# Phase 1

## Modelling the game

### The Board

* The game board is modelled as a 2D plane with an X plane (limit given by bWidth) and a Y plane (limit given by bHeight) : minesweeper/minesweeper/src/MinesweeperGame.hs: 37-38
* xy coordinates used to navigate around the board modelled as XYCors type which is just a tuple of Ints: minesweeper/minesweeper/src/MinesweeperGame.hs: 16
* Board itself implemented as a List of Lists of Cell objects AKA a List of rows of Cell objects: minesweeper/minesweeper/src/MinesweeperGame.hs: 17
* A cell is a square/tile on the game board and has a state (VisState), content (mine or empty), and position on the board (coords): minesweeper/minesweeper/src/MinesweeperGame.hs: 19-34
* The game board can be created initially as an empty board (there are no mines anywhere) with boardInit or ready to play straight away with generateMinefield: minesweeper/minesweeper/src/MinesweeperGame.hs: 42-65
* The random generation of mines is done with

### The Game

* Calculating what cells are adjacent and their properties is key to the game running smoothly. Some coordinate geometry revision (credited in the --comments) and logical thinking lead to: minesweeper/minesweeper/src/MinesweeperGame.hs: 104-127
* EndGame:
  + In order to win, a row can have cells that are:

1. Visible and empty (mineless)
2. Hidden and have a mine