13 Hacker Defense

Jake Mitchell, Ryan Craig, Phil Leonowens

Processing

Processing is a flexible software sketchbook built on top of java designed to quickly prototype graphical software.

Runs on a event loop structure

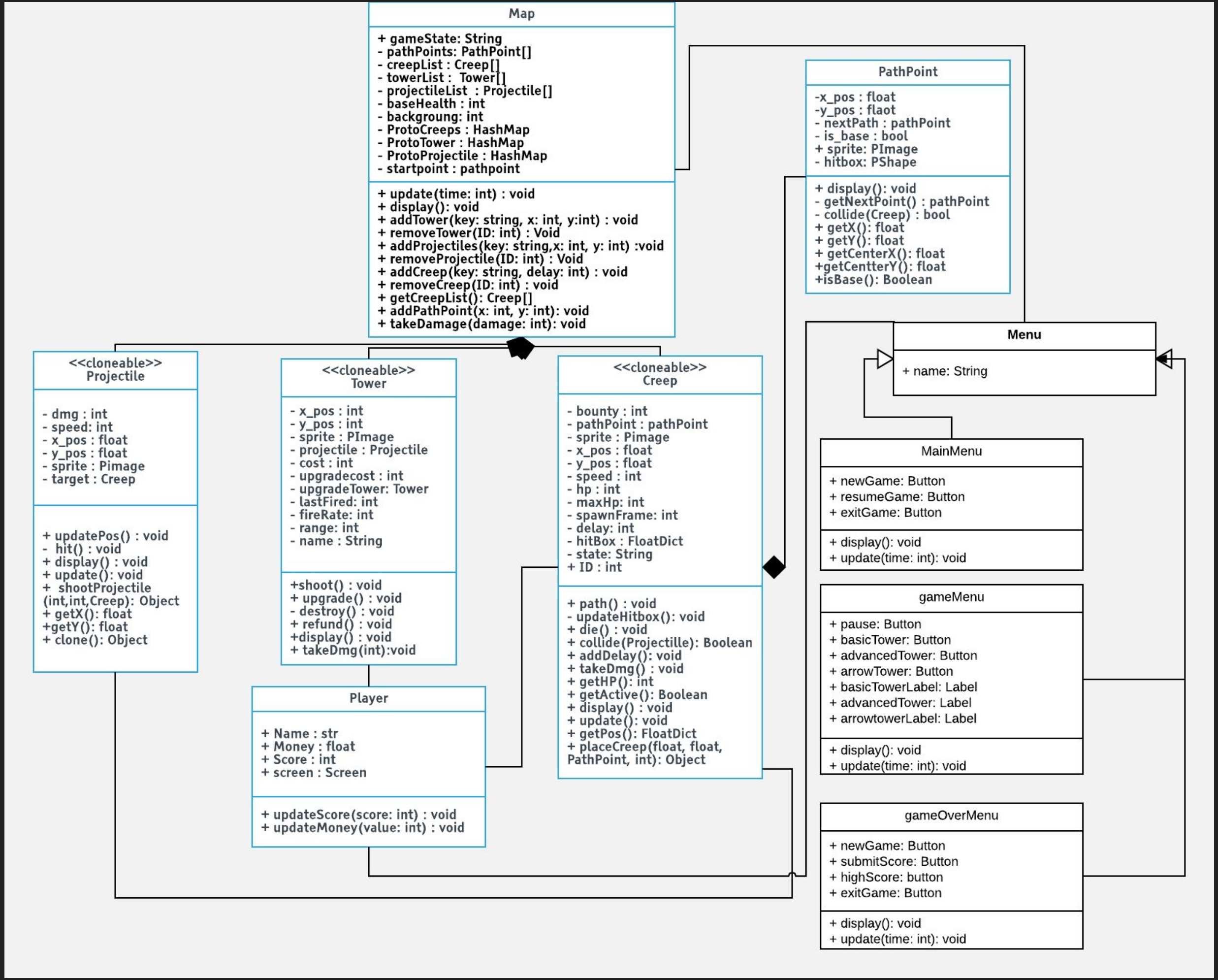
Our Goal

