Where did we get with ammo boxes?

- We didn't manage to finish integrating ammo boxes before half-term.
- For the first 10 minute let's get everyone caught up with working item boxes.
- If your item boxes already work, then have a go at adding some ammo boxes!

- Instructions for these can be found here:

github.com/JamesAdey/zompy/blob/master/slides/cc_week3.pdf

Creative Computing. Week 4

(making computers do cool stuff)

If you don't have the project yet...

- I've hosted the basic code on my GitHub

- Go to: ***github.com/jamesadey/zompy/tree/week4***
  - *This is the project containing finished code from week 3, ready for week 4.*
- 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…

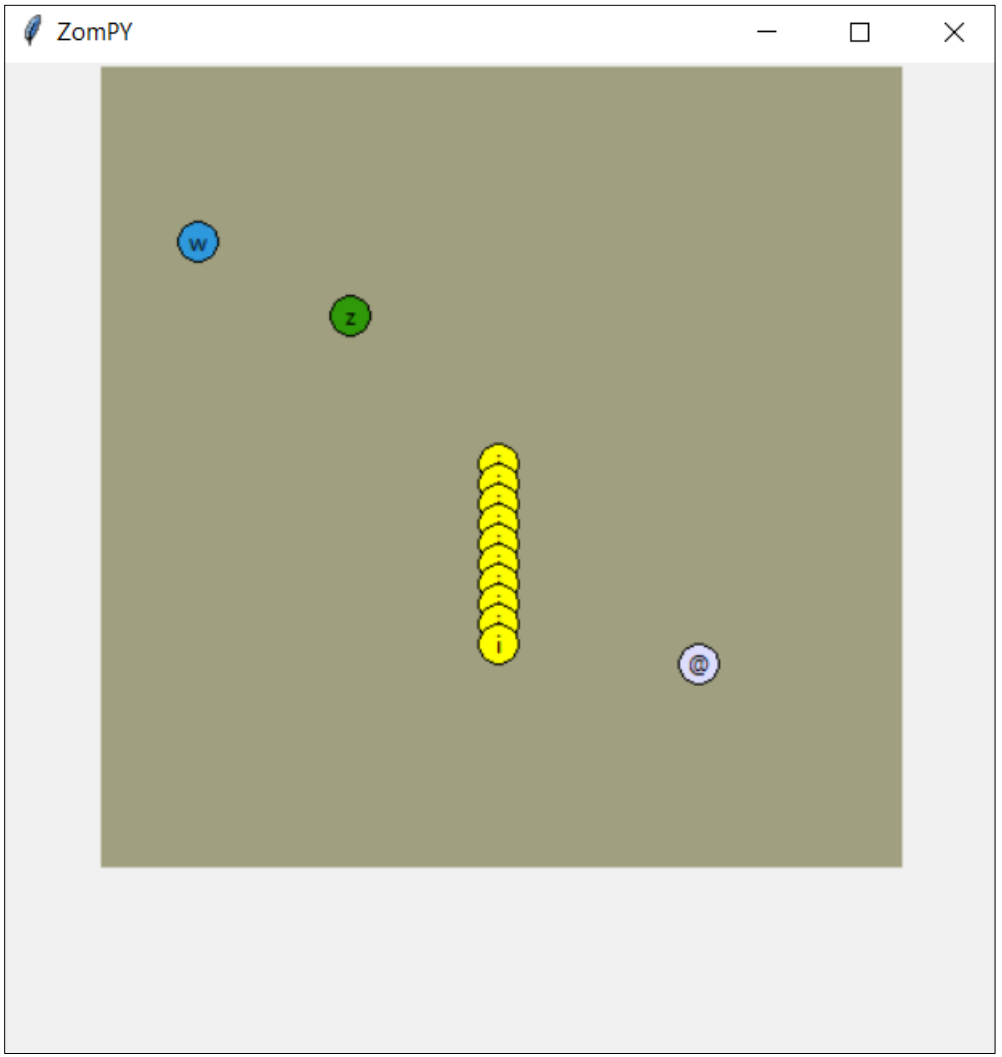
What next?

- Our zombies can move…
- And they can be shot…
- We have spawners…
- And item pickups!

Improved Pathfinding!

- Our zombies can move…
- And they can be shot…
- We have spawners…
- And item pickups!

- Now it's time for some smarter pathfinding!



Advanced Pathfinding.

(giving zombies more brains...)

What we have so far...

- Our zombies are dumb…
  - They just walk in (roughly) straight lines to the player
- They ignore each other, and items placed down.
  - Avoiding each other is tricky, but it would be nice to get them to avoid certain obstacles in the game.

Giving zombies a brain.

