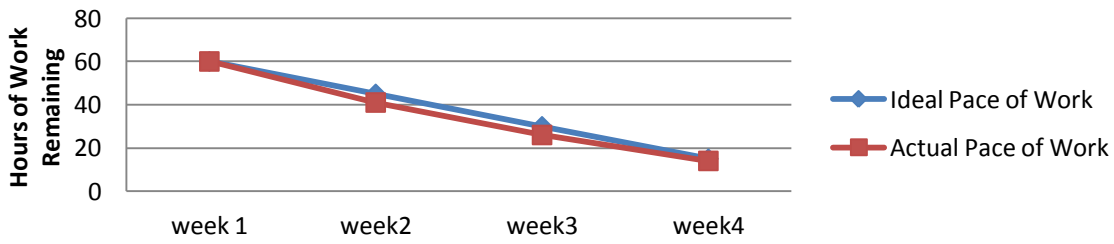
Sprint Backlog

Backlog Item	Task	Task Owner	Initial Estimate	Hours Of Work Remaining			
				Week 1	Week 2	Week 3	Week 4
Game involving motion detection	Indroducing the kinnect motion sensors to the code to detect the Players motion.	Sneha	4	4	2	1	1
	Introduce special powers to players when they traverse thro' various levels as an incentive. Implemented using Singleton and Factory Design Pattern.	Sneha	4	4	2	2	1
Change scenarios in each level	Introduce different levels within a selected Game mode.	Madhu	4	4	3	1	2
	Apply the State Design Pattern to implement the diferent states.	Madhu					
Score card display and Animation and Sound Effects	ScoreCard is designed to observe and update the score and other entities as they change dynamically and mend the design. Observer Design Pattern used.	Gayathri	4	4	3	2	0
	Animations and Sound Effects given to enhance the game feel.	Gayathri	4	4	2	1	1
Introduce different game modes	Introduce different game modes to choose from like Easy,Medium,Hard by implementing Strategy Design Pattern.	Aishwariyaa	4	4	2	1	0
Documentation	Designed the UML Class Diagram for game implemented.	Aishwariyaa	4	4	3	2	1
	Drew up the Scrum cards and the write up for the Design Patterns implemented in the game	Aishwariyaa	4	4	3	2	1
	Drew up the Scrum cards and the write up for the Design Patterns implemented in the game	Madhu	4	4	3	2	1
	Drew up the Scrum cards and the write up for the Design Patterns implemented in the game	Sneha	4	4	3	2	1
	Drew up the Scrum cards and the write up for the Design Patterns implemented in the game	Gayathri	4	4	3	2	1
	Designed the Sequence and Use Case Diagram.	Gayathri	4	4	3	2	1
	UI wireframes taken of the game for the Presentation.	Sneha	4	4	3	2	1
	Designed the UML Activity Diagram for game implemented.	Madhu	4	4	3	2	1
	Final consolidation of the requirements in to a document.	Madhu	4	4	3	2	1

Ideal Total	60	45	30	15
Actual Total	60	41	26	14

Burndown Chart



Weekly Scrum Charts : Snehlata Kulkarni

Project Group 1,Sprint#2,Week 1

Team Member Name
Snehlata Kulkarni

What I plan to do today
1. Brainstorm on feasible design patterns for implementation
2. Install required driver softwares and other dependencies for Kinect connection.

Project Group 1,Sprint#2,Week 2

Team Member Name
Snehlata Kulkarni

What I did since the last Weekly Scrum
1. Finalized on the possible patterns:Strategy, State, Observer Factory Method and Singleton.
2. Narrowed down different interaction and problem aspects of the game.
3. Tried Installing kinect drivers on Mac and Linux.

What I plan to do today
1. Install VM for Windows and try to reinstall kinect greenfoot server.
2. Baseline the code and classes for others to use.

Project Group 1,Sprint#2,Week 3

Team Member Name
Snehlata Kulkarni

What I did since the last Weekly Scrum
1. Drafted the Factory and Singleton Design pattern on Astah.
2. Installed the software and server for Kinect and tried different example scenarios.
3. Started Implementation of design pattern.

What I plan to do today
1. Complete the code implementation for the pattern.
2. Try different kinect motion integration.

Project Group 1,Sprint#2,Week 4

Team Member Name
Snehlata Kulkarni

What I did since the last Weekly Scrum
1. Implemented the desired patterns and completed coding.
2. Integrate code with the other team members
3. Apply changes as desired.

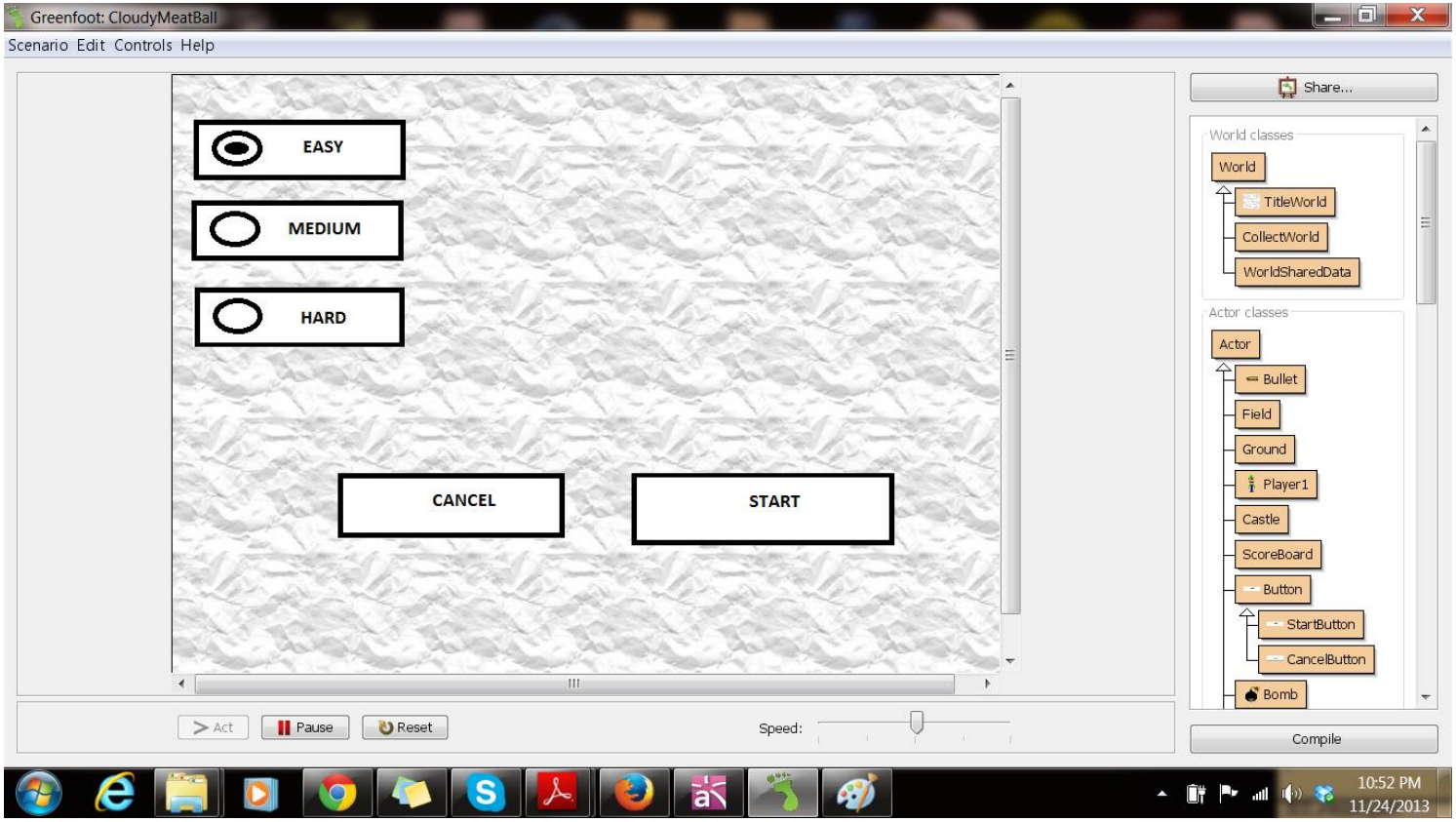
What I plan to do today
1. Refine functionality of the Game
2. Integrate the whole project run and test it with kinect.

