Continuing on from last week...

- We didn’t quite finish these, so 2 tasks to get the game up and running!

- 1) Make the zombies follow the player
- 2) Give the zombies health so they can be shot

For any of you new to the club...

- I've hosted the basic code on my GitHub
- Go to: ***github.com/jamesadey/zompy/tree/week2***
  - *This is the project containing finished code from week 1, ready for week 2.*
- Download all the files
  - *There should be a button to **download/clone***
- Open zompy_launcher.py in IDLE
  - Python version 3.x please!
- Run this file, and the game should start…

1) Make zombies follow the player

- A few key questions to ask ourselves…
- How do the zombies know where the player is?
- Once we know where the player is… How do we move towards them?

1) Make zombies follow the player

- A few key questions to ask ourselves…
- How do the zombies know where the player is?
  - How do zombies know about the player?
  - ***Use our Globals for storing the player!***
- Once we know where the player is… How do we move towards them?
  - We want our coordinates to be the same as the player’s coordinates…
  - ***Use if statements!***
  - ***If our player has a higher X coordinate than us, then we want to move in the positive X direction…***
    - Repeat for all 4 directions!

2) Give the zombies health so they can be shot.

- Again, another few key questions…
- How does the zombie know how much health it has?
- How do we know if we hit a zombie?
- How do we damage a zombie?
- How does a zombie know if it’s dead?
  - *On a philosophical note… Can the undead die? Hmm…*

2) Give the zombies health so they can be shot.

- Again, another few key questions…
- How does the zombie know how much health it has?
  - ***Use a variable to remember our health!***
- How do we know if we hit a zombie?
  - This is more tricky… How can we tell if an object is a zombie?
  - ***We can to use a python function to check if what we hit is an instance of a zombie!***
- How do we damage a zombie?
  - ***We need a method on the zombie that we can call (from the player) to tell it to take damage!***
- *How does a zombie know if it's dead?*
  - ***When it's health goes below zero, it is dead.***
  - ***When a zombie dies, remove the zombie from the game!***

Creative Computing. Week 2

(still doing something cool, with computers)

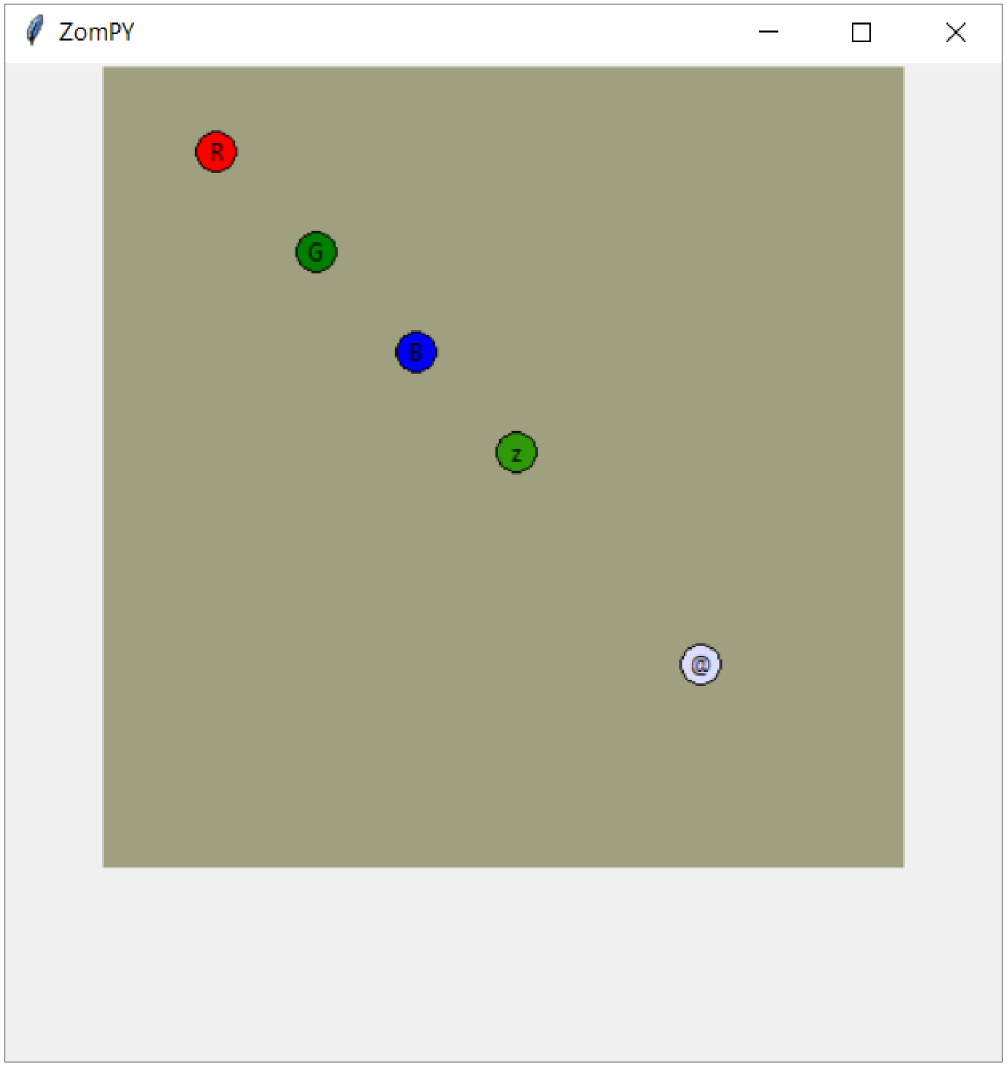
If you don’t have the project yet...

