Design patterns

1.

The singleton pattern:

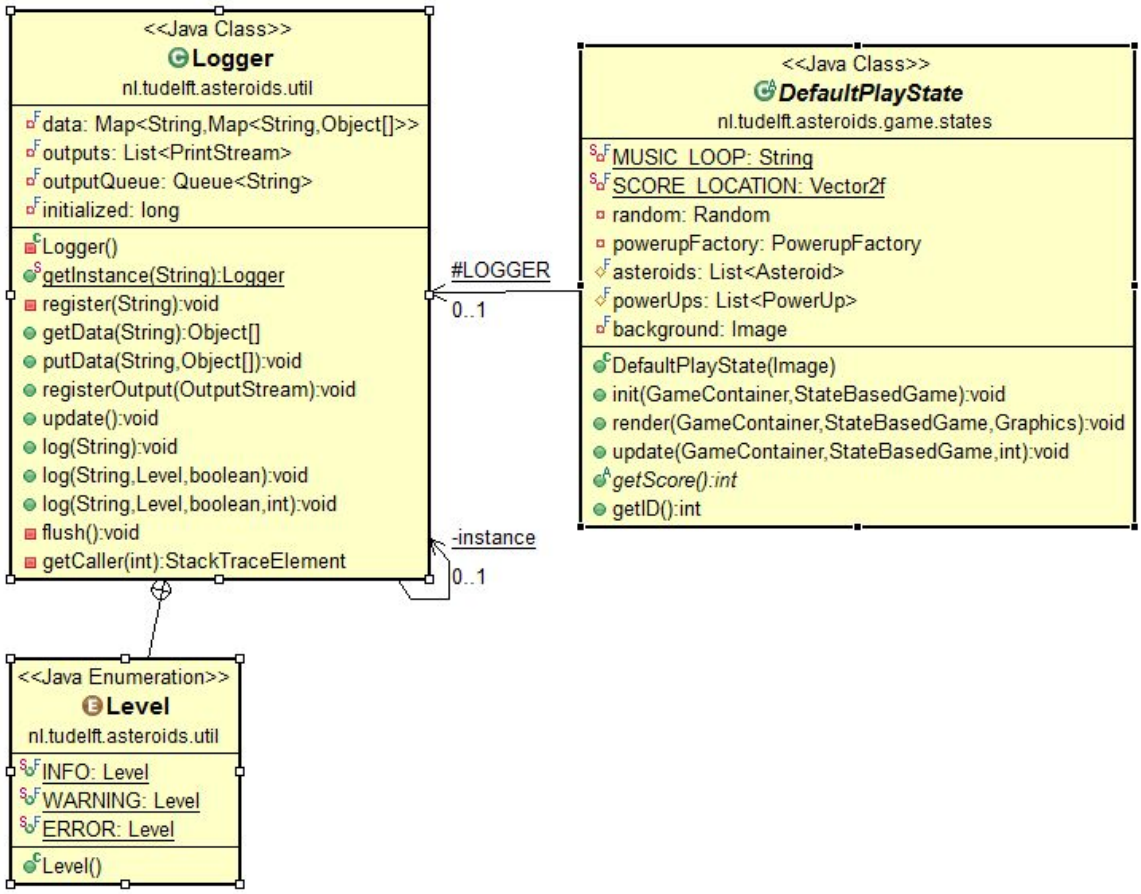
The Logger class implements the singleton desgin pattern. In every class that needs logging a new Logger is created. This class also has a getInstance method. The method that is called for the logger is moslty the log method.

If a class needs logging simply add a new Logger and you can call the log method wherever you want.

