UNIVERSITY OF Hull

Game Structure

Wrestling with Python

Chased By Crackers

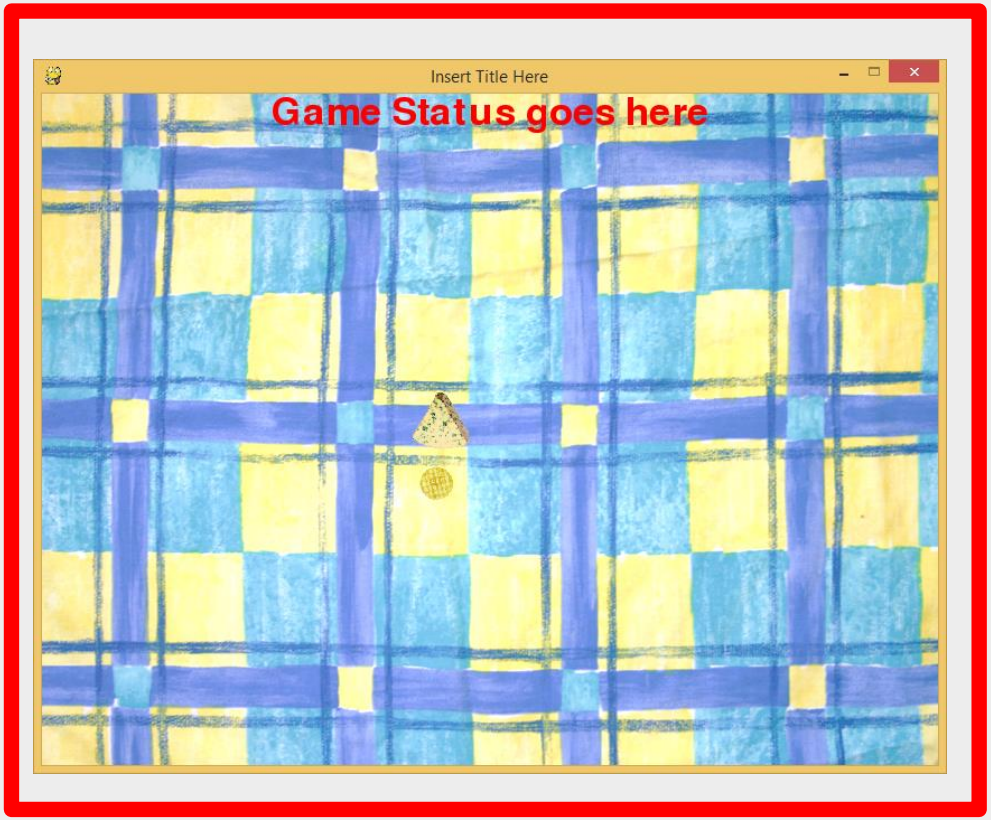
- This is how the game appears when it runs
- A piece of cheese is being chased by a cracker

Game Objects

- From a software point of view there are four objects in this solution
- Can you spot them?

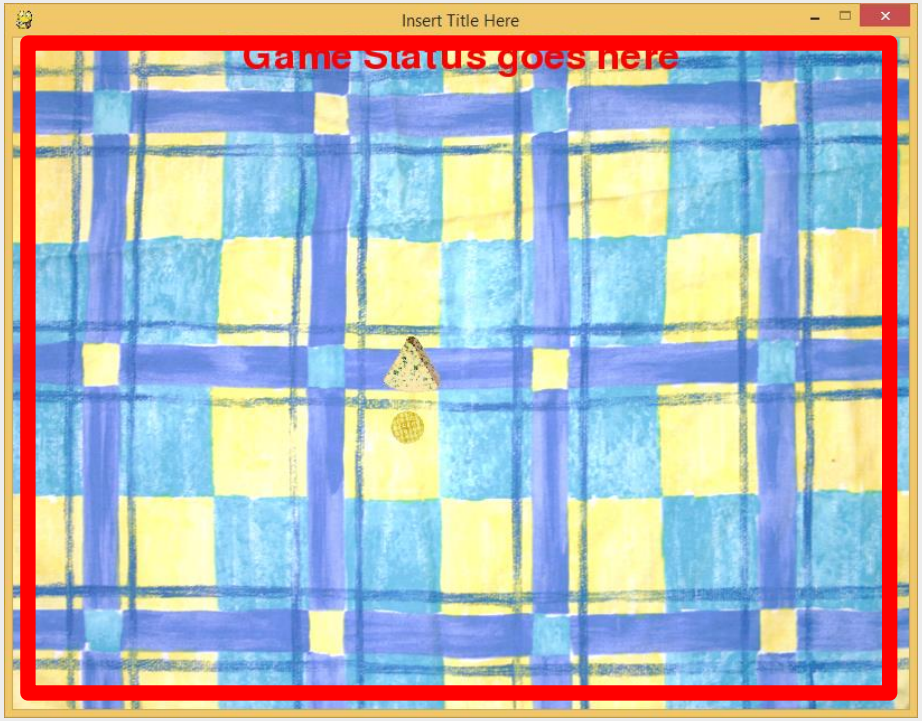
The ChasedByCrackers Game

- The game brings together all the other elements
- It serves as a kind of "container" for all the other game elements



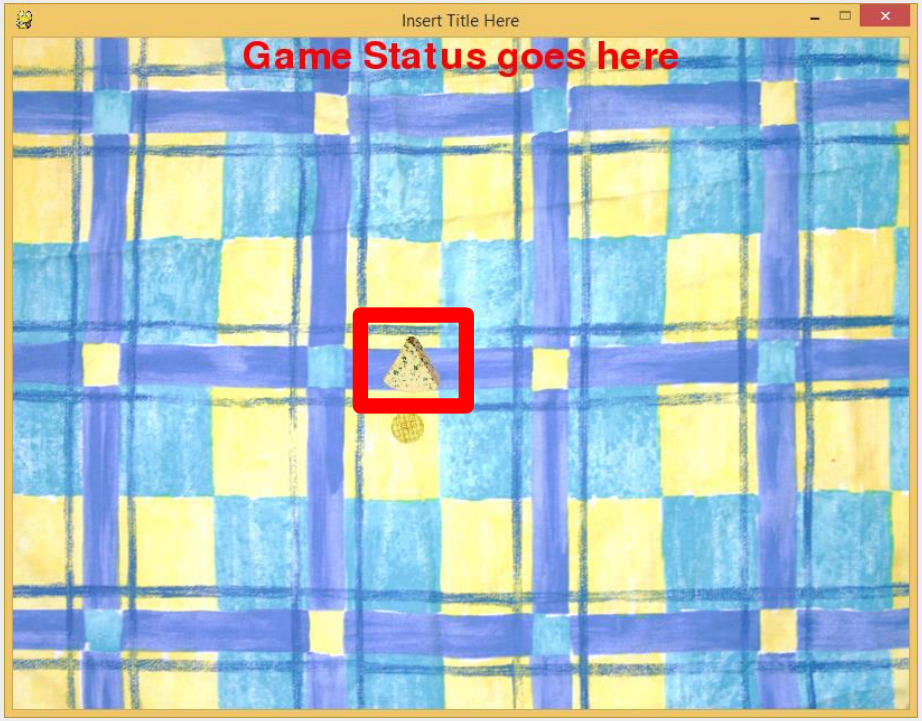
The Background

- The background holds the image for the background of the screen
- It draws itself when asked