Originally we wanted to do a tower defense game about hacking

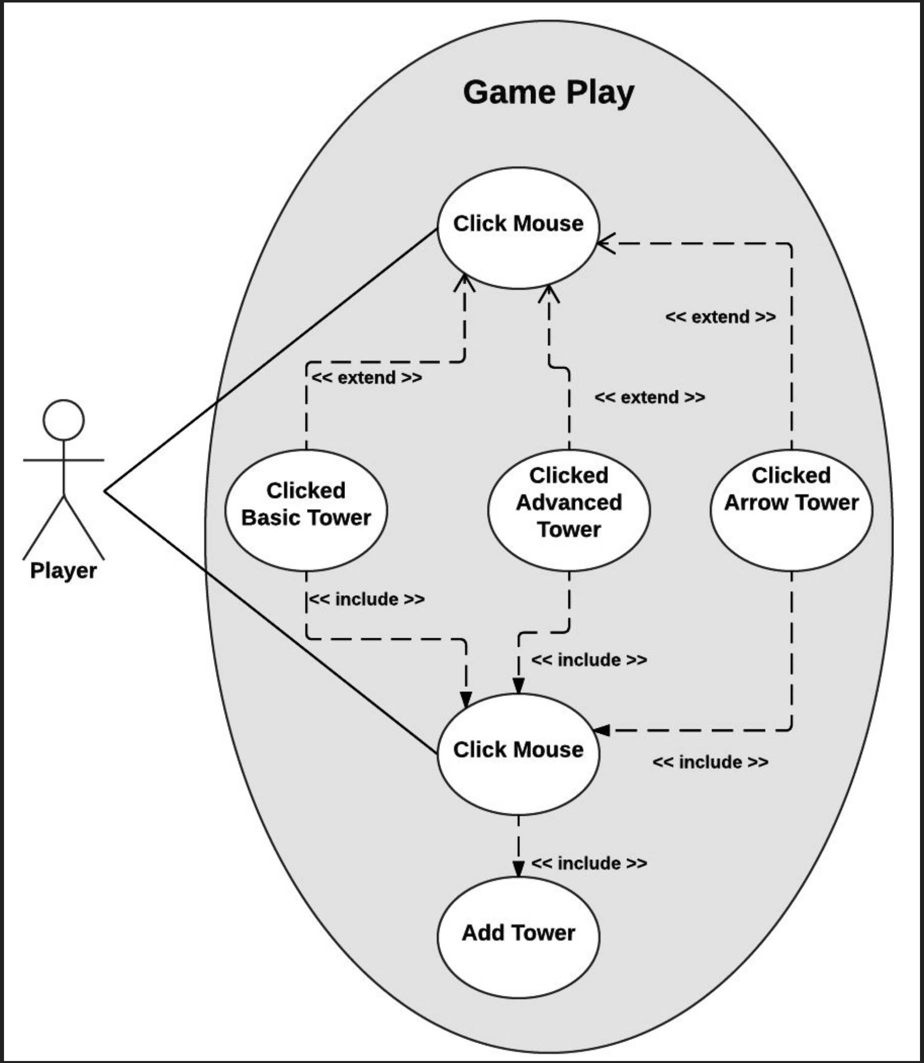
But then we decided to hack together a tower defense game

We wanted an easily extensible backend to create many variations of towers, creep, maps, to make game design easier