Individual Implementation – Snehlata Kulkarni

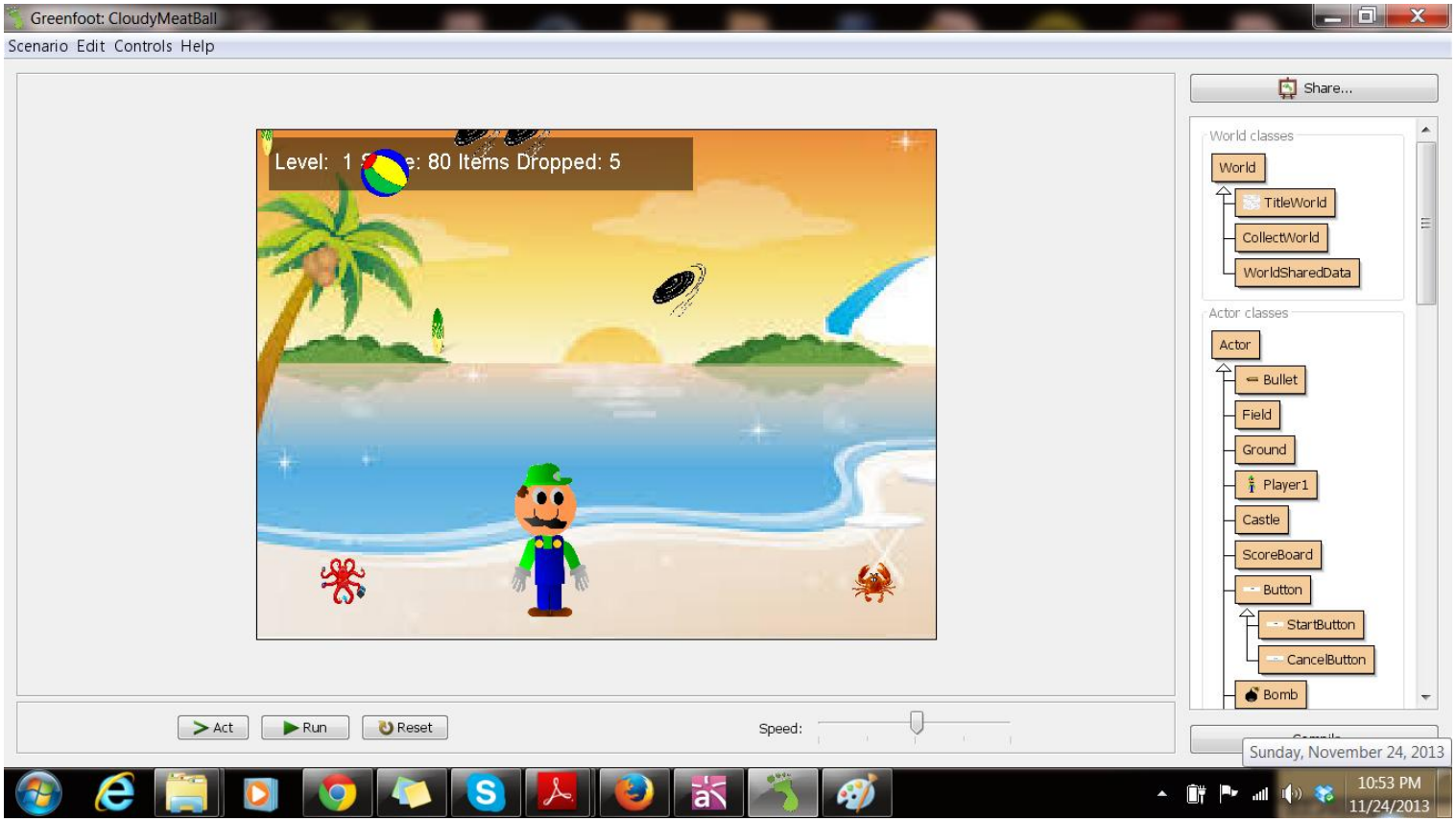
- Architected the code for project as per finalized project idea.
    - Prepared the skeleton code and added basic functionality for other team members to work upon.
    - Added and Implemented code and logic for Action of the Player, Random Monsters, Items and Weapons.
- Implemented the ‘Factory Method’ Pattern for Weapons and ‘Singleton Pattern’ for Player.
- Learnt and Installed Kinect Drivers.
- Modified the actions of the classes in accordance with Kinect.