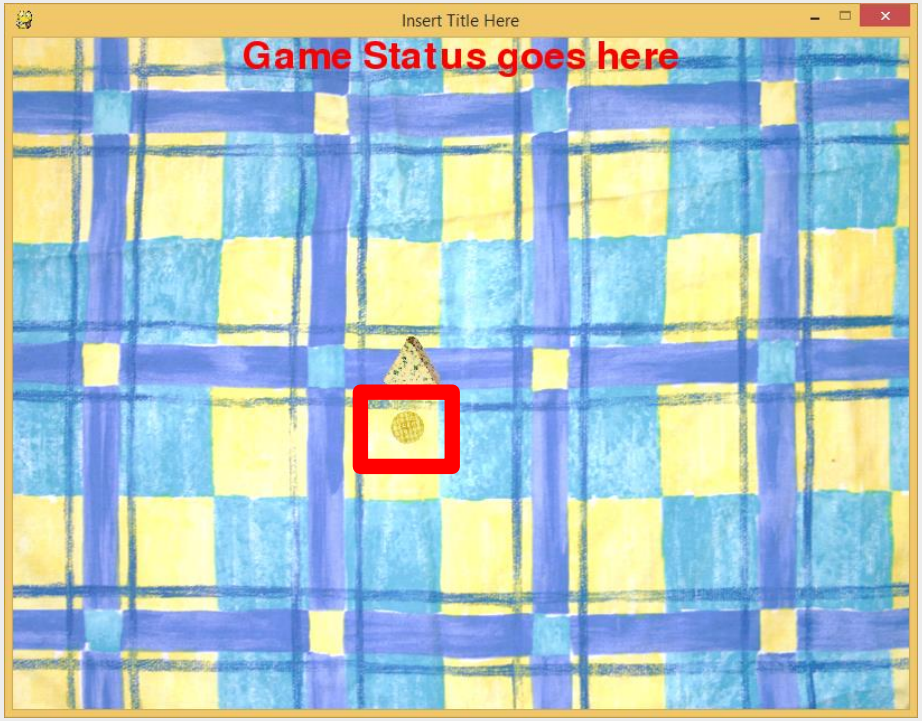
The Cheese

- This object can bounce around the screen
- The game tells it which direction to move in, based on the keys pressed by the user



The Cracker

- This object moves around the screen chasing a target
- The game tells it which target to chase and each time it is updated it moves towards the target



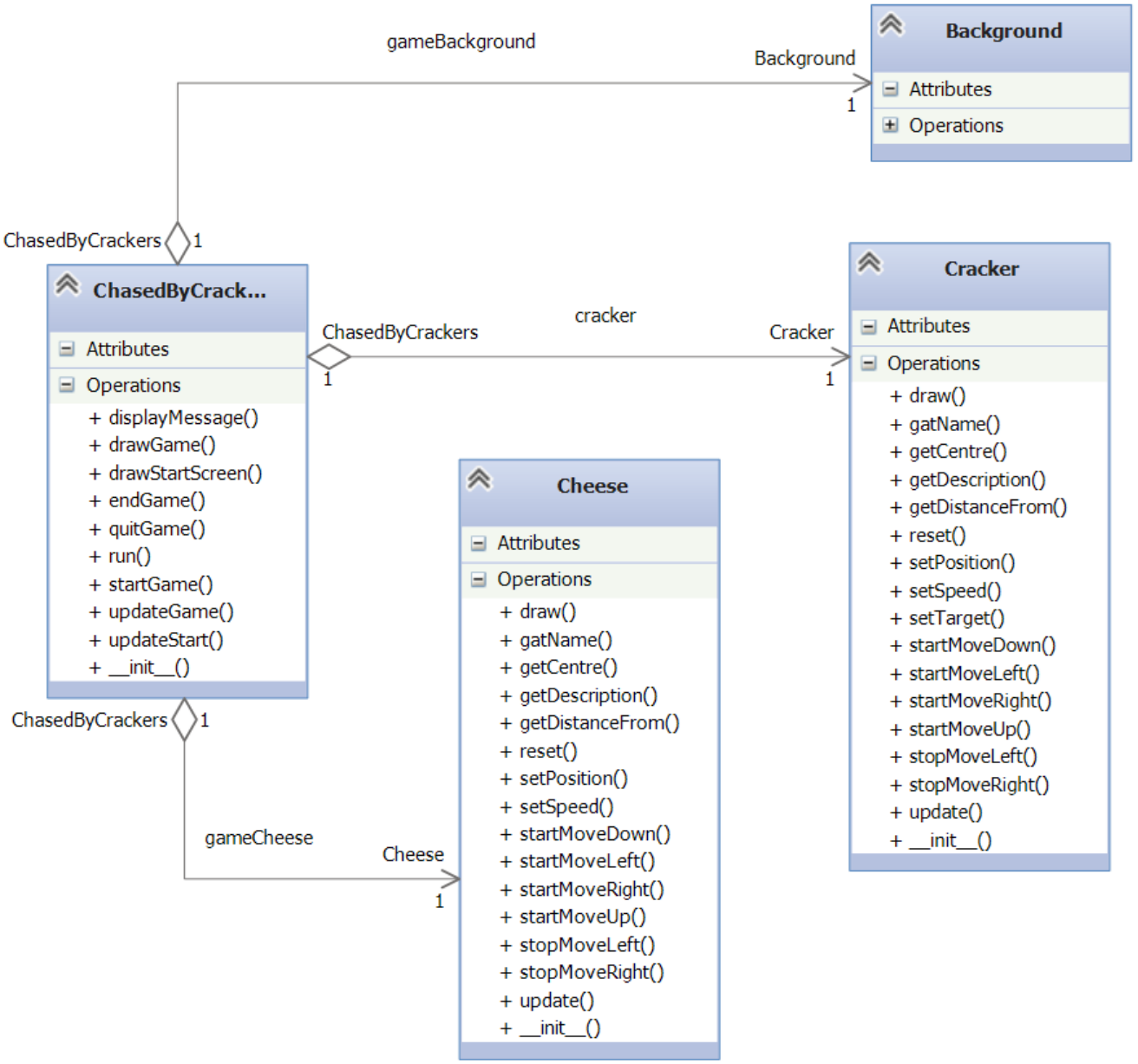
Starting the Game

```
g = ChasedByCrackers()
g.run()
```

- The game is started by a program creating an instance of `ChasedByCrackers`
- Then `run` method is called on that instance and the game begins