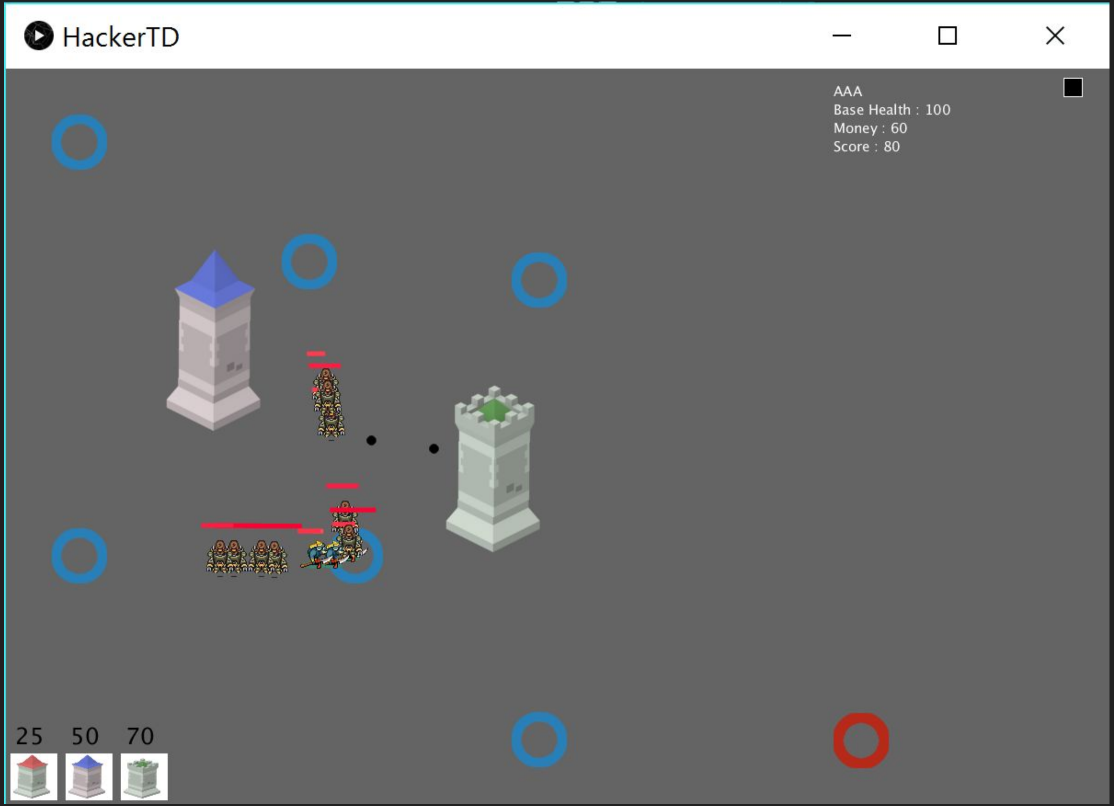
Class Diagram



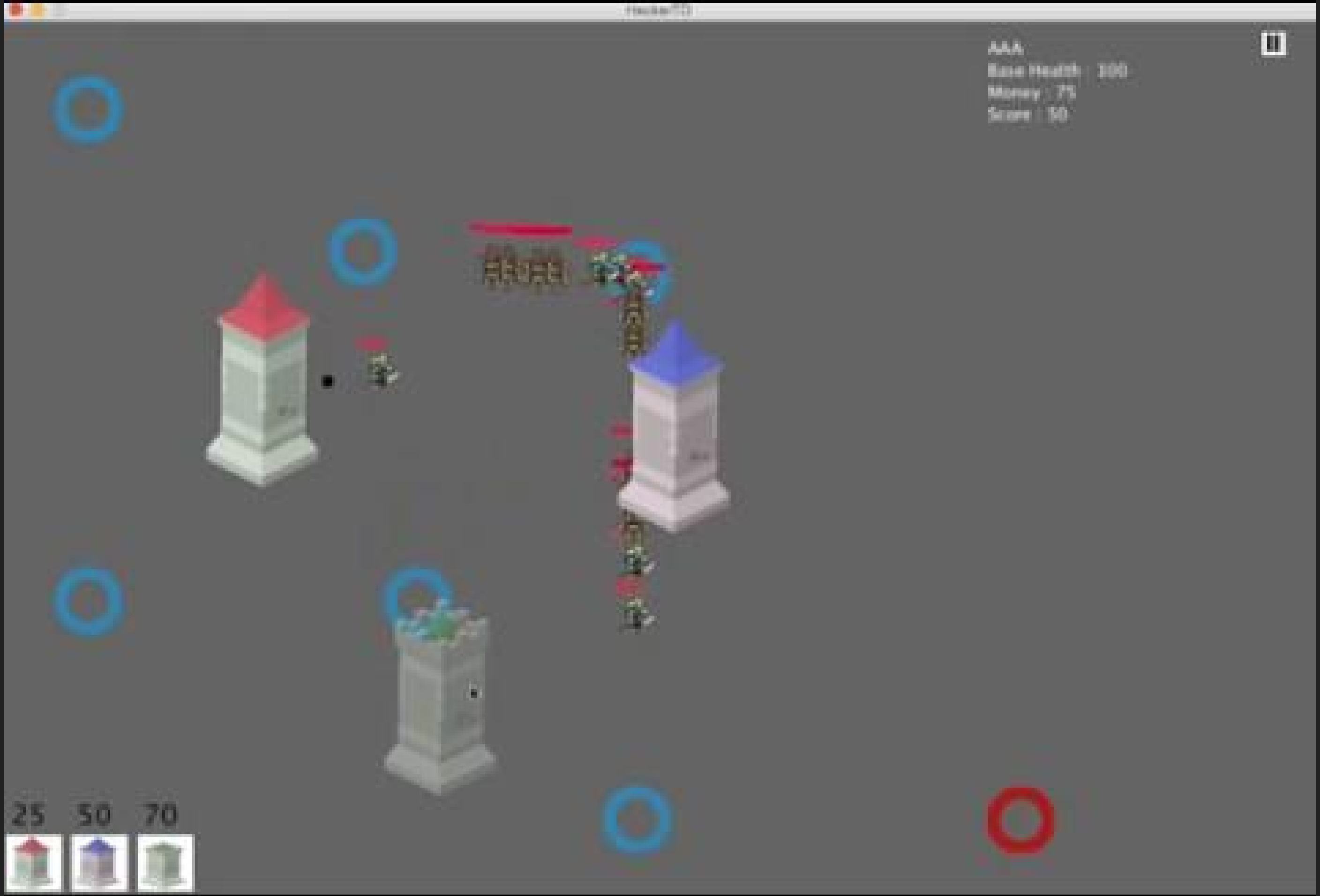
Use Case: Adding Towers



Use Case: Tower Shoots at a Creep



Use Case ID:	User-4	
Use Case Name:	Projectiles in flight	
Description:	The user must be able to see the projectiles as they fly towards their target. The projectiles must track towards their targets.	
Actors:	Projectiles, Creep	
Pre-conditions:	A projectile has been fired by a tower at a specific creep.	
Post-conditions:	The projectile's hitbox has interacted with the creep's hitbox	
Frequency of Use:	Every frame for every projectile	
Flow of Events:	Action	Response
	1. Creep moves	Projectile moves towards it
	2.	Projectile angle changes
	3. Projectile hits creep	
Variations:	Creep reaches base before projectile hits	
Notes and Issues:	Needs to run very fast in order to appear smooth	
Developer Notes:	Will likely be interacting with many classes between the draw() function and the projectile update function	

