Did we get the health working?

- The last task for last week was getting the player's health working. We really need this working before we get to this week's content.

- So… open up the slides and project from last week and continue where you left off!

github.com/JamesAdey/zompy/blob/master/slides/cc_week6.pdf

Creative Computing.
Week 7

(stuffing computing with cool)

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

- I've hosted the basic code on my GitHub

- Go to: ***github.com/jamesadey/zompy/tree/week7***
  - *This is the project containing finished code from week 6, ready for week 7.*
- 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…

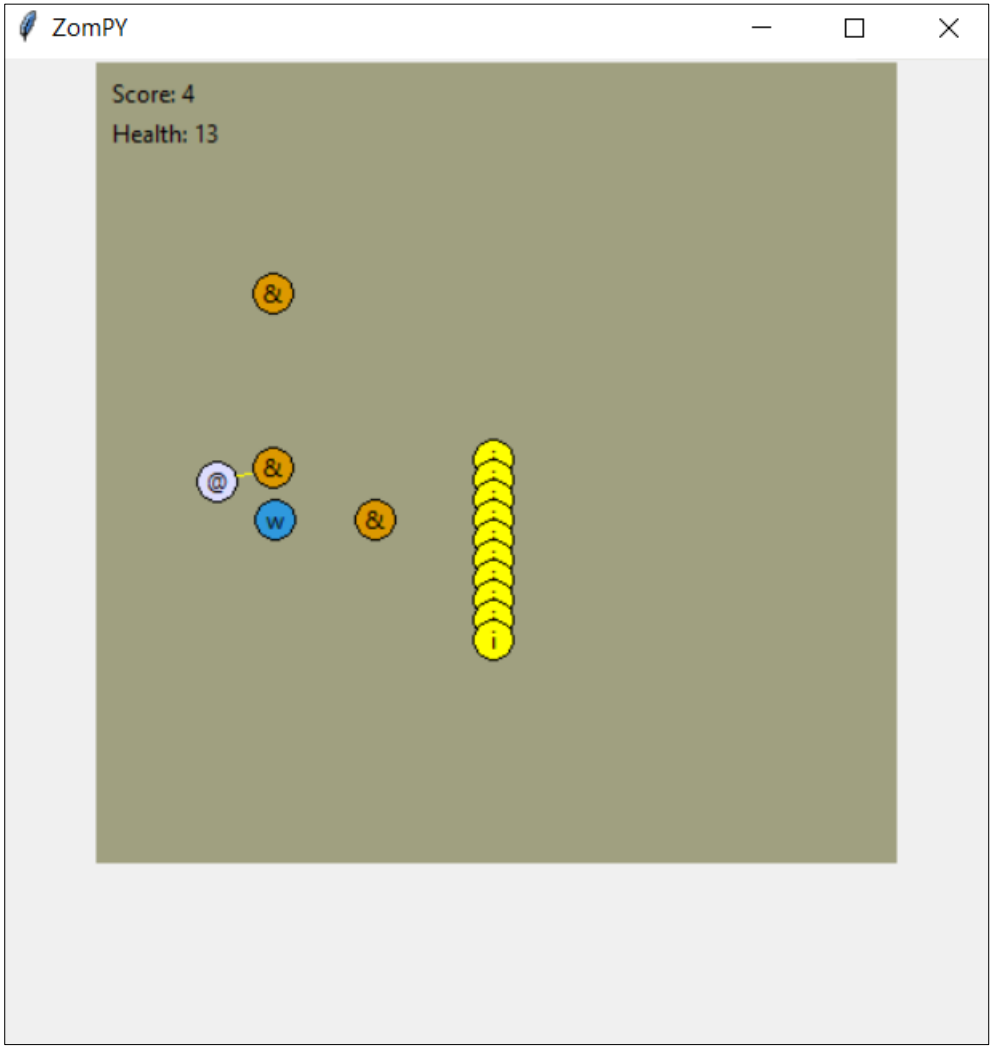
We have a lot so far… What next?

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

Special Zombies!

- Our zombies can move… intelligently!
- And they can be shot…
- We have spawners…
- And item pickups!
- With random maps!
- And UI!

- Special Zombies!



Special Zombies.

(with object oriented programming!)

Introducing OOP

- If you haven't noticed already, we've been using Object Oriented Programming in our game.
- Most of the stuff in our game is an object.
  - Zombies, players, obstacles
- The engine deals purely with objects.
- Each object defines it's own data members and behaviour.
- We'll be using Object Oriented Programming to speed up creating some very special zombies.