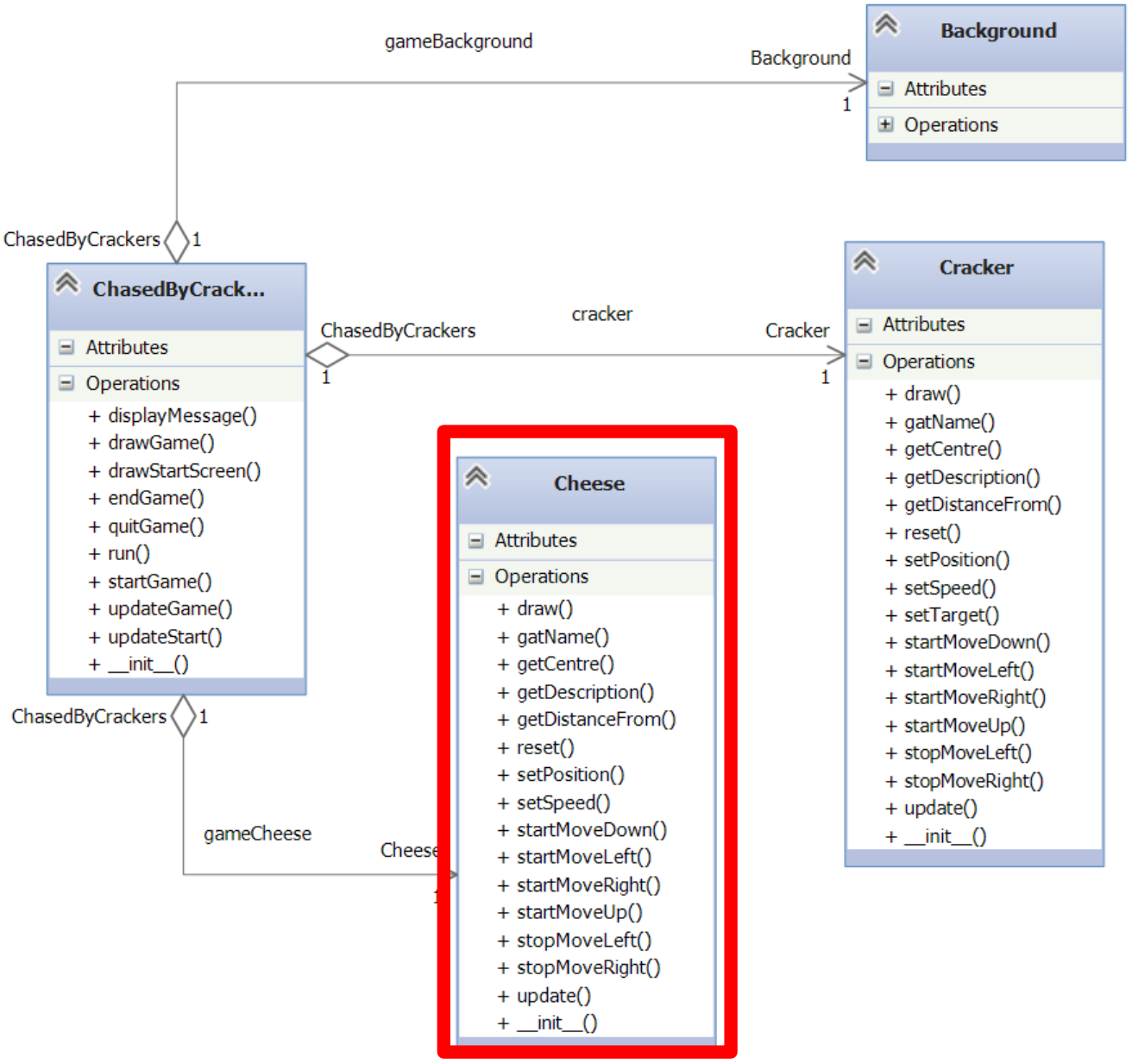
Classes

- This is another way to view the solution
- It shows classes and how they are related



Cheese

- This is the Cheese class
- The diagram shows all the things it can be asked to do



Driving the Cheese

```
if e.key == pygame.K_DOWN:
    self.gameCheese.StartMoveDown()
```

- The CrackerGame class contains a reference to the cheese being used in the game
  - The reference is called `gameCheese`
- When the game wants the cheese to do something it will call a method on the cheese

Driving the Cheese

```
if e.key == pygame.K_DOWN:
    self.gameCheese.StartMoveDown()
```

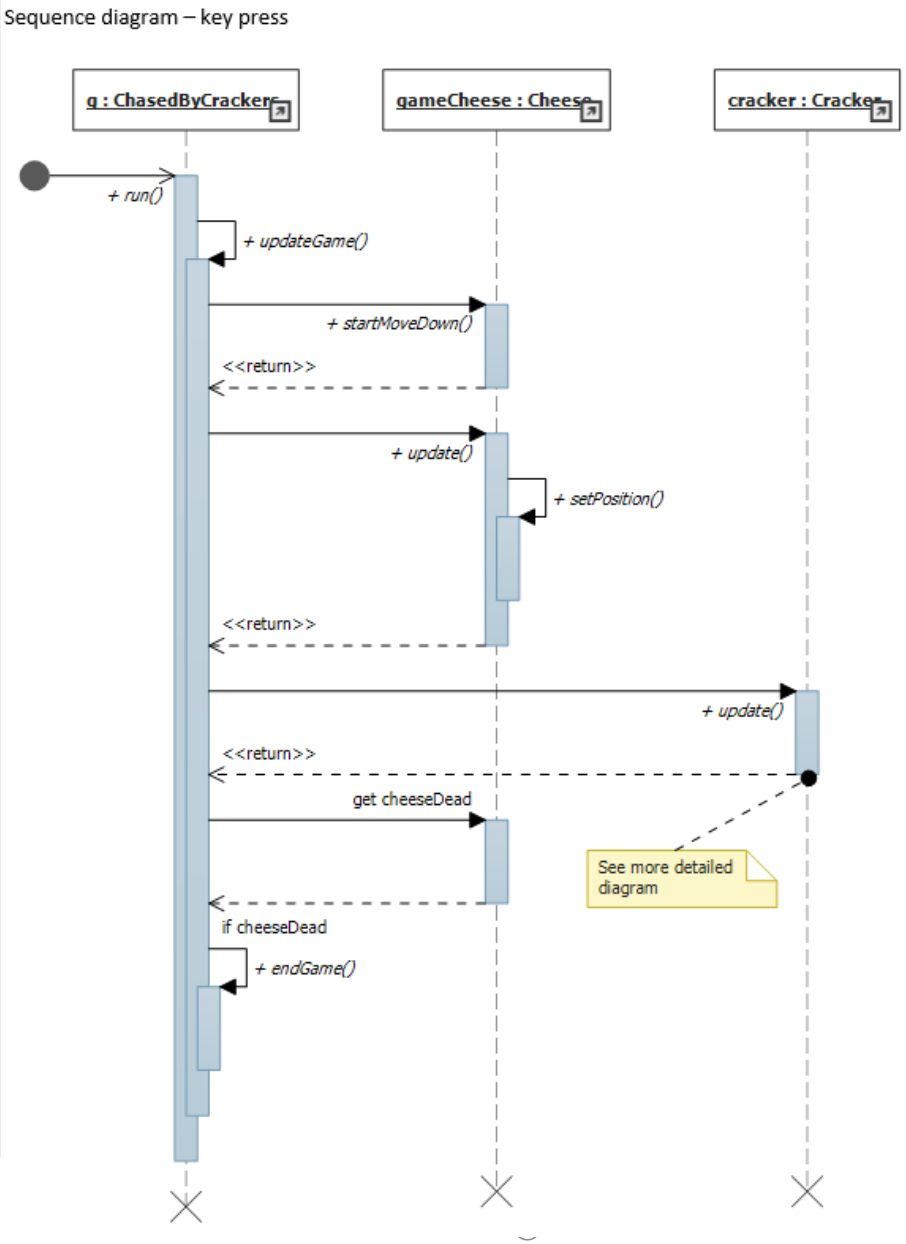
- In the code above the game has detected that the player has pressed the Down key and so the game wants to tell the cheese to start moving down