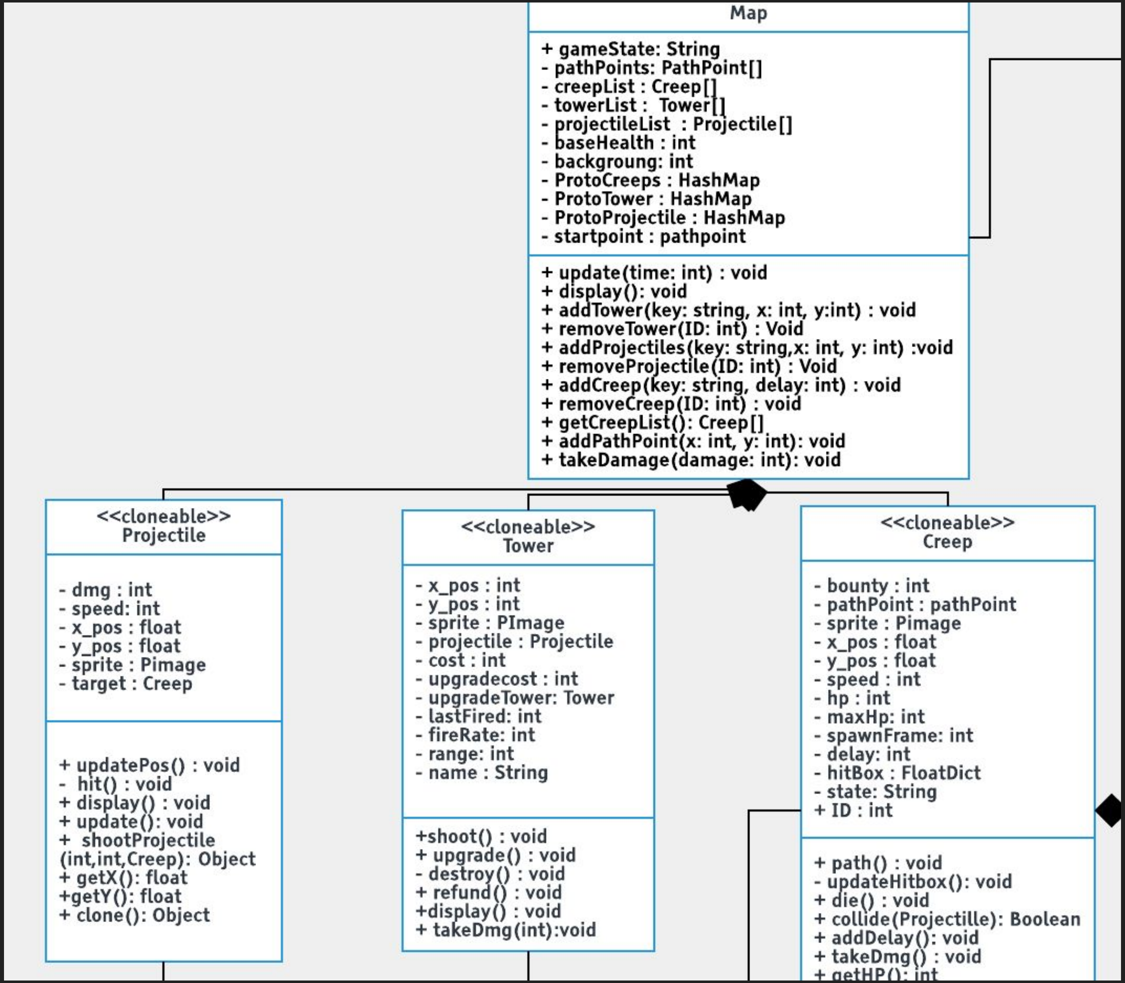

AAA
Base Health : 100
Money : 75
Score : 50
25
50
70

Design Pattern: Prototype

- Problem: We needed to create multiple instances of several objects with multiple instances having the same instance attribute values
- Subclassing wouldn’t be good because the only thing changing is the value of the instance variables
- Prototype is a creational pattern that stores a few instantiated ‘prototype’ instances, which we then clone from

Prototype Implementation

- We define hashmaps in each map that we insert objects with preset values into (towers, creeps, projectiles)
- Each of these classes implements the Cloneable interface
- When we want to instantiate one of them, we have a function we call which creates a shallow copy (clone) of the object, and we set its x and y values to place it
- A few issues
  - The fact clone is a shallow copy led to some issues where instance attributes that were objects ended up pointing
  - Have to re-instantiate each instance attribute object on creation


Map
+ gameState: String
- pathPoints: PathPoint[]
- creepList : Creep[]
- towerList : Tower[]
- projectileList : Projectile[]
- baseHealth : int
- backgroung: int
- ProtoCreeps : HashMap
- ProtoTower : HashMap
- ProtoProjectile : HashMap
- startpoint : pathpoint
+ update(time: int) : void
+ display(): void
+ addTower(key: string, x: int, y:int) : void
+ removeTower(ID: int) : Void
+ addProjectiles(key: string,x: int, y: int) :void
+ removeProjectile(ID: int) : Void
+ addCreep(key: string, delay: int) : void
+ removeCreep(ID: int) : void
+ getCreepList(): Creep[]
+ addPathPoint(x: int, y: int): void
+ takeDamage(damage: int): void
<<cloneable>>
Projectile
- dmg : int
- speed: int
- x_pos : float
- y_pos : float
- sprite : Pimage
- target : Creep
+ updatePos() : void
- hit() : void
+ display() : void
+ update(): void
+ shootProjectile (int,int,Creep): Object
+ getX(): float
+getY(): float
+ clone(): Object
<<cloneable>>
Tower
- x_pos : int
- y_pos : int
- sprite : PImage
- projectile : Projectile
- cost : int
- upgradecost : int
- upgradeTower: Tower
- lastFired: int
- fireRate: int
- range: int
- name : String
+shoot() : void
+ upgrade() : void
- destroy() : void
+ refund() : void
+display() : void
+ takeDmg(int):void
<<cloneable>>
Creep
- bounty : int
- pathPoint : pathPoint
- sprite : Pimage
- x_pos : float
- y_pos : float
- speed : int
- hp : int
- maxHp: int
- spawnFrame: int
- delay: int
- hitBox : FloatDict
- state: String
+ ID : int
+ path() : void
- updateHitbox(): void
+ die() : void
+ collide(Projectille): Boolean
+ addDelay(): void
+ takeDmg() : void
+ getHP(): int

Play Quick Round

Demo Video Link

https://drive.google.com/file/d/0BwGYWAoSg9ABS2J5Wmkyd0JURTg/view?usp=sharing