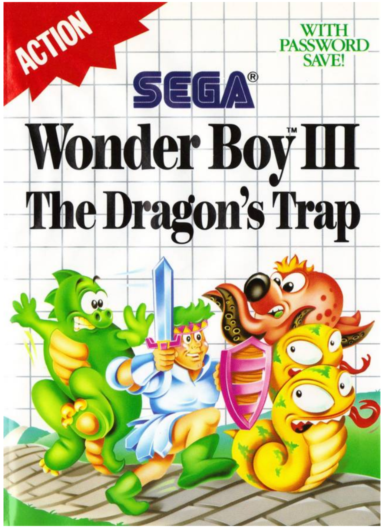
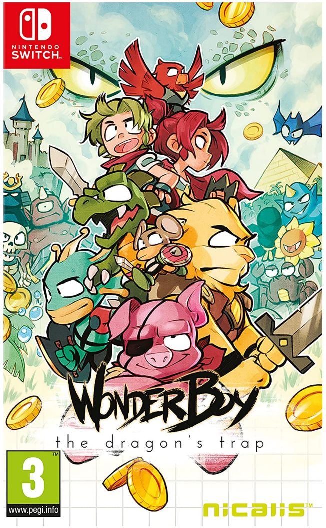
KU LEUVEN

Media Processing

Final Project Details

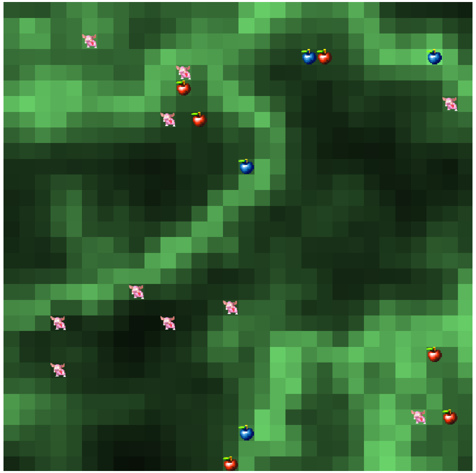
Two views

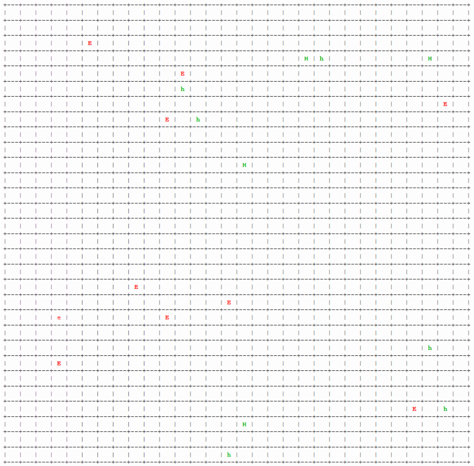

ACTION
WITH PASSWORD SAVE!
SEGA®
Wonder Boy™ III
The Dragon's Trap


NINTENDO SWITCH
WONDER BOY
the dragon's trap
3
www.pegi.info
nicalis™

Two views

- In this case:
    - 2D view
    - Text view

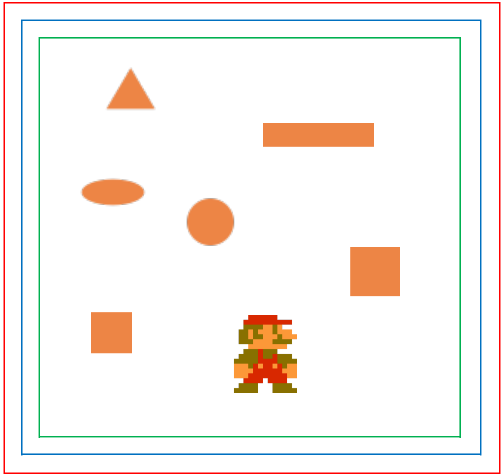




- Make sure you can easily switch between the two while the game is running
    - This means your code structure should support this!