Methods and Messages

- You can think of calling a method in an object as a way of sending a *message* to that object
- In this respect the message is a bit like a telephone call:
  - I make calls to tell people to do things, or to find things out from them
  - I wait on the line until they come back with their response

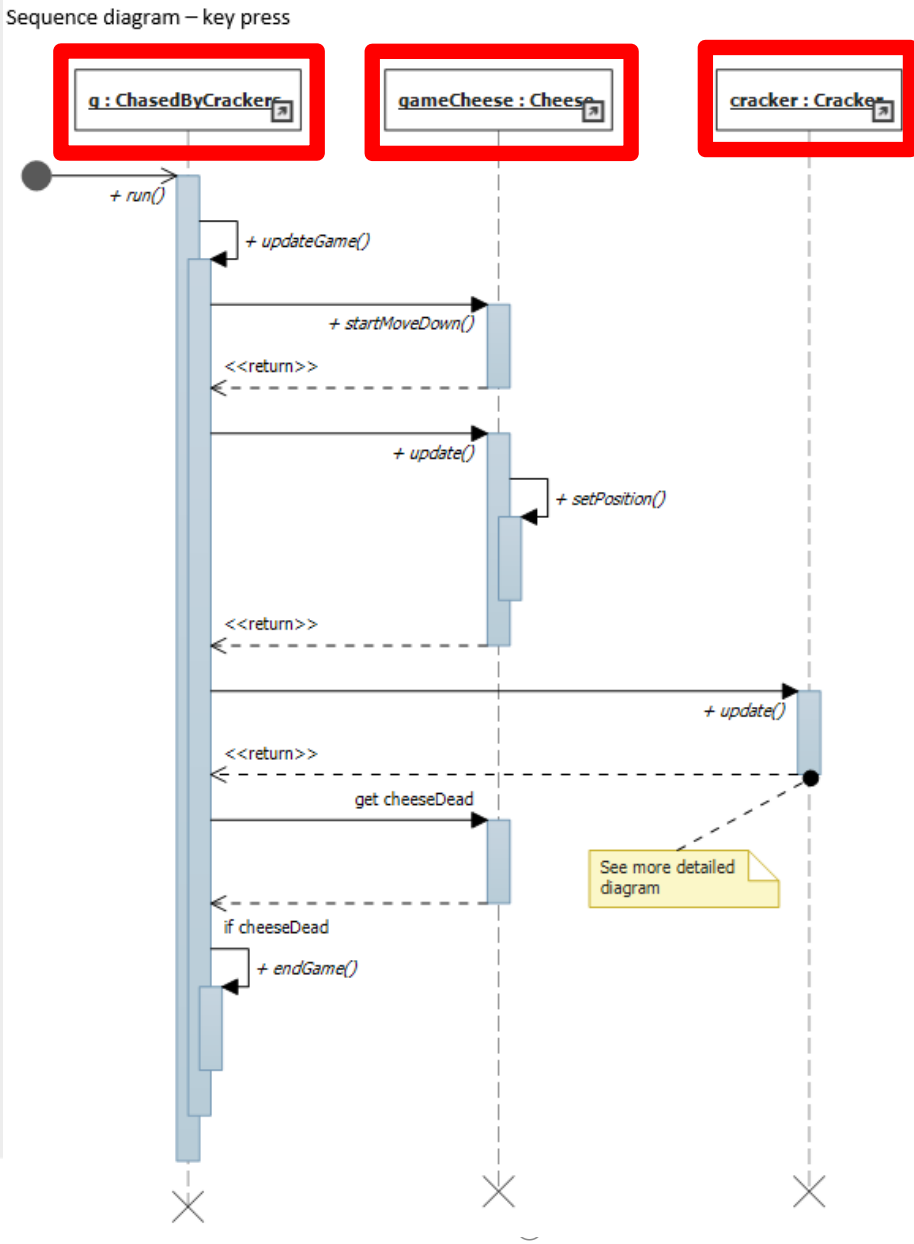
Message Sequencing

- The previous diagram showed the *static* relationship between objects
- This diagram is *dynamic*, showing messages being sent and replies received