- We can use the same "local" movement
  - Telling it which direction to walk in, and actually moving the zombie
- But change the "global" movement to be following a path.
  - A path here is just a sequence of short lines
- We need something to represent the walkable areas, so let's split the world into a grid of cells!

Pathfinding Issues

- Firstly, pathfinding is a computationally expensive operation.
- How do we get from the zombie to the player?
  - Simple. Find a path from the zombie to the player
- What about lots of zombies, won't their paths overlap?
  - Yes they will. (It's probably not an issue for a project of our scale)
  - Can we improve this, re-using paths perhaps?

…Thinking Differently

- What about asking the opposite question…
- ***How do we get from the player to the zombies?***
  - We could search outwards from the player looking for zombies.
  - During the search we can mark all the cells visited with a pointer back to previous one in the search.
- **Then backtrack!**
  - This will make a chain of "signs" leading back to where the player started.
  - Then all a zombie needs to do is read the nearest sign and it will be told where to go!

Dijkstra's Algorithm!

- The navgrid code uses a modified version of Dijkstra's Algorithm. This is a graph search algorithm.
- I'm not going to teach that here as it's A-Level content.
  - Feel free to look it up in your spare time though!
- There is no goal, it just continues until all cells have been explored, marking the shortest path to each one.

The Fun Part.

(actually coding stuff)

Download these slides.

github.com/JamesAdey/zompy/blob/master/slides/cc_week4.pdf

Integrating the NavGrid

- I've provided you with the code for the NavGrid, as well as a sample navgrid follower.
- You'll need to download these files here and add them into the project:

  github.com/JamesAdey/zompy/blob/week4/navgrid.py

  github.com/JamesAdey/zompy/blob/week4/zombie_gridwalker.py

- I've included instructions in the following slides to integrate these with the existing game.

Edit: *zompy_globals.py*

- *Don't forget the import statements!*
- *We need to add the navgrid to our global variables so we can access it from anywhere.*
- ***Inside the setup_game function:***
- *First create the navgrid.*
- *Then create a grid walker to follow it.*
- *Finally create a line of items to test the pathfinding.*

```
from item_manager import *
from example_item import *
from navgrid import *
from zombie_gridwalker import *

class ZompyGlobals(GameGlobals):
    zoms = 10
    bulletManager = None
    itemManager = None
    gameWorld = None
    navGrid = None
    player = None

class ZompyEngine(GameEngine):

    def __init__(self):
```

```
# create a zombie spawner
zs = ZombieSpawner(x=100,y=100)
super().add_game_object(zs)

# create a nav grid
ng = NavGrid(resolution=20)
super().add_game_object(ng)
gGlobals.navGrid = ng

# create a gridwalker
gw = ZombieGridwalker(x=50,y=50)
super().add_game_object(gw)

# create a line of items
for i in range(10):
    it = ExampleItem(x=200,y=200+(i*10))
    super().add_game_object(it)



pl = Player(x=300,y=300)
gGlobals.player = pl
```

Edit: *example_item.py*

- *Add a variable to store the navigation cost*

```
colour = "#ffff00"
char = 'i'

navCost = 5

m_ovalId = None
m_textId = None
```

- *Edit the **on_start** function to add this item as a navigation blocker.*

```
        self.char = char

    def on_start(self,gameGlobals):
        # register ourselves with the gameWorld as an item
        gameGlobals.itemManager.add_item(self)
        gameGlobals.navGrid.add_blocker(self)

    def on_remove(self, gameGlobals):
        # remove ourselves from the list of items
        gameGlobals.itemManager.remove_item(self)
        gameGlobals.navGrid.remove_blocker(self)

    def setup_gfx(self, tkCanvas):
        x0 = self.x - self.radius
```

- *Remember to edit the **on_remove** function to remove the navigation blocker!*

- *We also need to add the **get_nav_cost** function that is required by all blockers used with the navgrid.*

```
        gameGlobals.engine.remove_gam

    def get_collision_radius(self):
        return self.radius

    def get_nav_cost(self):
        return self.navCost
```

Free Play & Extensions.

- Now it's unguided work! But here's some suggestions:
- **Integrate the pathfinding with the old zombies.**
  - The gridwalker isn't currently able to be shot; take this movement code as an example and integrate it with the existing zombies.
- **Add obstacles that damage zombies and players**
  - This may require adding some kind of "status effect" to the player. Or just damaging them whilst they're on it (maybe a timer could be used here?).

Feedback

- How was this week's session compared to the previous?
- Are the code examples included on the slides helpful?

- We have 4 weeks left, and there's still more I want to cover.
  - ~~Pathfinding~~
  - Levels/Maps
  - Scoring
  - Special Zombies