2D View

- Use Qt libraries



Each application runs in a window, typically represented by an instance of the QMainWindow class. When making a new Qt Widgets application, a subclass of QMainwindow is automatically added to the project, and initialized in the main.cpp file

Inside this window you can add one or more QGraphicsView objects. These are essentially 'windows into your scene'. For example, you could have a huge scene of 1000x1000 tiles, of which your QGraphicsView would only show a small fraction

Inside this view you can add a number of things, for this project one of the most interesting options is a QGraphicsScene object. This is a 2D canvas that can contain thousands of 2D objects. This would be what you add your tiles, enemies etc to

Inside the scene you can add objects. You can add anything, as long as it is a subclass of QGraphicsItem. There's a number of predefined subclasses, such as QGraphicsRectItem (a rectangle) and QGraphicsPixmapItem (an image). If you want, you can of course also create your own subclasses

You can use the Qt Designer to build this structure through a graphical tool. However, we recommend that you try building such a structure completely through code at least once (creating, initializing and linking all necessary objects in your source code, without using the graphical designer. You will understand what happens behind the scenes much better that way, and you will need to create objects in code at some points in your project anyway.

Text View

DON'T	DO
Copy-paste your 2D structure with QGraphicsTextItems	Build a string representing your scene Check out QString, QStringBuilder, std::stringstream, …
Print in a separate terminal	Integrate the text view in your MainWindow
Show the full playing field at all times	Show only a subset of your scene if it is very big (e.g. always show at most a 30x30 piece of your scene)

Model-View-Controller

VIEW

Anything in your application that is visual

2D visualization
Text visualization
GUI
…

CONTROLLER

The link between the two
(OR: algorithms and logic)

MODEL

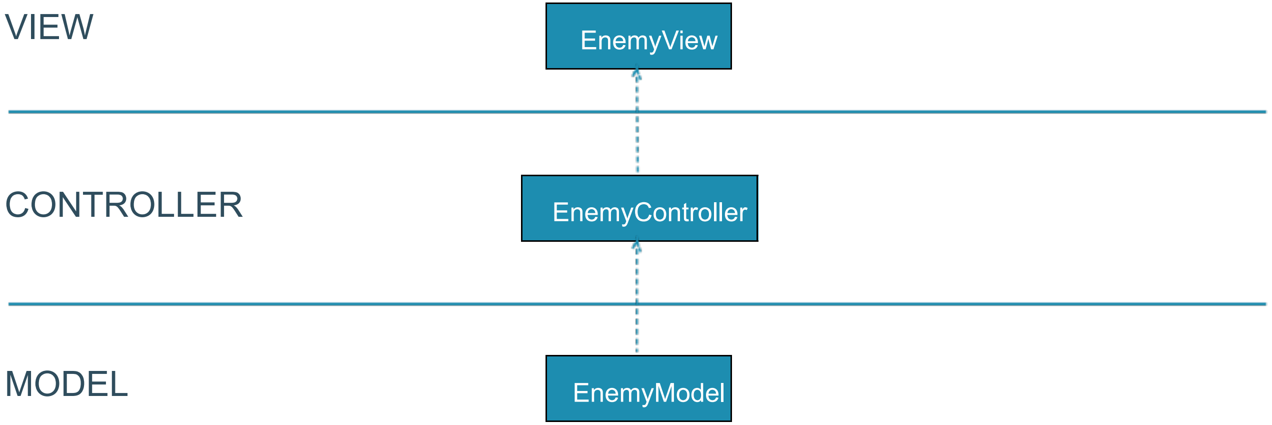
Information about the state of your application

Library classes

Player position
Enemy position and status
World structure
…

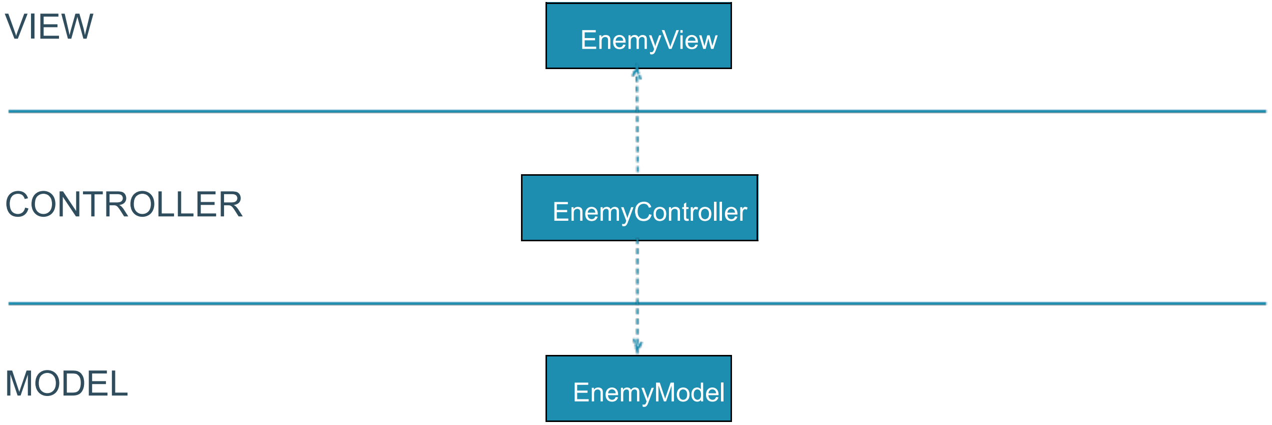
Model-View-Controller

Who informs whom?