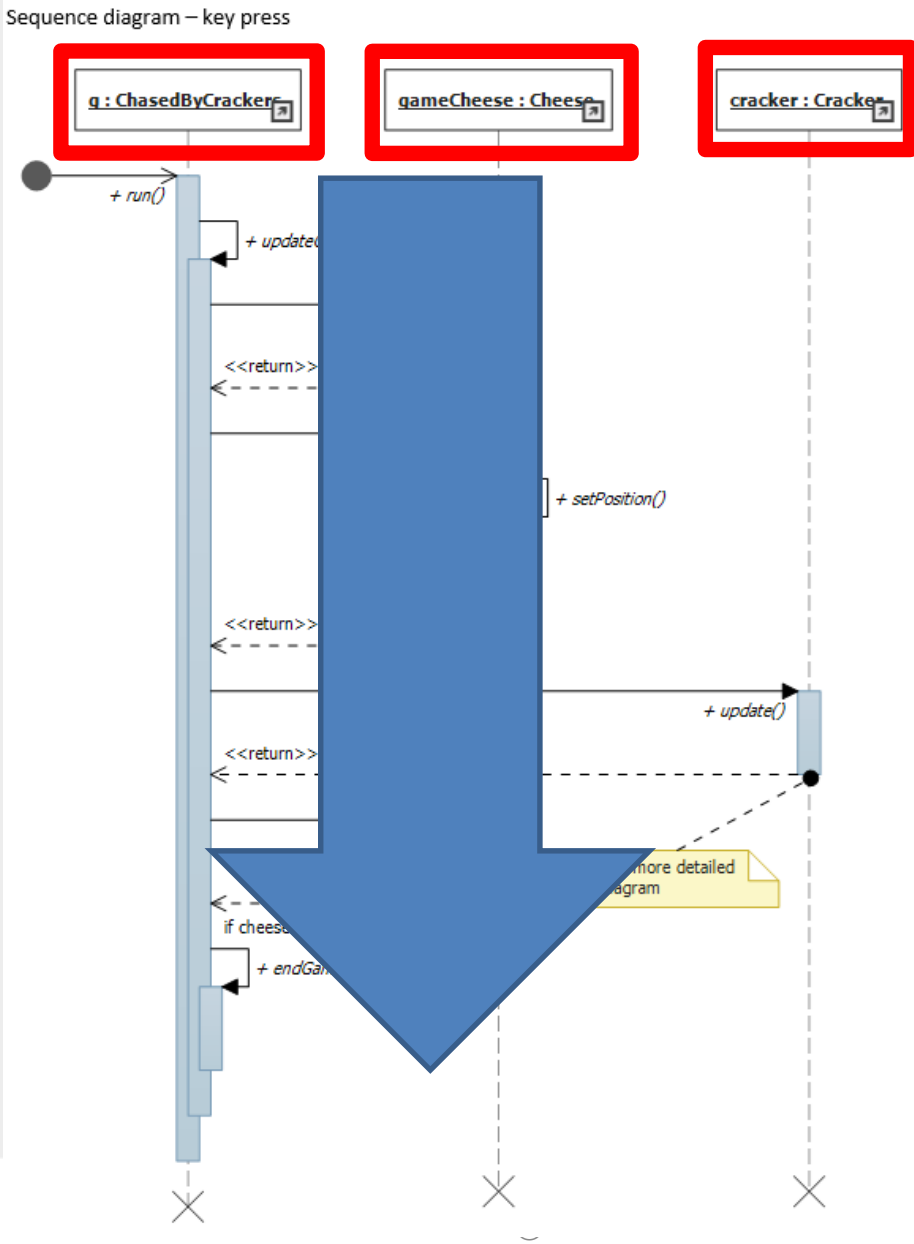
Game Objects

- The diagram shows each of the game objects and how they exchange messages over time



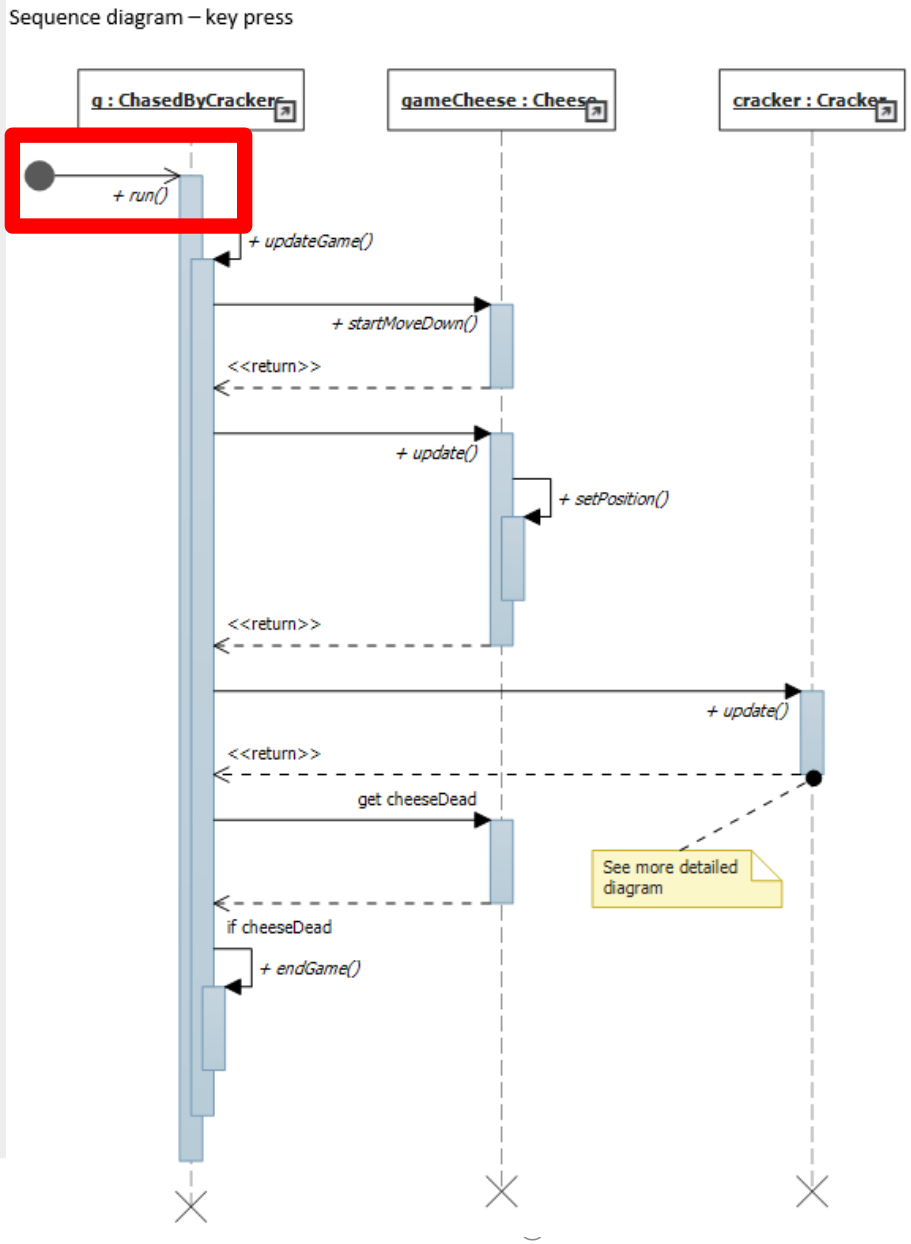
Game Objects

- The diagram shows each of the game objects and how they exchange messages over time
- Time runs down the screen