UI Wireframes of the Code

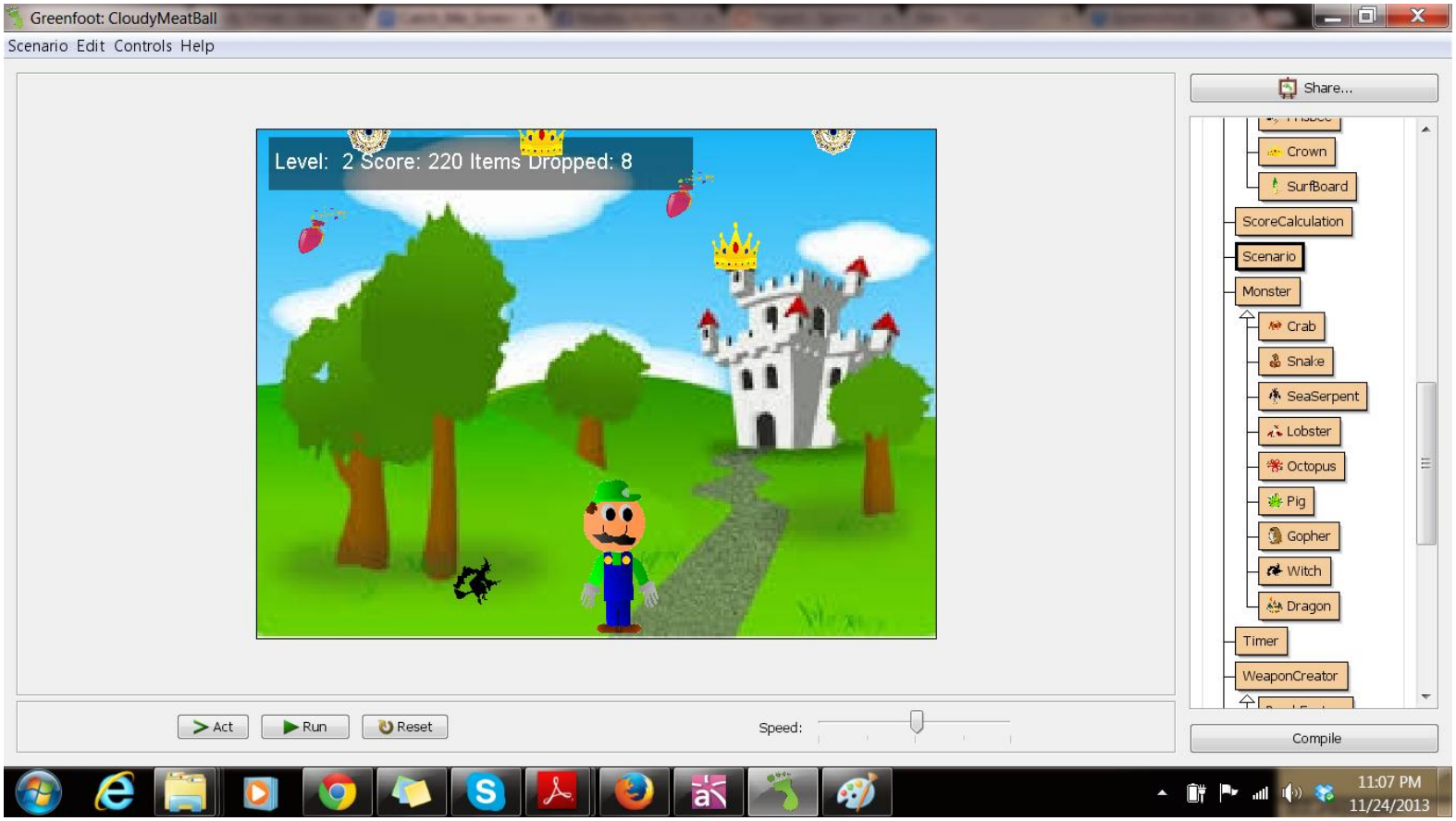
Start Screen -Easy Mode Selected



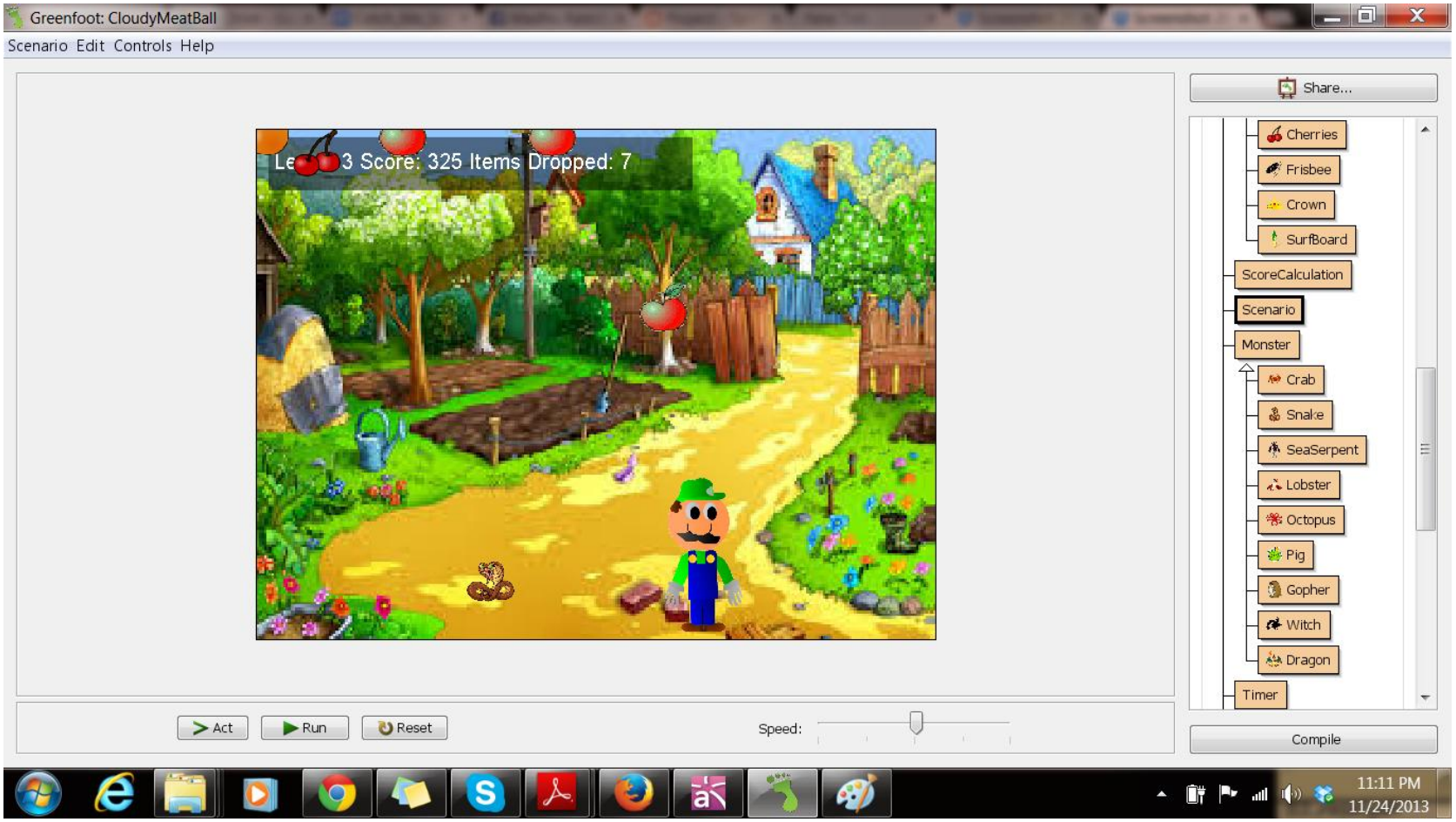
Level --1 Beach



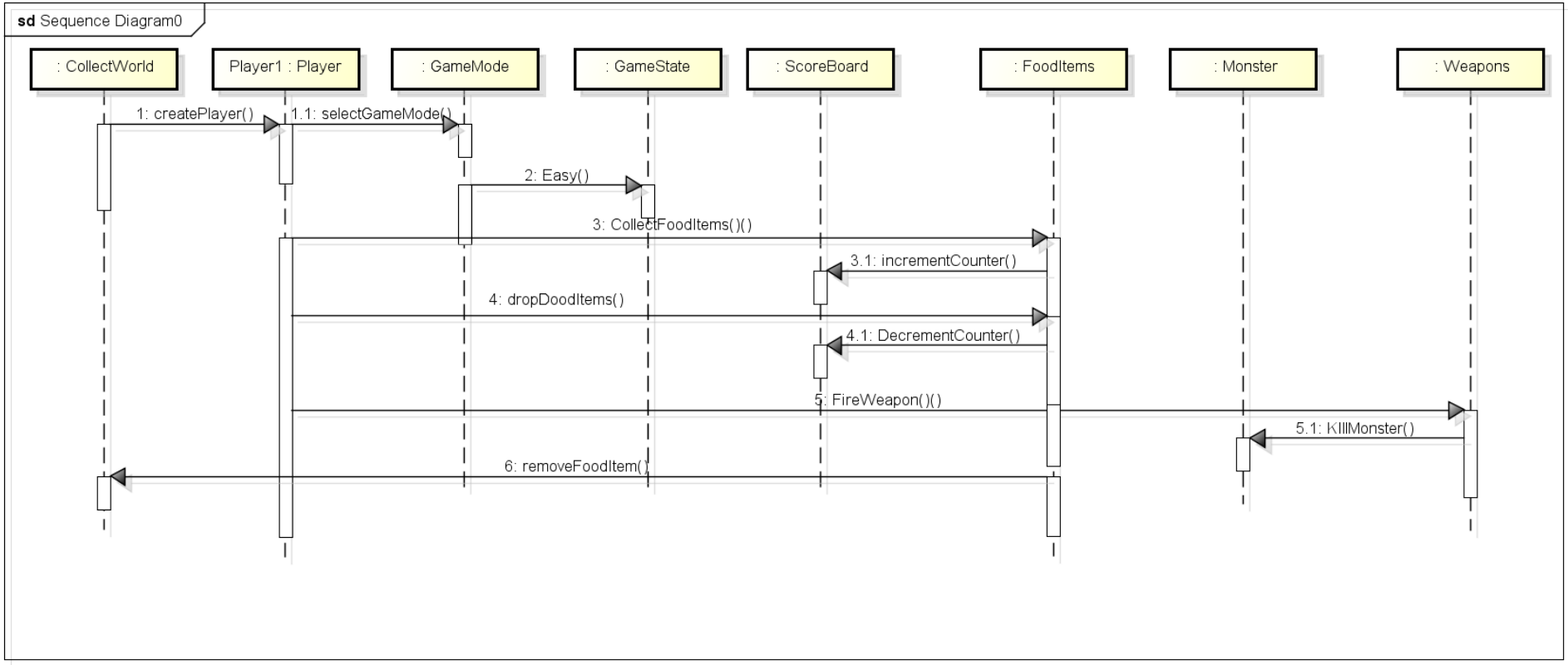
