# Asteroid Portfolio part 1

# Introduction

The strategy behind this portfolio is to first build the Asteroid game with all its features, these being Split able asteroids, health components and shootable players and enemies as well as an end state even though it was not in the assignment. The Base version is built from the “Asteroids Entity component” version found in the git repo, from week 8’s game lap exercise, I found this to be the most suited base as it was already well prepared for modular design as each “component” was its own separate project.

I felt that I would get the most from the portfolio assignment, by first implementing all of the features from the laps to the base game and then try the different module frameworks, that way I have full separation of concern as I do not need to implement new game features on top of trying to understand the frame work.

I ended up creating one subcommen component called common entity, the reasoning behind this component was that I need the entity player, asteroid and enemy in the collision component to check if they collided. And since a component can only know what is in common I found this to be the best solution.

Controlle skeme is:   
Up - moves the spaceship forward

Left - turns the space ship left

Right - turns the space ship right

Space - shoot a stream of lazers

Enter - resets the game.

# 01 - Service loader.

The service loader was the fastest and simplest of the bunch to get working. This was done with adding a new class named the service loader. This class implements functionality for looking up the different services modeles. For this to work we need to add some meta data as well to each module.   
this is done by adding the follow file structure to each module that depends on common.

\src\main\resources\META-INF\services

Inside here you add files, (easiest done with visual studio code, as it does not care with file extensions)

I needed 3 files named with the fully qualified name convention

* dk.sdu.mmmi.cbse.common.services.IGamePluginService
* dk.sdu.mmmi.cbse.common.services.IEntityProcessingService
* dk.sdu.mmmi.cbse.common.services.IPostEntityProcessingService

inside these files you put the fully qualified names of the classes that implements these interfaces.

And now you are ready to build and start the the game.

# 02 - OSGi - Pax

Changed the project from a netbeans project to a maven project. Also changed the packages deklerations.

# 02 - Netbeans part 1

# 03 - netbeans part 2