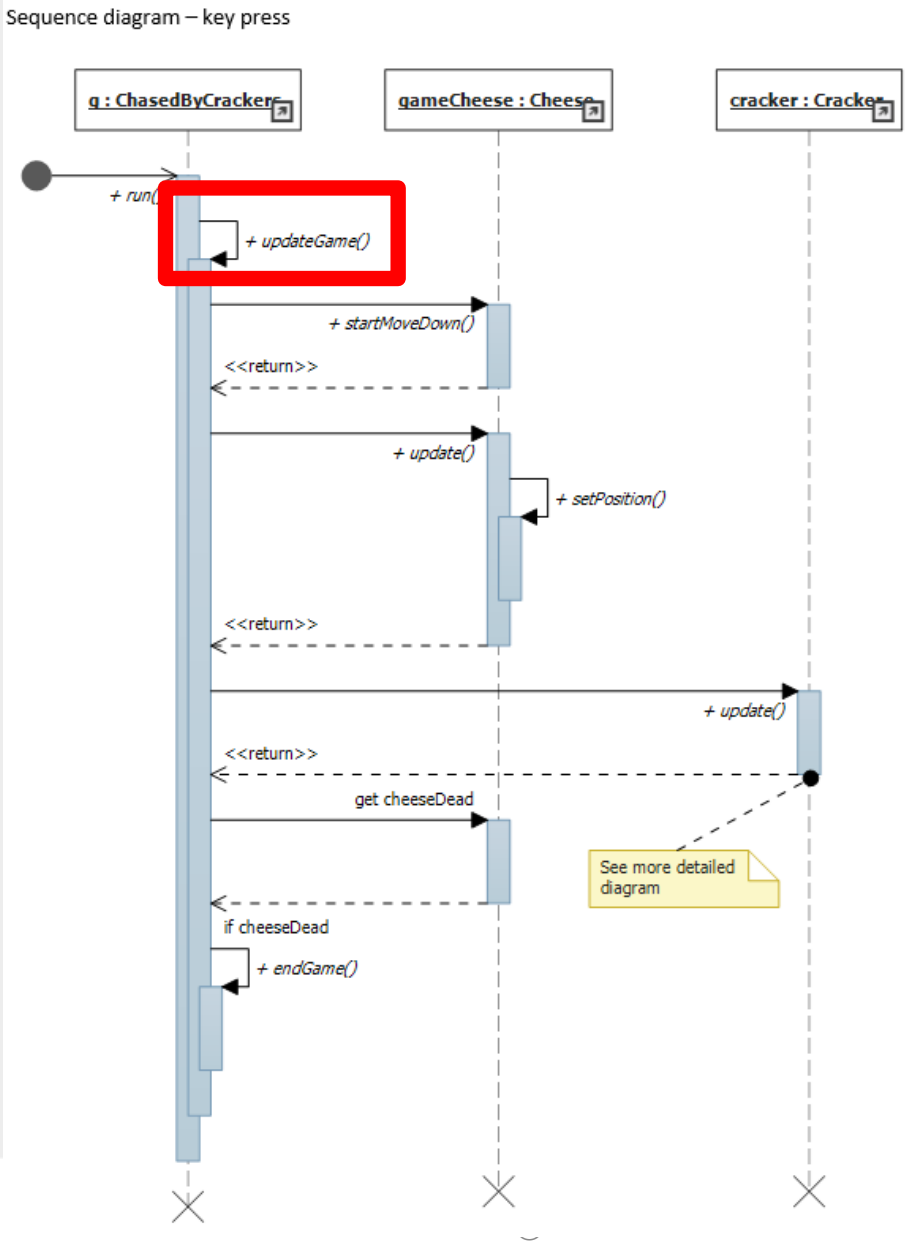
Calling run

- This is where we start
- The run method is called from outside our classes
- It is the message that starts the game



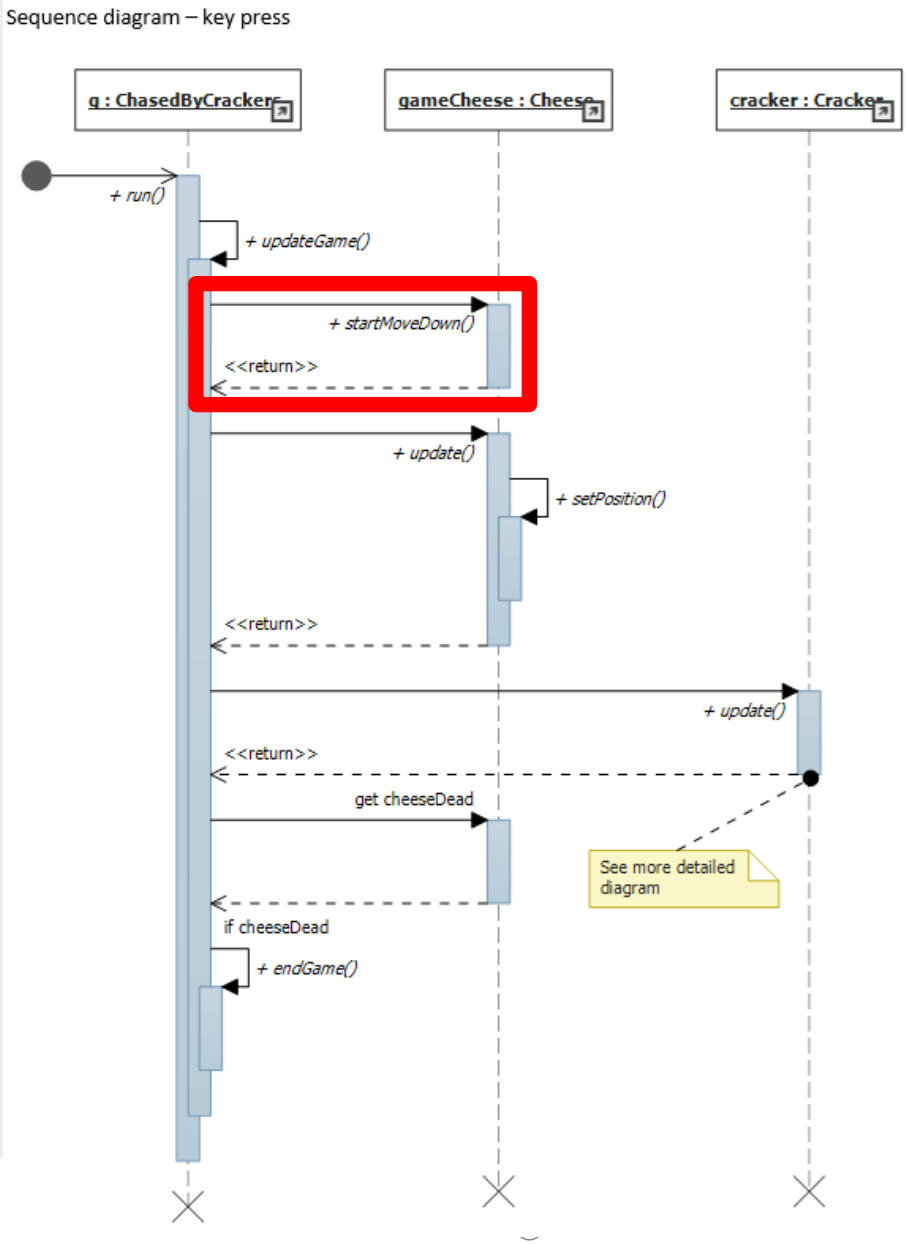
run and updateGame

- The run method calls updateGame
- The updateGame method is actually called repeatedly, we are just showing a single call here