When 10 Items collected ,Level changes to Castle



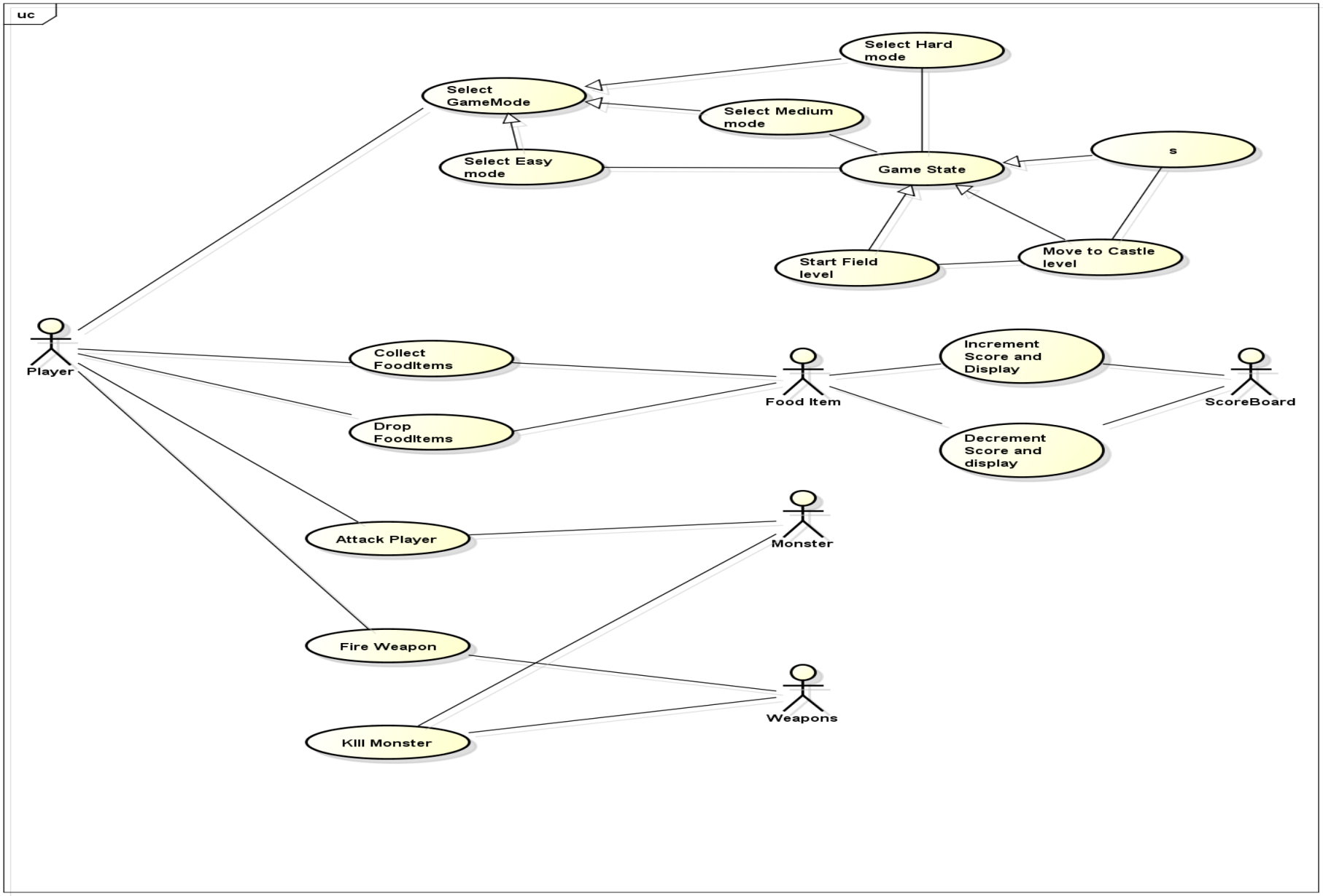
Again when 10 items collected on this level,the player enters to next level Field



