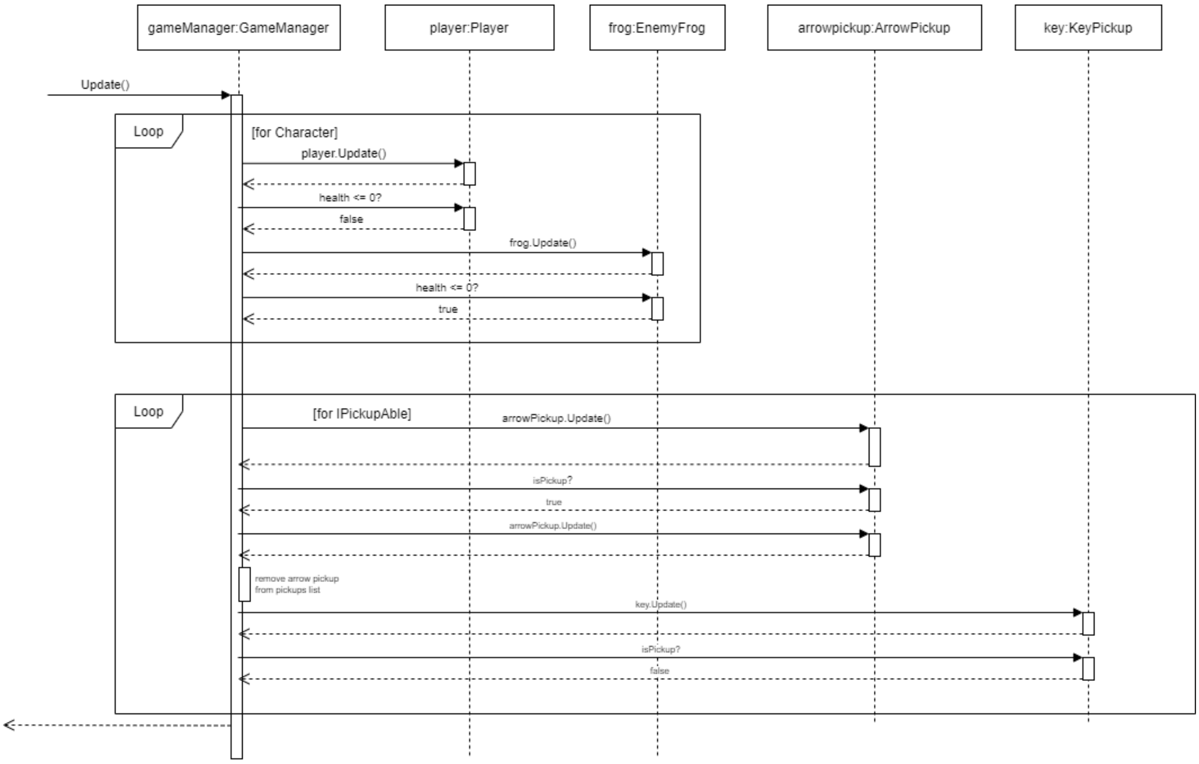


Sequence Diagram

Sequence diagram for HitManager


player:Player
hitManager:HitManager
:InstantiateableObject
frogWeapon:InstantiateObjects
Tongue:objectToInstan
DealDamge(charBitmap, charX, charY)
IsHit(charBitmap, charX, charY)
Loop
[foreach objectToInstan]
Loop
[foreach InstantiateObjects]
Loop
[for InstantiateableObject]
if SelectedType == "Tongue"
if BitmapCollision
(tongueBmp, tongueX, tongueY,
playerBmp, playerX, playerY)
true
true
true
currentDamage = 2

Sequence diagram for GameManger


gameManager:GameManager
player:Player
frog:EnemyFrog
arrowpickup:ArrowPickup
key:KeyPickup
Update()
Loop
[for Character]
player.Update()
health <= 0?
false
frog.Update()
health <= 0?
true
Loop
[for IPickupAble]
arrowPickup.Update()
isPickup?
true
arrowPickup.Update()
remove arrow pickup
from pickups list
key.Update()
isPickup?
false