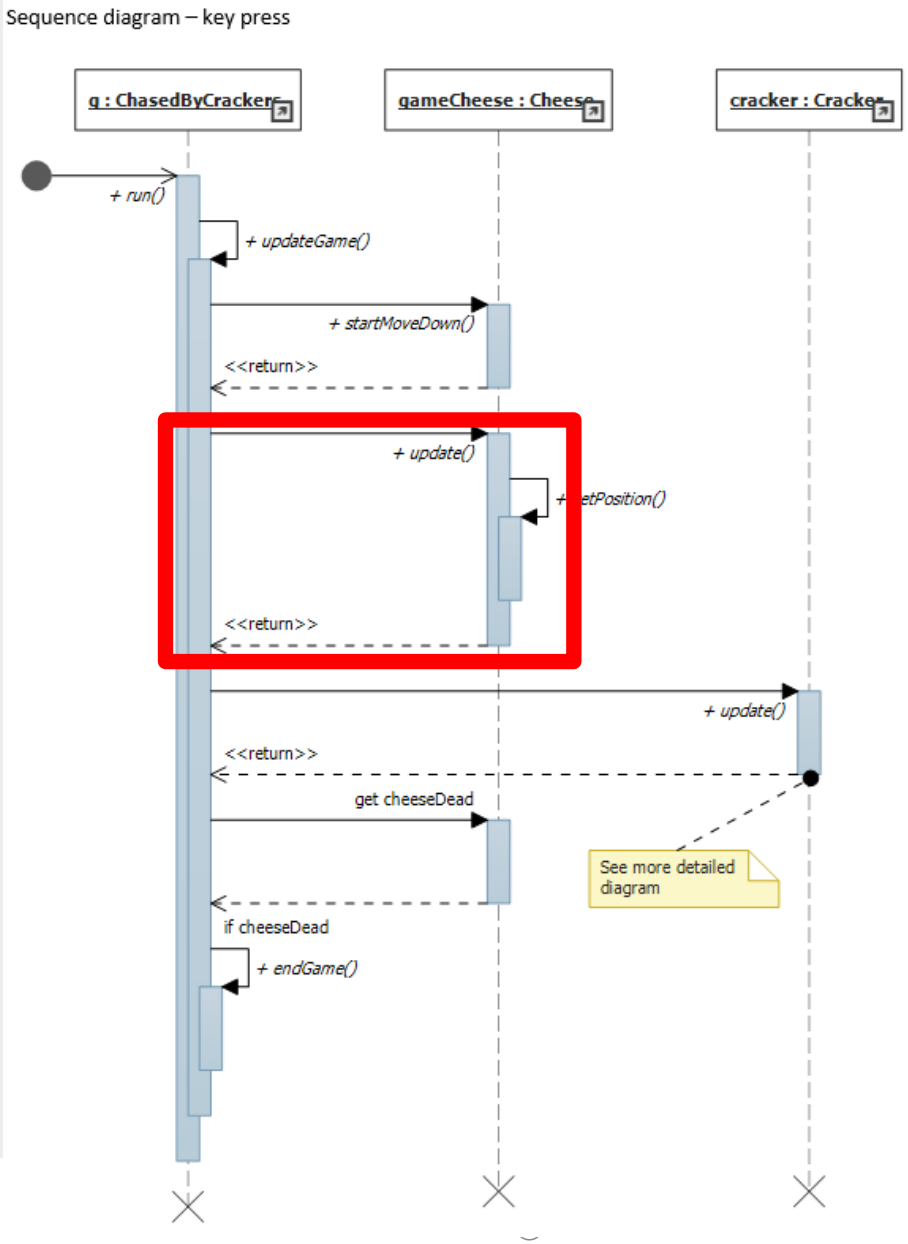
Moving Down

- If the `updateGame` method wants the cheese to start moving down it will send a message to the Cheese to indicate this
- The message is sent if the down key is pressed



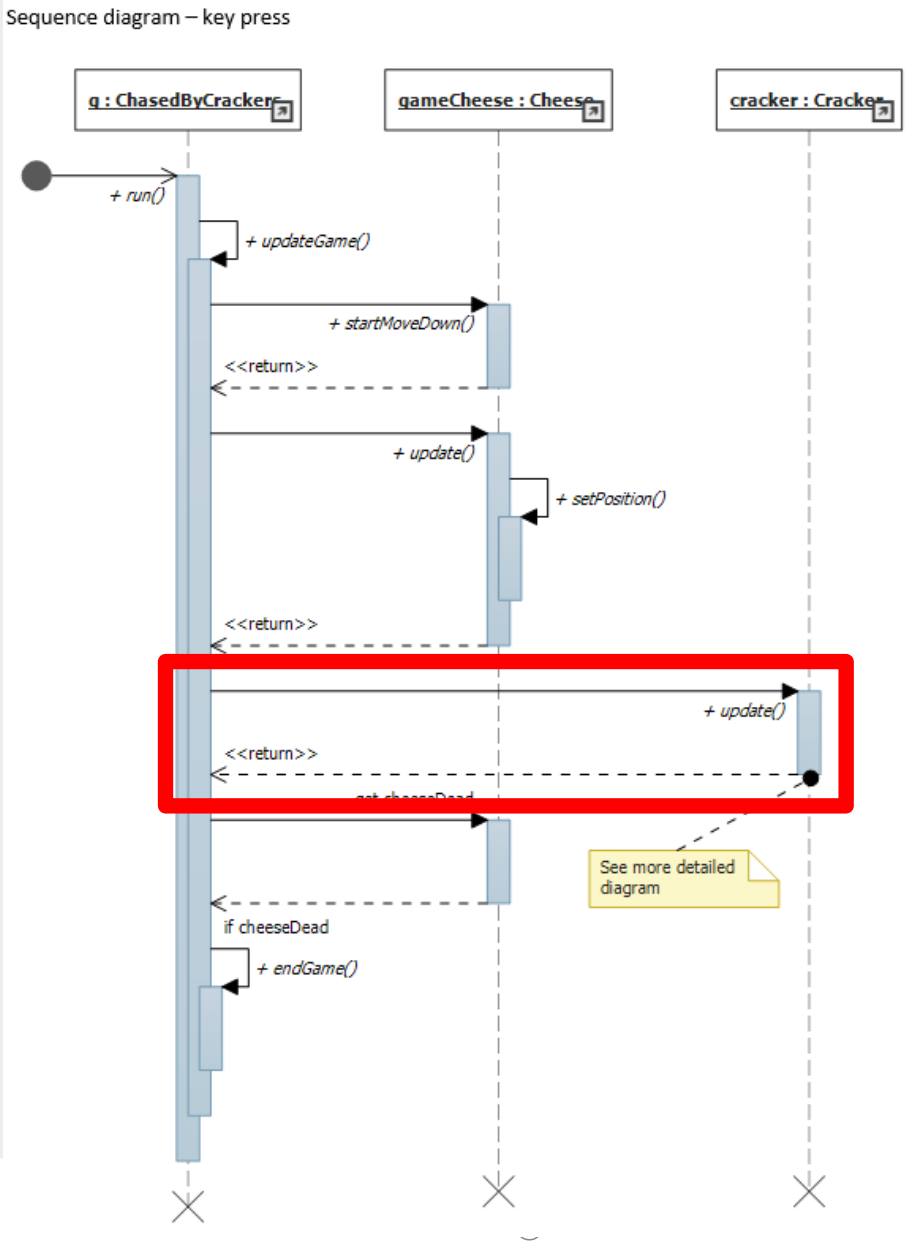
Updating Cheese

- The game always tells the cheese when to Update
- The cheese is then in charge of just what happens during the Update