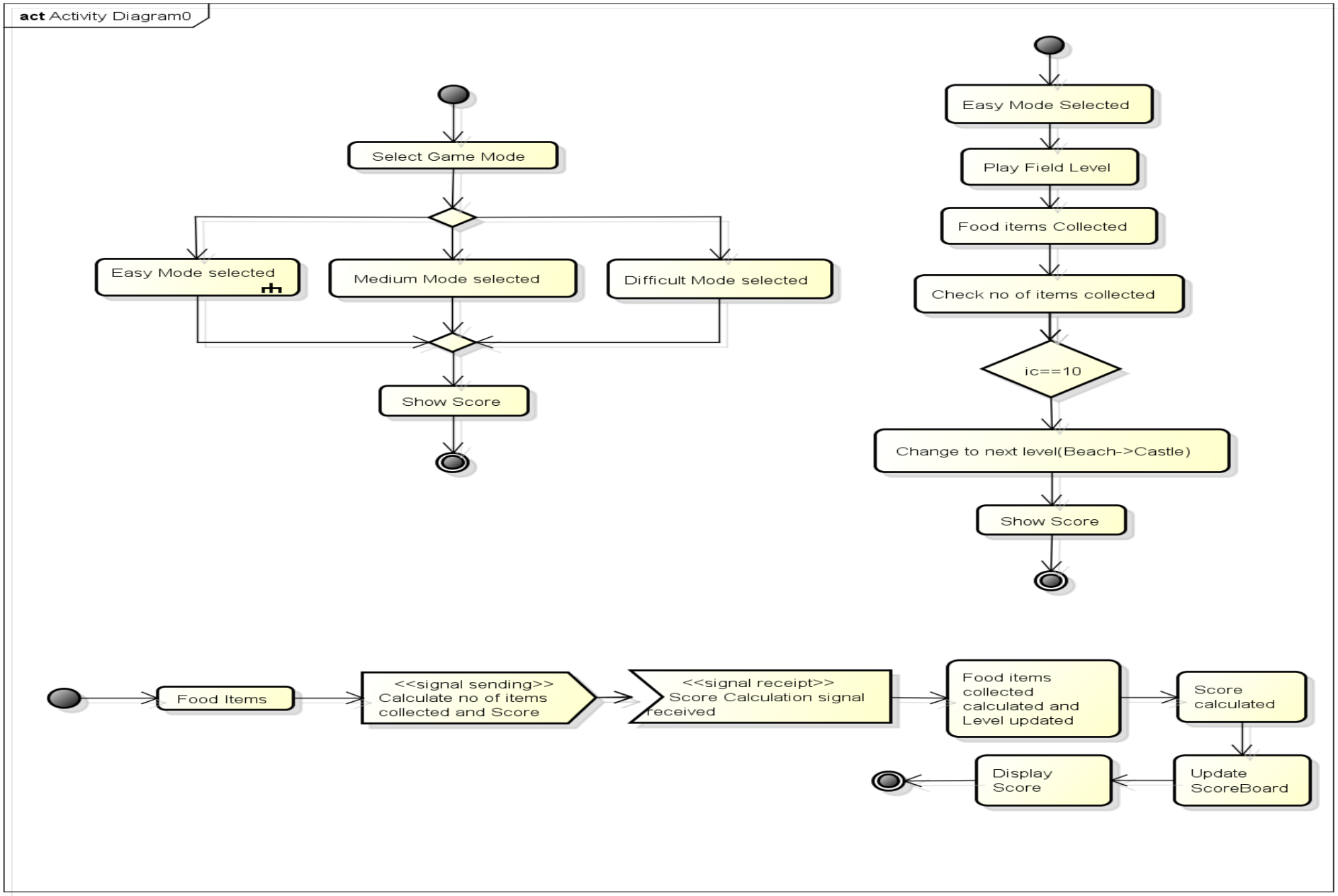
Sequence Diagram



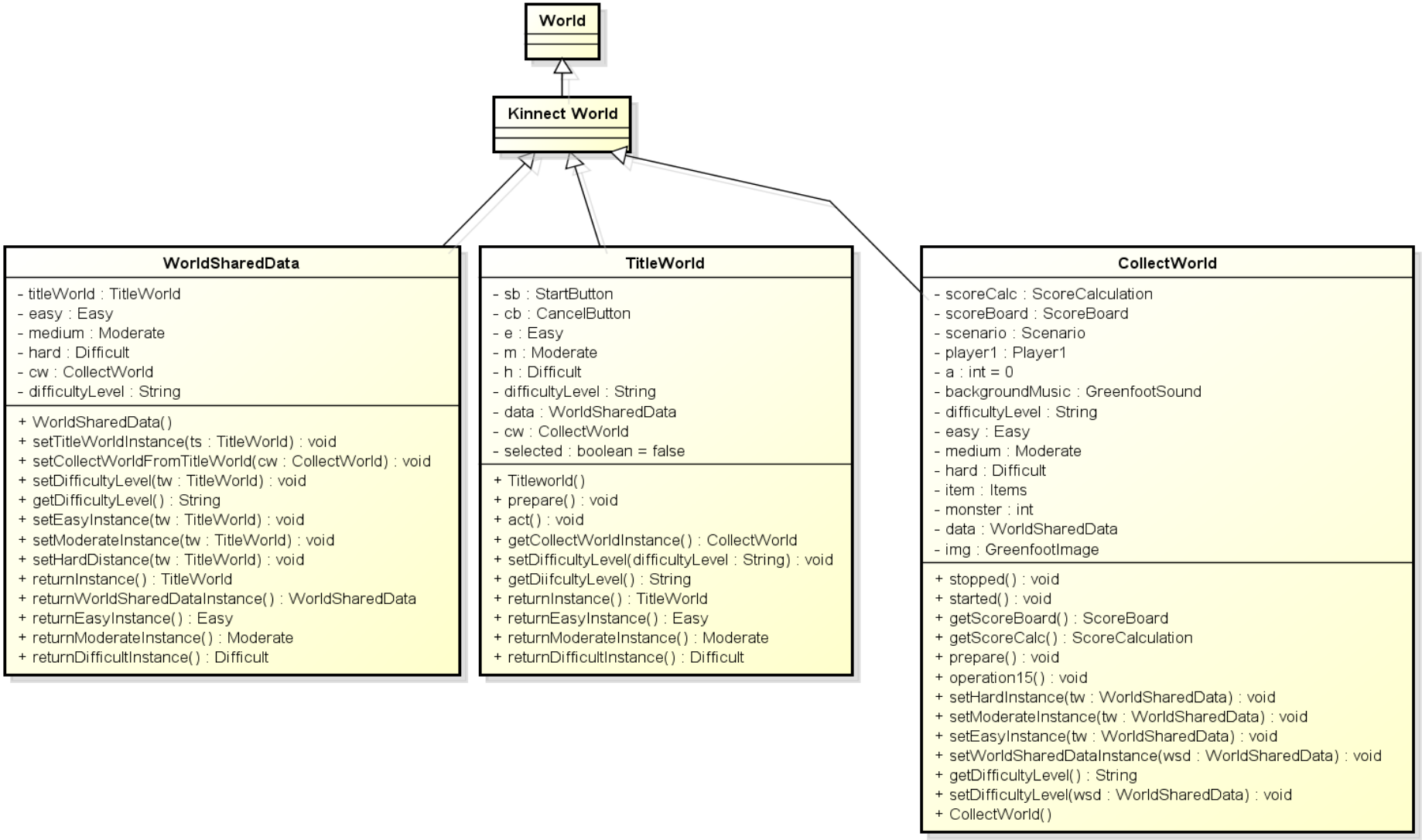
Use Case Diagram

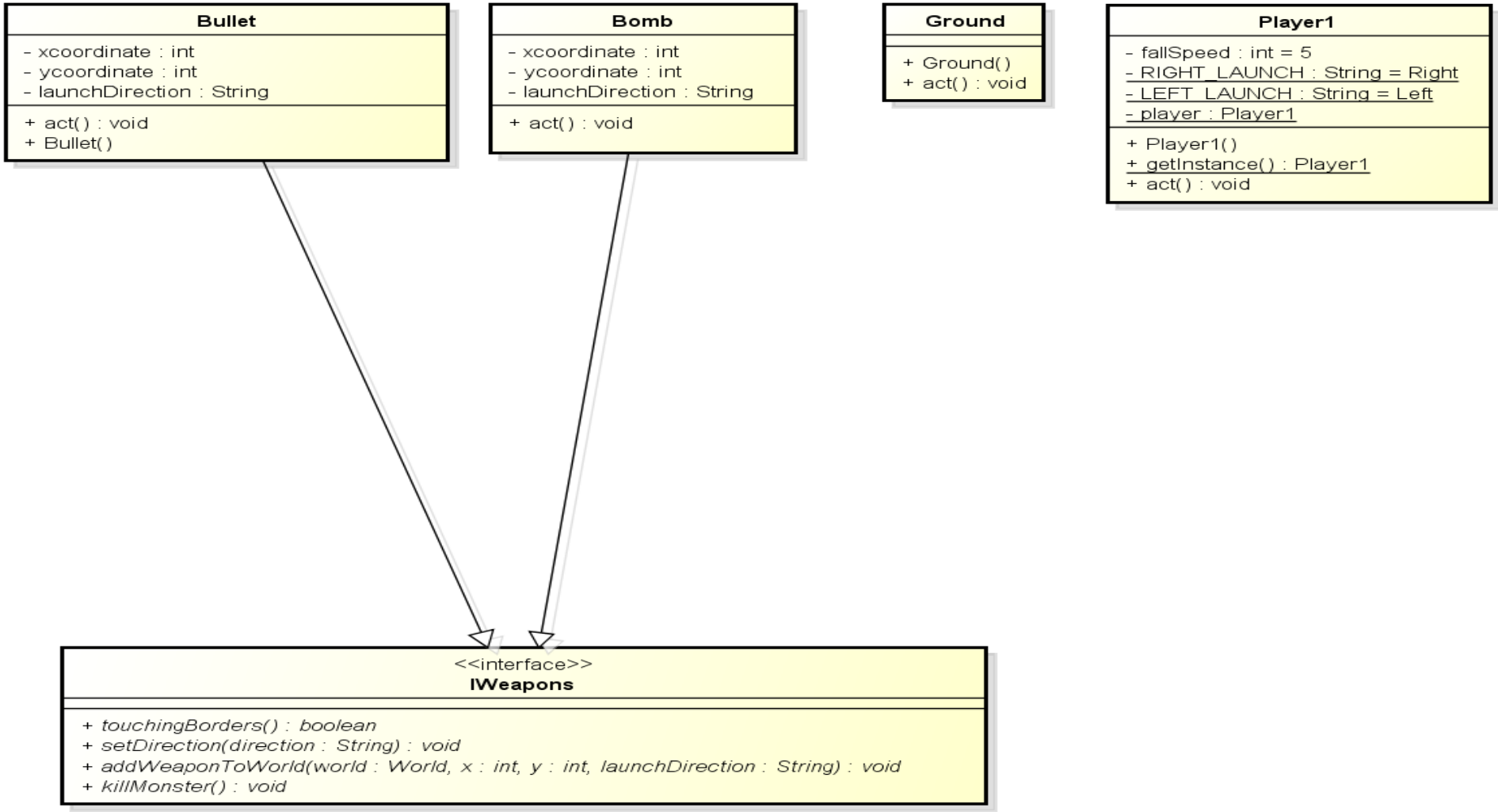


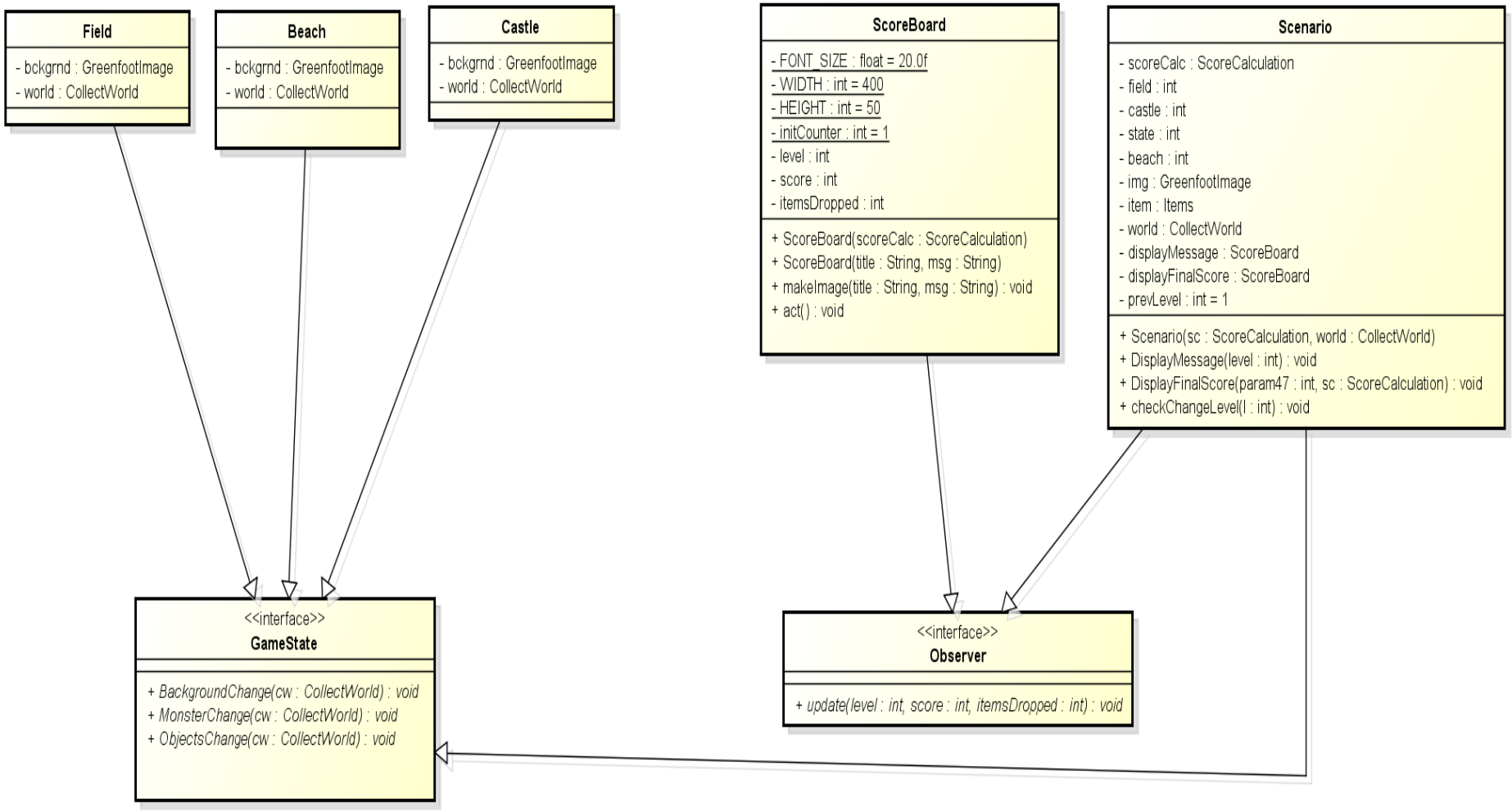
Activity Diagram

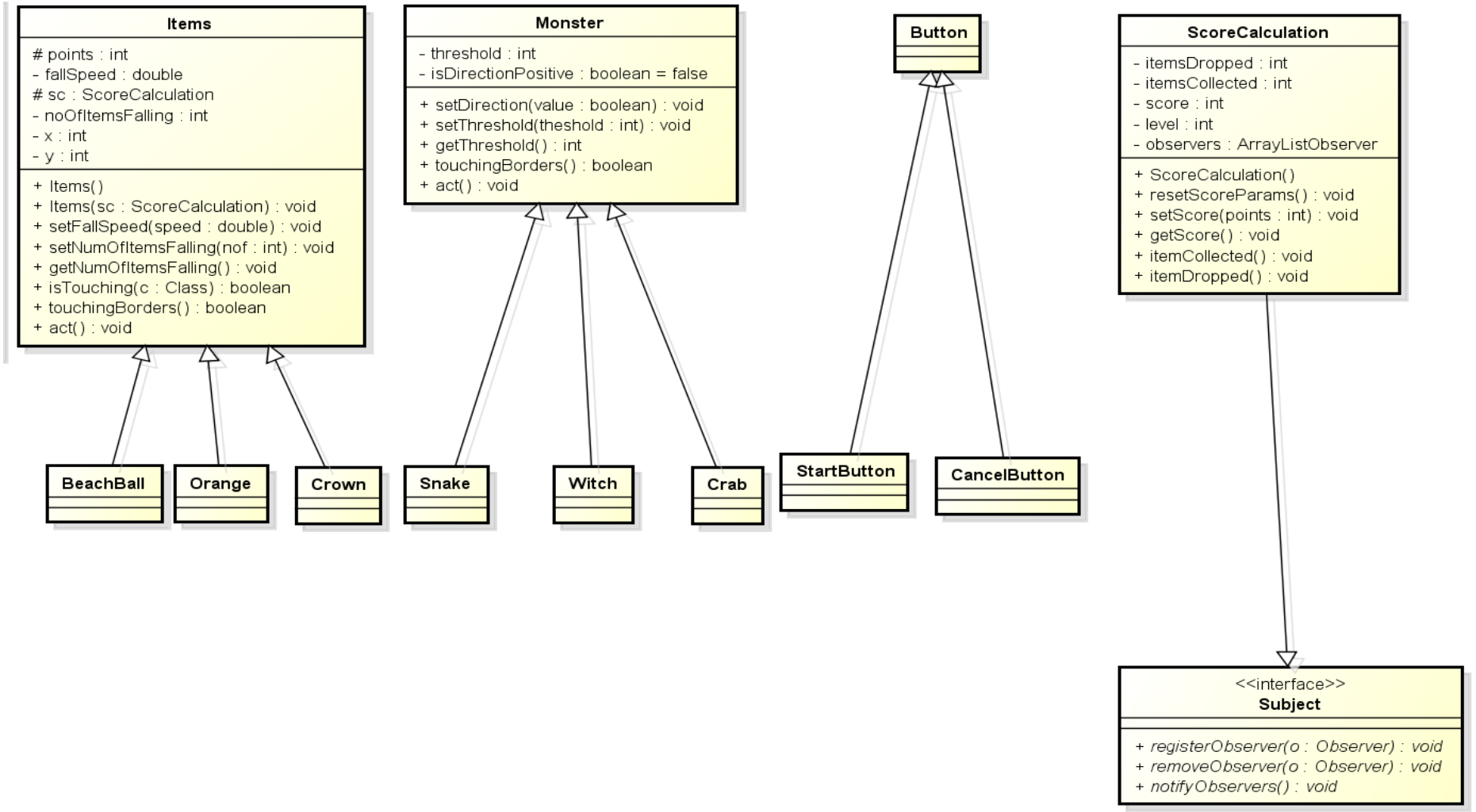


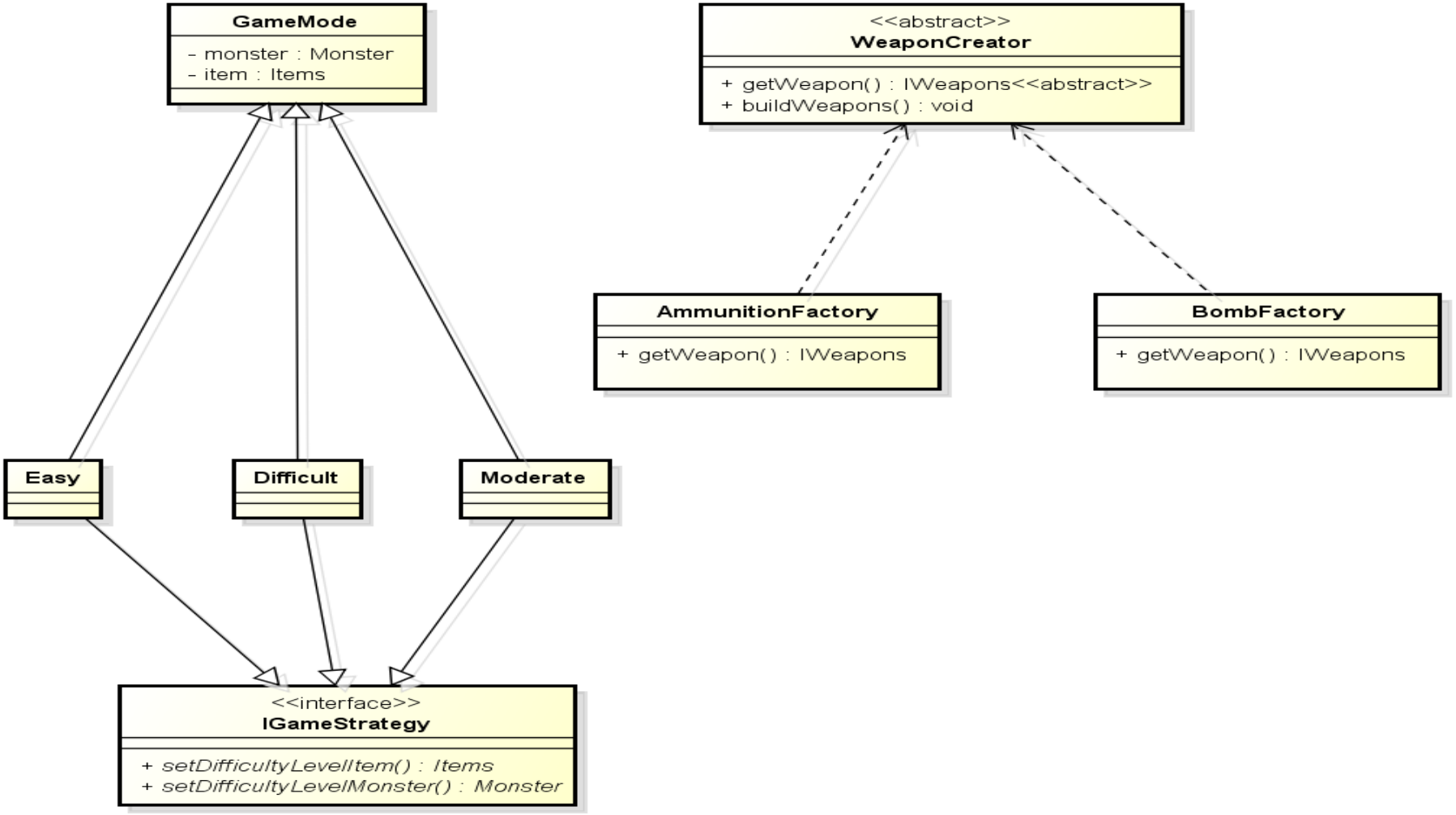
UML Class Diagram




Bullet
- xcoordinate : int
- ycoordinate : int
- launchDirection : String
+ act( ) : void
+ Bullet( )
Bomb
- xcoordinate : int
- ycoordinate : int
- launchDirection : String
+ act( ) : void
Ground
+ Ground( )
+ act( ) : void
Player1
- fallSpeed : int = 5
- RIGHT_LAUNCH : String = Right
- LEFT_LAUNCH : String = Left
- player : Player1
+ Player1( )
+ getInstance( ) : Player1
+ act( ) : void
<<interface>>
IWeapons
+ touchingBorders( ) : boolean
+ setDirection(direction : String) : void
+ addWeaponToWorld(world : World, x : int, y : int, launchDirection : String) : void