2 Methods of customisation

- Difference in scale.
- Works the same, just has different values.
  - e.g. all cars are cars, they all have 4 wheels, but the exact engine power and top speed differs
- Bigger/Faster/Stronger zombies

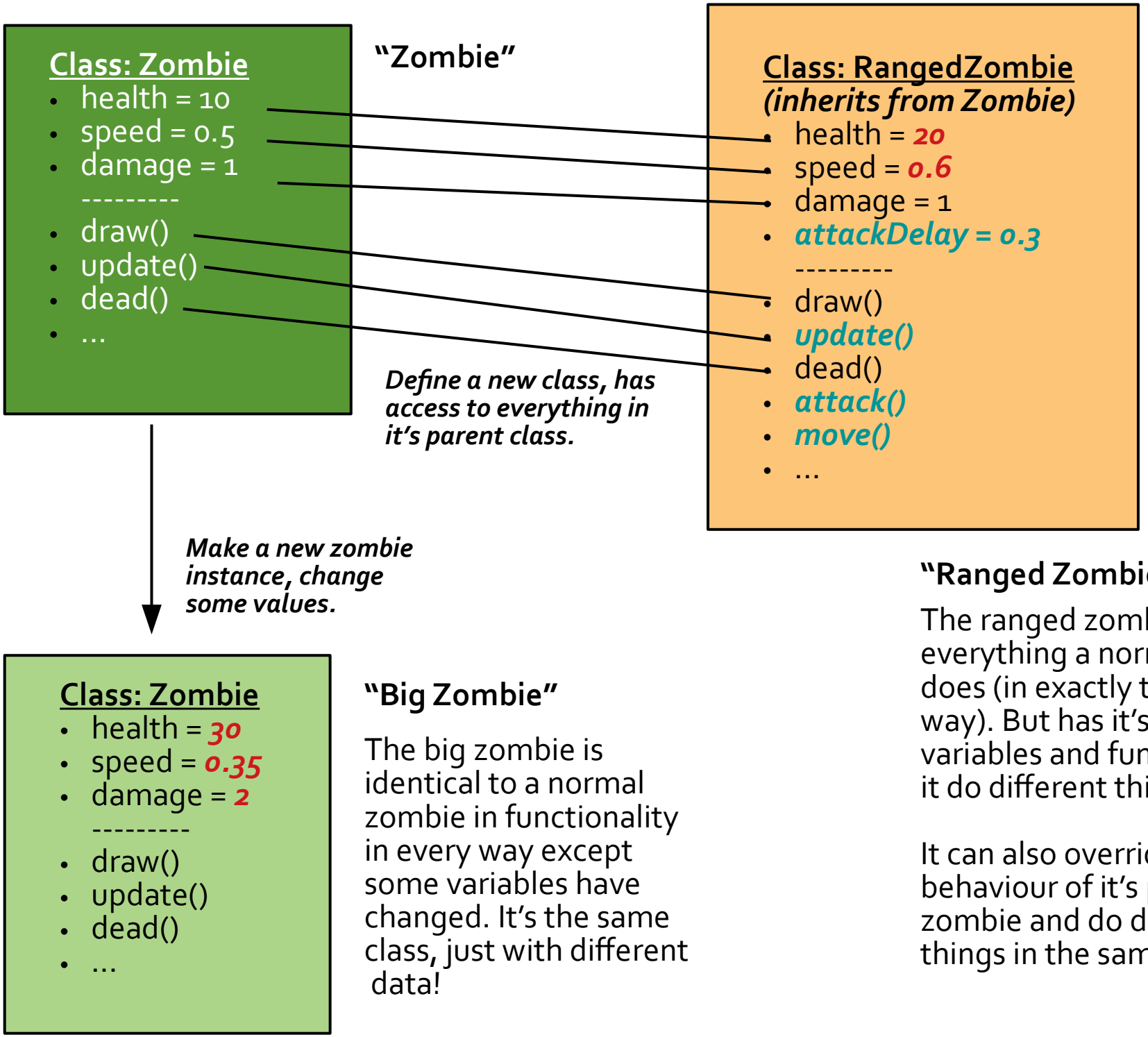
- Difference in kind.
- Is functionally / physically different to other objects.
  - e.g. a lorry is different to a car, both have wheels and can drive, but do so in different ways.
- Exploding / splitting / ranged zombies

Implementing differences in scale.