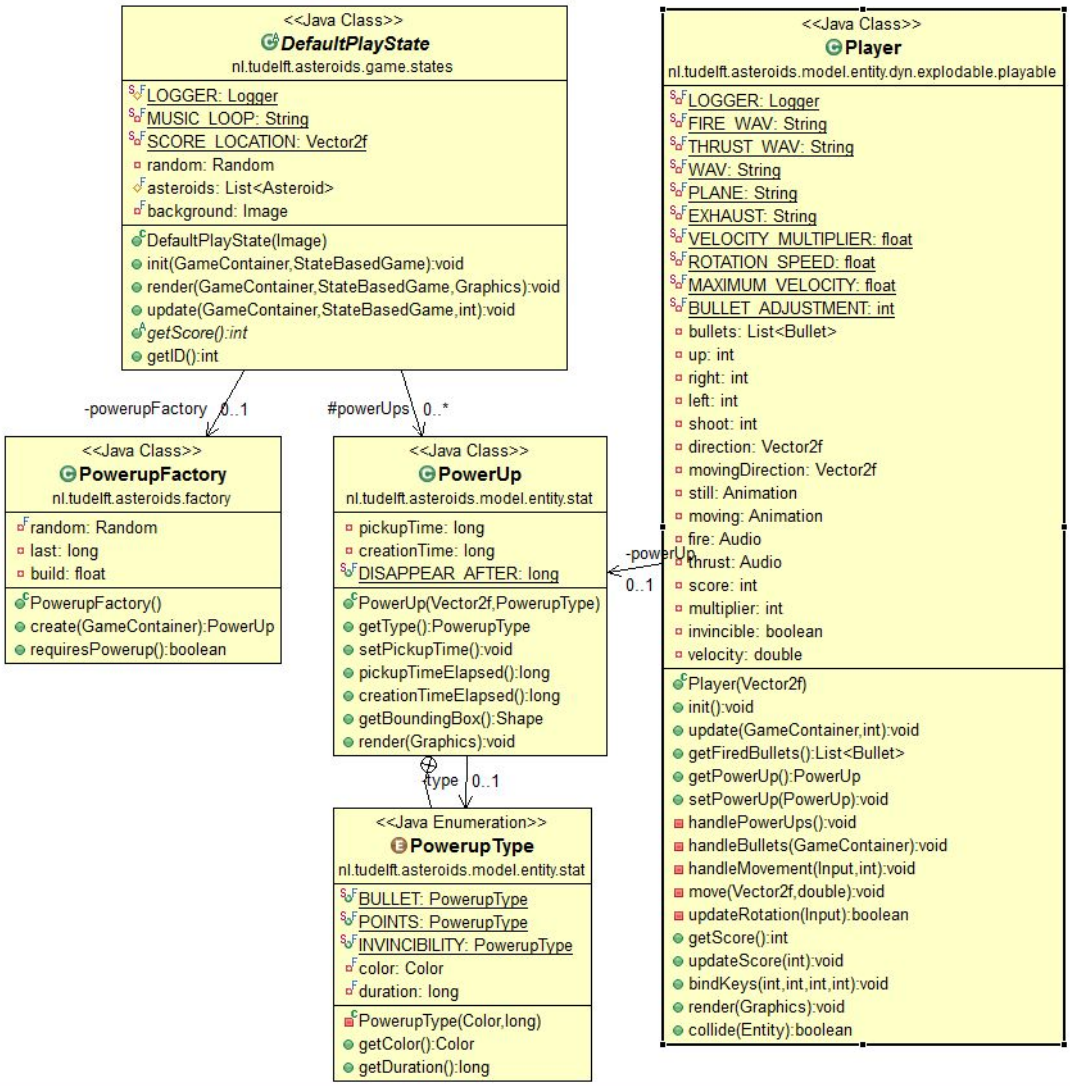
The factory pattern:

The PowerUpFactory class in our code is the factory for the PowerUp class. The PowerUpFactory creates a PowerUp. This PowerUp is then added to a List of PowerUps in the DefaultPlayState. This makes it easier to add new PowerUps to the game. The PowerUpFactory create method is simply used to make a new PowerUp.

2.

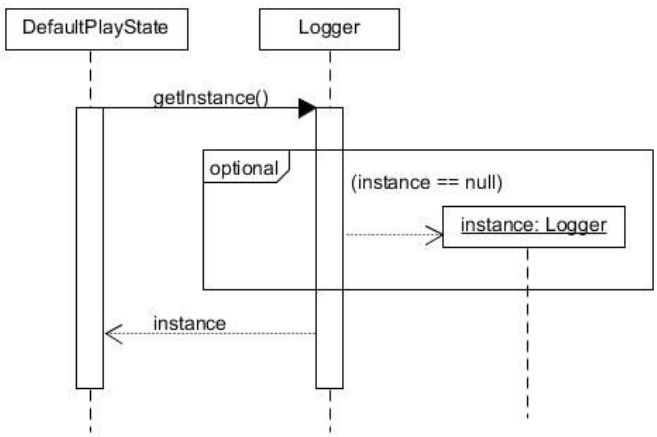


Statical representation of the Singleton design pattern. The Logger class is used in more classes than DefaultPlayState, but this is just to give an example.