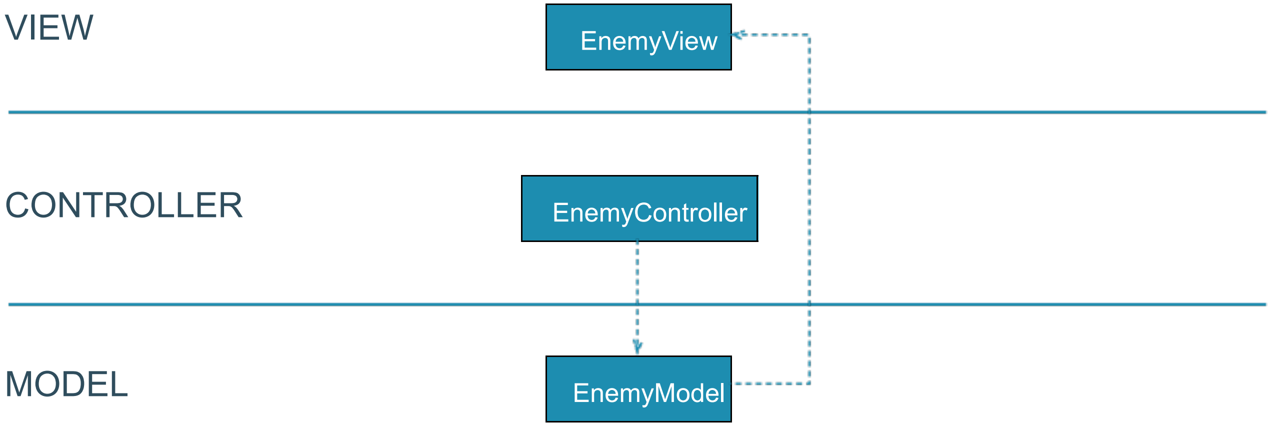
Model-View-Controller

Who informs whom?



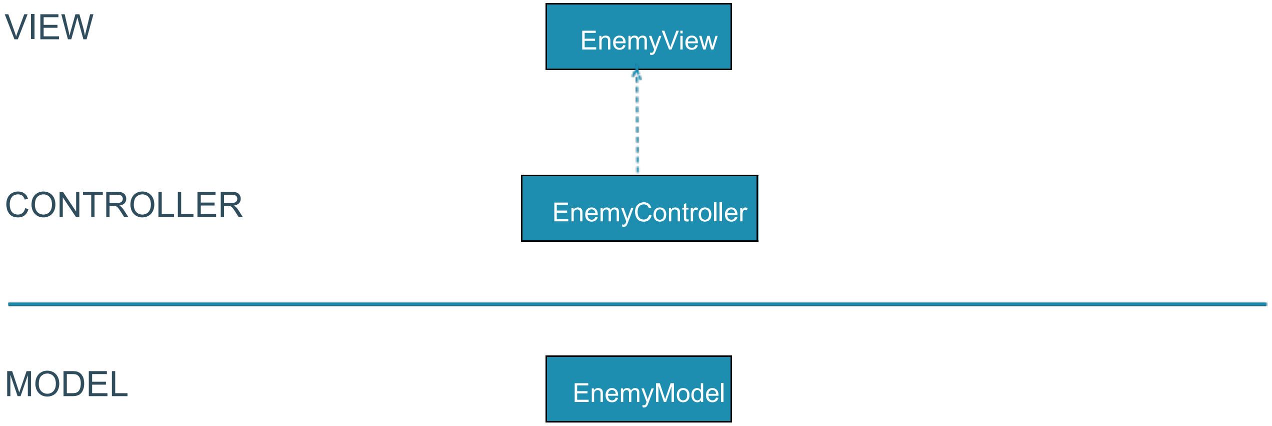
Model-View-Controller

Who informs whom?