+ killMonster( ) : void


Field
- bckgrnd : GreenfootImage
- world : CollectWorld
Beach
- bckgrnd : GreenfootImage
- world : CollectWorld
Castle
- bckgrnd : GreenfootImage
- world : CollectWorld
ScoreBoard
- FONT_SIZE : float = 20.0f
- WIDTH : int = 400
- HEIGHT : int = 50
- initCounter : int = 1
- level : int
- score : int
- itemsDropped : int
+ ScoreBoard(scoreCalc : ScoreCalculation)
+ ScoreBoard(title : String, msg : String)
+ makeImage(title : String, msg : String) : void
+ act() : void
Scenario
- scoreCalc : ScoreCalculation
- field : int
- castle : int
- state : int
- beach : int
- img : GreenfootImage
- item : Items
- world : CollectWorld
- displayMessage : ScoreBoard
- displayFinalScore : ScoreBoard
- prevLevel : int = 1
+ Scenario(sc : ScoreCalculation, world : CollectWorld)
+ DisplayMessage(level : int) : void
+ DisplayFinalScore(param47 : int, sc : ScoreCalculation) : void
+ checkChangeLevel(l : int) : void
<<interface>>
GameState
+ BackgroundChange(cw : CollectWorld) : void
+ MonsterChange(cw : CollectWorld) : void
+ ObjectsChange(cw : CollectWorld) : void
<<interface>>
Observer
+ update(level : int, score : int, itemsDropped : int) : void


Items
points : int
- fallSpeed : double
sc : ScoreCalculation
- noOfItemsFalling : int
- x : int
- y : int
+ Items()
+ Items(sc : ScoreCalculation) : void
+ setFallSpeed(speed : double) : void
+ setNumOfItemsFalling(nof : int) : void
+ getNumOfItemsFalling() : void
+ isTouching(c : Class) : boolean
+ touchingBorders() : boolean
+ act() : void
Monster
- threshold : int
- isDirectionPositive : boolean = false
+ setDirection(value : boolean) : void
+ setThreshold(theshold : int) : void
+ getThreshold() : int
+ touchingBorders() : boolean
+ act() : void
Button
ScoreCalculation
- itemsDropped : int
- itemsCollected : int
- score : int
- level : int
- observers : ArrayListObserver
+ ScoreCalculation()
+ resetScoreParams() : void
+ setScore(points : int) : void
+ getScore() : void
+ itemCollected() : void
+ itemDropped() : void
BeachBall
Orange
Crown
Snake
Witch
Crab
StartButton
CancelButton
<<interface>>
Subject
+ registerObserver(o : Observer) : void
+ removeObserver(o : Observer) : void
+ notifyObservers() : void


GameMode
- monster : Monster
- item : Items
<<abstract>>
WeaponCreator
+ getWeapon( ) : IWeapons<<abstract>>
+ buildWeapons( ) : void
AmmunitionFactory
+ getWeapon( ) : IWeapons
BombFactory
+ getWeapon( ) : IWeapons
Easy
Difficult
Moderate
<<interface>>
IGameStrategy
+ setDifficultyLevelItem( ) : Items
+ setDifficultyLevelMonster( ) : Monster