- I’ve hosted the basic code on my GitHub
- Go to: ***github.com/jamesadey/zompy/tree/week2***
  - *This is the project containing finished code from week 1, ready for week 2.*
- Download all the files
  - *There should be a button to* ***download/clone***
- Open zompy_launcher.py in IDLE
  - Python version 3.x please!
- Run this file, and the game should start…

Remember ZomPY?

- Now our zombies can move…
- And they can be shot!
- So… What next?

- Spawners.

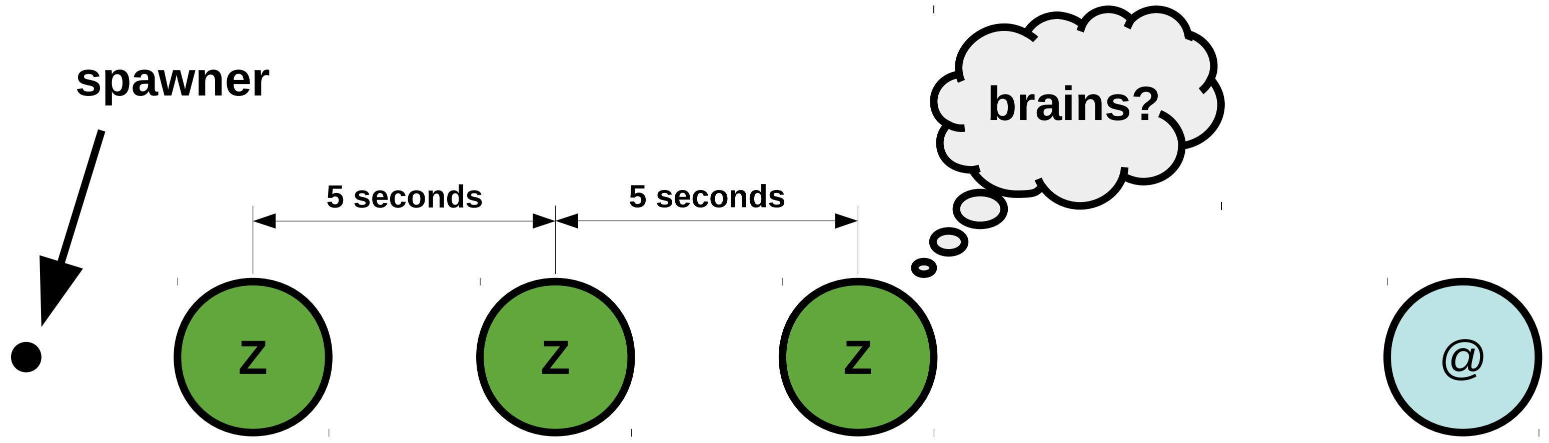


What is a Spawner?

(yep, here’s some theory)

What is a Zombie spawner?

- A zombie spawner is a game object that creates zombies!
- The spawner we'll create today will make new zombies at regular intervals.



Spawners aren’t just for zombies…

- Almost all games use some form of spawner.
- They’re good ways of creating new objects at “runtime”
- Item boxes (think mario-kart)
  - Could be created by one big item spawner.
  - Or… They might take a distributed approach…
- Random loot spawners
  - Think of your favourite battle-royale game, plenty of random loot spawns there…
- NPC Spawners
  - Think of an MMO, those npc groups that keep coming back so you can “farm” them for XP.

Distributed spawners?

- Instead of having all the responsibility piled into one separate object…
- Why not share it around all the instances of that class?

- Objects then control how they respawn (if at all!)
- Some objects don't need to respawn, but just "hide" temporarily until they re-appear. Maybe destroying and creating a new one is overkill?

- Why?
  - It's a different style of programming.
  - It's much better for certain kinds of objects
    - Think item boxes in MarioKart
    - Also works for players.
  - Sometimes whether or not an object spawns is dependent on the local conditions.

The Fun Part.

(actually coding stuff)

This is a big task

- It seems simple. But will require a lot of code.
- So let’s split it into parts!
  - *Here we’ll be focussing on a creating single spawner, as opposed to a distributed approach*

- 1) Design the spawner.
- 2) Plan the code.
- 3) Code the spawner.

1) Design a zombie spawner

- A few questions to ask ourselves…
  - *There are no right or wrong answers to these ones…*
- How many zombies should it spawn?
  - Is there a maximum?
- When should it spawn them?
- Does it need to keep track of the spawned zombies?
  - If so, do zombies need to know about their spawner?

2) Plan the code.

- *How are we going to actually implement this?*
- How many zombies should it spawn?

- When should it spawn them?

- Does it need to keep track of the spawned zombies?

- Do zombies need to know about their spawner?

3) Code the spawner.

- How many zombies should it spawn?
- When should it spawn them?
- Does it need to keep track of the spawned zombies?
- Do zombies need to know about their spawner?

3) Code the spawner.

- How many zombies should it spawn?
  - Infinite, or limited… This is entirely up to you.
  - ***Either way… store it in a variable!***
- When should it spawn them?
  - We need a timer…
  - ***Use our globals to get the current time, and plan future times in which we'll spawn zombies!***
- Does it need to keep track of the spawned zombies?
  - ***If so, we could use a list…***
- Do zombies need to know about their spawner?
  - ***When we create the zombie, we could tell it which spawner created it…***
  - ***If the spawner knows about us… we need to notify it when we've died.***

Spawning forward...

- As usual, spawners don't end here, we could extend them indefinitely. Here's a few ideas to get you started...

- Can different spawners have different rates?
- Could zombies come in waves?
- Could a spawner move? Or spawn zombies randomly?