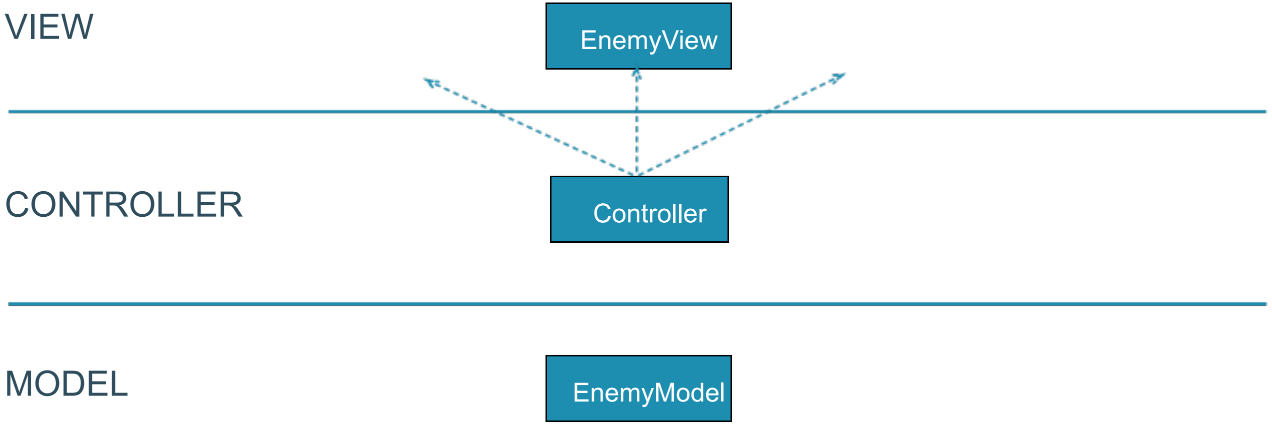
Model-View-Controller

Controller class(es)?



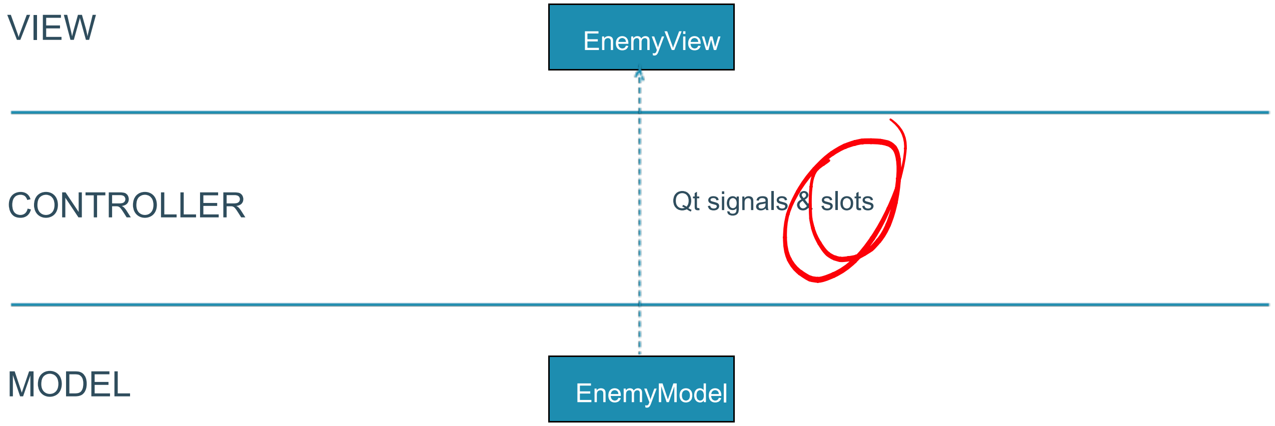
Model-View-Controller

Controller class(es)?