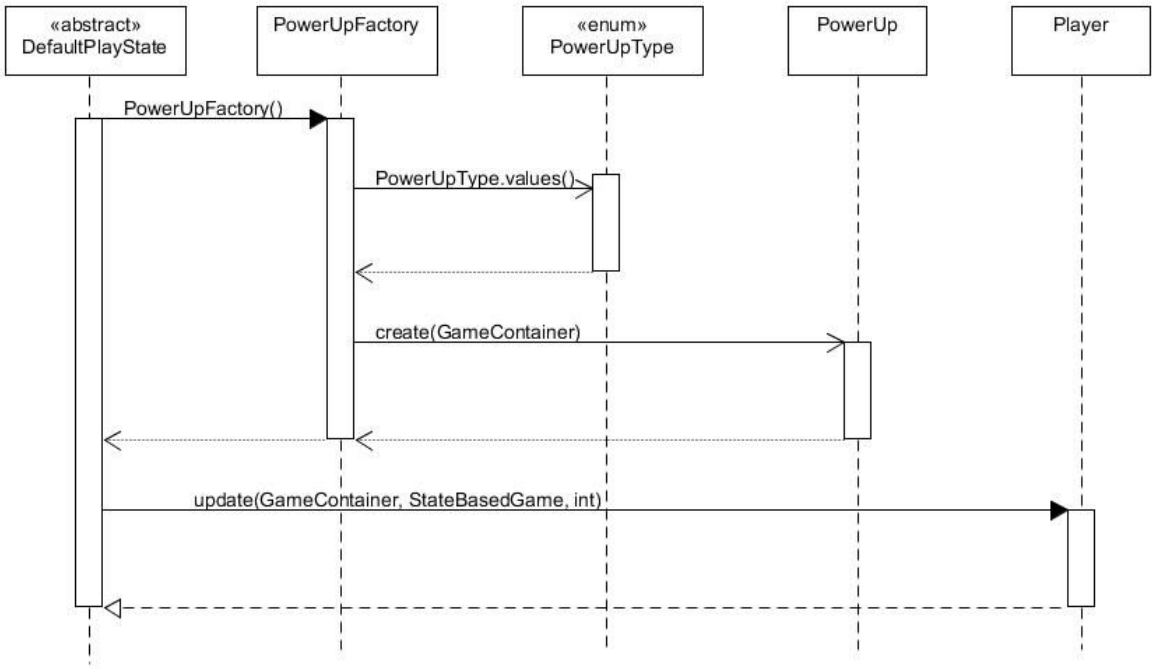


Statical representation of the Factory design pattern

3.



Dynamic representation of the Singleton design pattern.



Dynamicrepresentation of the Factory design pattern

Your wish is my command - power ups

1.

Requirements:

- The game shall display power ups with a powerup sprite
- The game shall spawn a powerup on a random location
- The game shall have 3 different types of powerups
- The game shall despawn power ups after a specified period of time
- The game shall remove the effect of a picked up power up after a specified amount of time

2.

UML

For UML see "*Statical representation of the Factory design pattern".*

CRC

PowerUpFactory

-

-

Create PowerUp	PowerUp element
Spawn algorithm	Time element

PowerUp

Entity

-

Have different types	Type element
Record active time on screen	Time element
Set color of PowerUp	Render element
Return hitbox/bounding box	Bounding box element