- We can implement this by using the same object as a template but filling it with different values.
  - e.g. make a tough zombie by simply making a normal zombie and then increasing it's health
- We use an initialisation method to set the specific values for a zombie.
  - We could also do it in the constructor…

Creating differences in kind.

- Most differences in kind are subtle changes, but share many traits with similar objects.
- We will do this through "inheritance". The child class inherits all the attributes and functionality of their parent (super) class.
  - All our objects inherit from the parent class of GameObject
- We can then override specific methods in each child class to provide different behaviours as well as creating new behaviours.



"Big Zombie"

The big zombie is identical to a normal zombie in functionality in every way except some variables have changed. It's the same class, just with different data!

"Ranged Zombie"

The ranged zombie can do everything a normal zombie does (in exactly the same way). But has it's own extra variables and functions to let it do different things.

It can also override the behaviour of it's parent zombie and do different things in the same functions.

The Fun Part.

(actually coding stuff)

Download these slides.

github.com/JamesAdey/zompy/blob/master/slides/cc_week7.pdf

First, add zombie_ranged.py

- Here is an example of a difference in kind, it's still a zombie but attacks at range now.
- 1 new file to add to the project:

  github.com/JamesAdey/zompy/blob/week7/zombie_ranged.py

- Then follow the instructions in the following slides.

Edit: *zombie.py*

- ***Add a variable*** *to track how much damage this zombie does.*

```
# zombie stats
health = 10
speed = 0.5
damage = 1

radius = 10
```

- *Then edit the* ***on_collision()*** *function to deal our damage (instead of just 1)*

```
def on_collision(self, gameGlobals, other):
    if(isinstance(other, player.Player)):
        other.take_damage(self.damage)
```

- ***Create a new function*** *called* ***set_stats()****, that will set the stats and properties of this zombie.*

```
        other.take_damage(self.damage)

def set_stats(self,speed=0.5,radius=10,health=10,damage=1):
    self.speed = speed
    self.radius = radius
    self.health = health
    self.damage = damage
```

Edit: *zombie_spawner.py*

- *Import zombie_ranged.py*

```
from zombie import *
from zombie_ranged import *

class ZombieSpawner(GameObject):

    nextSpawnTime = None
```

- ***Create a spawn_big_zombie function*** *to create a larger Zombie (difference in scale)*

```
def spawn_big_zombie(self,gGlobals):
    zo = Zombie(x=self.x, y=self.y, colour="#9fff99", char='Z')
    zo.set_stats(speed=0.35,radius=15,health=30,damage=2)
    gGlobals.engine.add_game_object(zo)
```

- ***Create a spawn_ranged_zombie function*** *to create a RangedZombie (difference in kind)*

```
def spawn_ranged_zombie(self,gGlobals):
    zo = RangedZombie(x=self.x, y=self.y, colour="#DD9900", char='&')
    zo.set_stats(speed=0.6,radius=10,health=20,damage=1)
    gGlobals.engine.add_game_object(zo)
```

Spawning the new zombies.

- *These new zombies won't spawn in by default, you'll need to **edit the spawner** or **create them as part of a level.***
- *Editing zombie_spawner.py:*

```
def update(self, gGlobals):

    # check if the next spawn time has elapsed
    if(gGlobals.realTime > self.nextSpawnTime):
        # delay the next spawn by the given delay
        self.nextSpawnTime = gGlobals.realTime + self.spawnDelay
        # spawn a new zombie
        self.spawn_ranged_zombie(gGlobals)
```

This is the line you'll need to edit

- *You might even want to make this random, so it picks a random zombie every time.*

Making your own difference.

- That's the end of the guided part of this session, just an example of differences in kind vs differences in scale.
- **Tasks to do:**
- *Can you add a fast runner zombie but with low hp?*
- *Can you add a splitter zombie that splits into 2 normal zombies when it dies?*
- *Can you make it so special zombies give more points for being killed?*
- *Can you merge the ZombieGridwalker into the Zombie, to make all the zombies now properly avoid obstacles?*

Feedback

- We now have a complete working prototype, what did you think of the project?

- We have 1 week left until the end of my university term and I've covered all I want to. What do you guys want to do next week?
  - ~~Pathfinding~~
  - ~~Levels/Maps~~
  - ~~Scoring~~
  - ~~Special Zombies~~
  - ?????