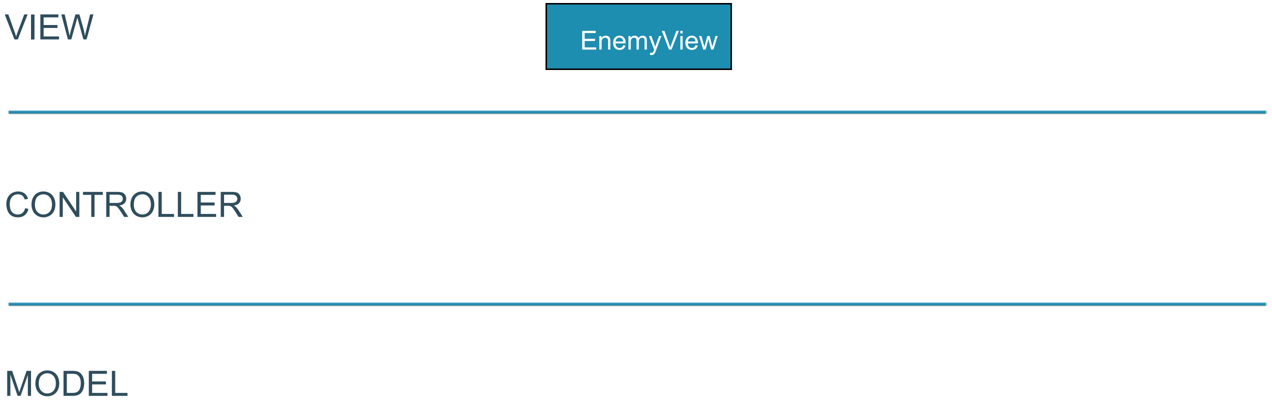
Model-View-Controller

Controller class(es)?



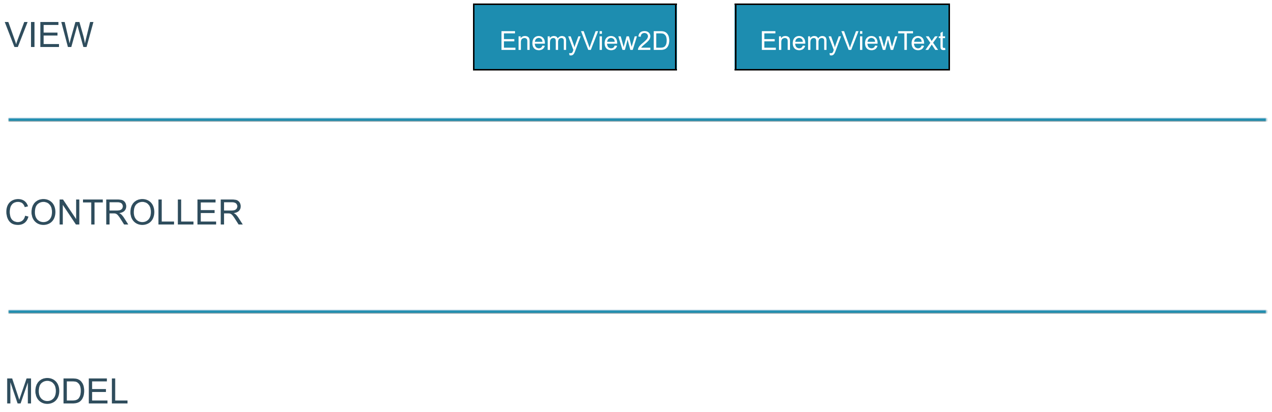
Model-View-Controller

Different views?

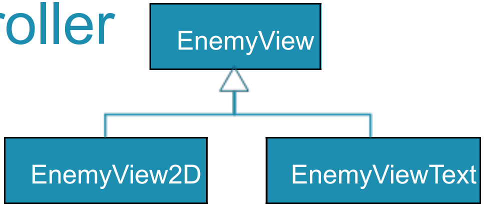


Model-View-Controller

Different views?



Model-View-Controller



Different views?

VIEW

PEnemy?

XEnemy?

Other views?