20-time - multiplayer

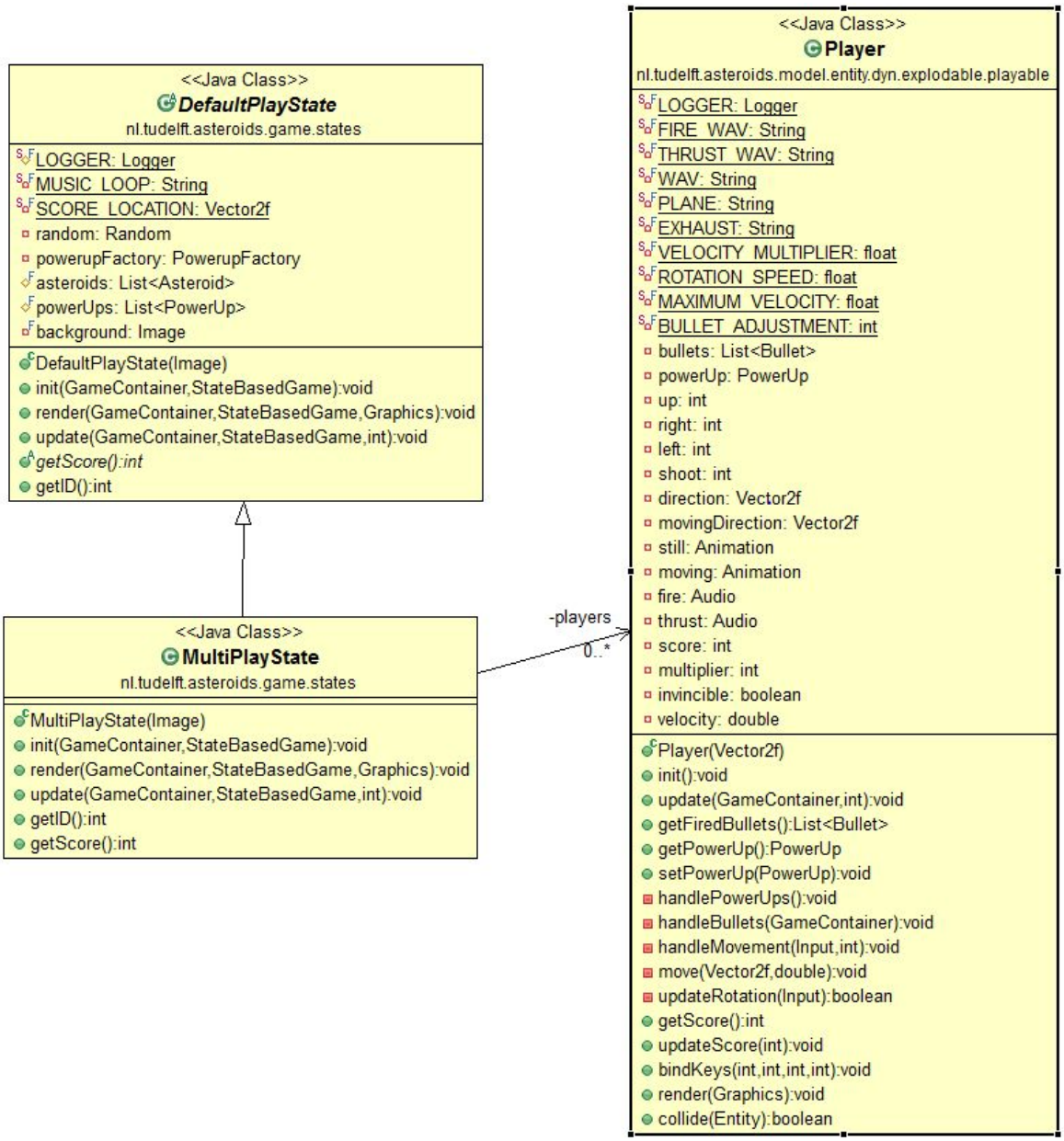
1.

Requirements:

- The game shall be able to have multiple Players
- The Players can be controlled by different users
- The Players can shoot their own Bullets
- The Players have their own score, which is updated by the game
- The Players shall be able to pick up power ups

2.

UML



CRC

MultiPlayState

DefaultPlayState

-

Return score	Score element
Render sprites	Render element
Update Entities in game	Update element

DefaultPlayState

-

MultiPlayState, SinglePlayState

Create PowerUps	PowerUpFactory element
Render Sprites	Sprite/render element
Set background	Background Image element
Update Entities in game (multiple Players)	Update element