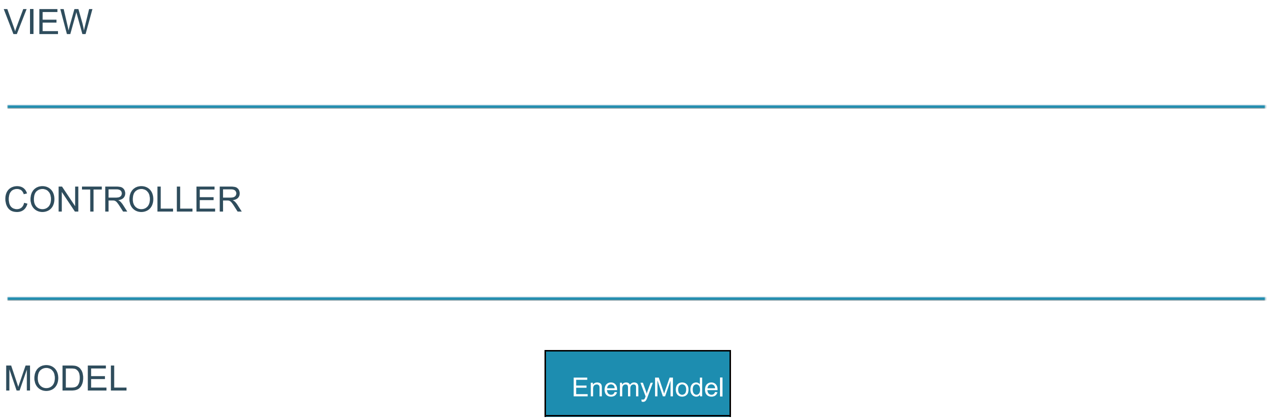
CONTROLLER

How to switch between views?

MODEL

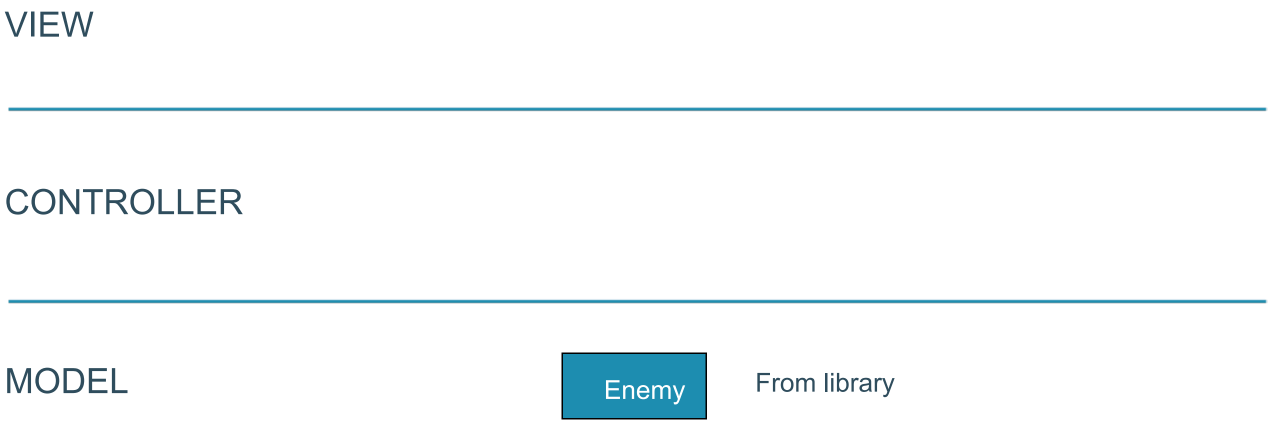
Model-View-Controller

Custom models vs library?



Model-View-Controller

Custom models vs library?



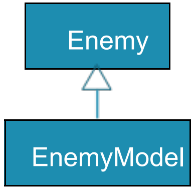
Model-View-Controller

Custom models vs library?

VIEW

CONTROLLER

MODEL



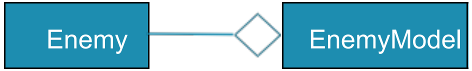
Model-View-Controller

Custom models vs library?

VIEW

CONTROLLER

MODEL



Library

- All classes belong to model layer
  - Probably…
- But model layer isn’t made up of ONLY library classes
  - Probably…
- Note that signals are sent on certain actions
  - Check source code for more details
- Don’t forget you can extend library classes
- But you cannot change the source code
  - It is possible that we will release an updated version of the library later in the semester

Polymorphism

- Never the only solution, but…
- Where will it be used?
- Avoid large if-statements
- Consider extendibility and future-proofing of your code
  - What if extra enemy types?
  - What if animations for player or enemies?
  - What if status effects are added?
  - What if extra views?
  - What if extra collectable items?
  - What if more than one player?
  - …

Pathfinding

- Work iteratively
  - Make a Best-First and Dijkstra first, then adapt it into A*
- Correct
  - Test implementation on a (very!) small map (e.g.3x3) and inspect what happens step by step
  - Draw your path (and your search space!)
- Efficient
  - Use maze3.png to test (2400x2380 pixels = over 5.5 million tiles)
  - it should be possible to find a path in under 1 second

Pathfinding

- Heuristic Weight
  - Use worldmap4.png
- Switch maps
  - Make sure you can choose map from in-game UI
- Visit cost of node
  - Node value or difference between values of current and next node?
- Consider adding fixed step-cost
- Consider reparenting consequences
  - Especially in combination with pointers
- DON'T. COPY. YOUR. NODES.