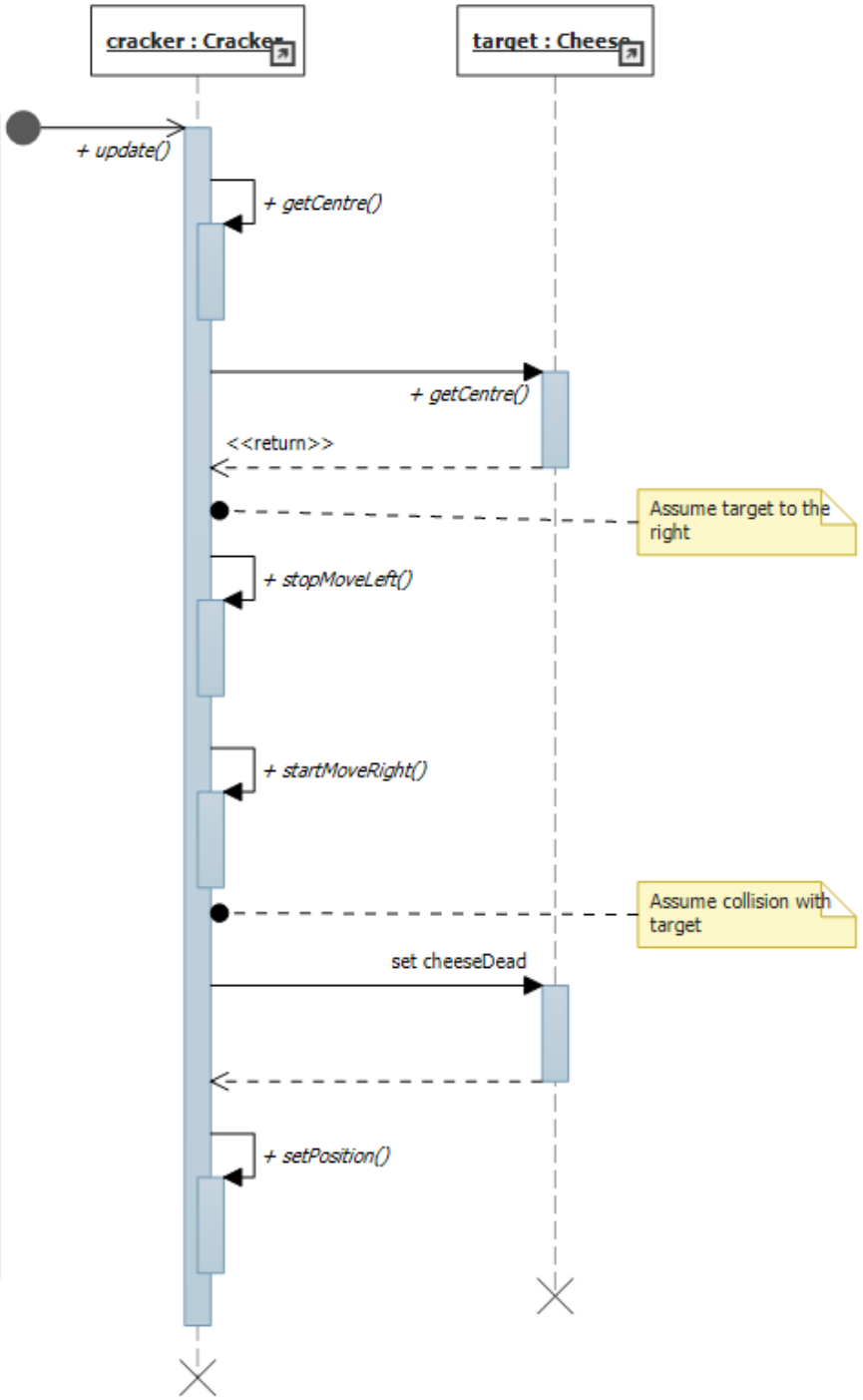
Updating Cracker

- The game also tells the Cracker when it needs to Update
- Again, the Cracker will perform the appropriate response for the request



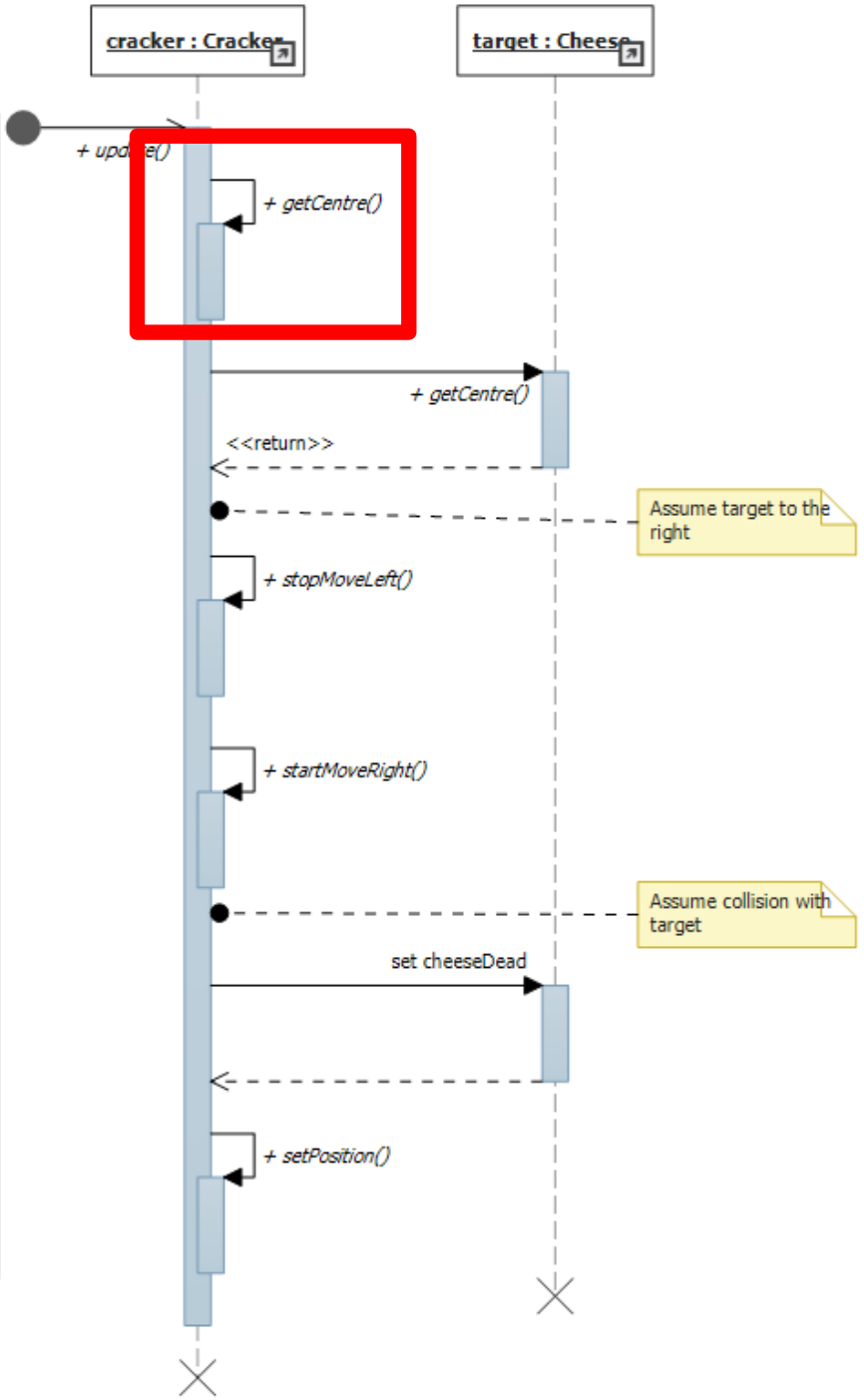
Updating Cracker

- This is the behaviour of the Cracker when it updates
- The Cracker finds out where the cheese is and then moves towards it



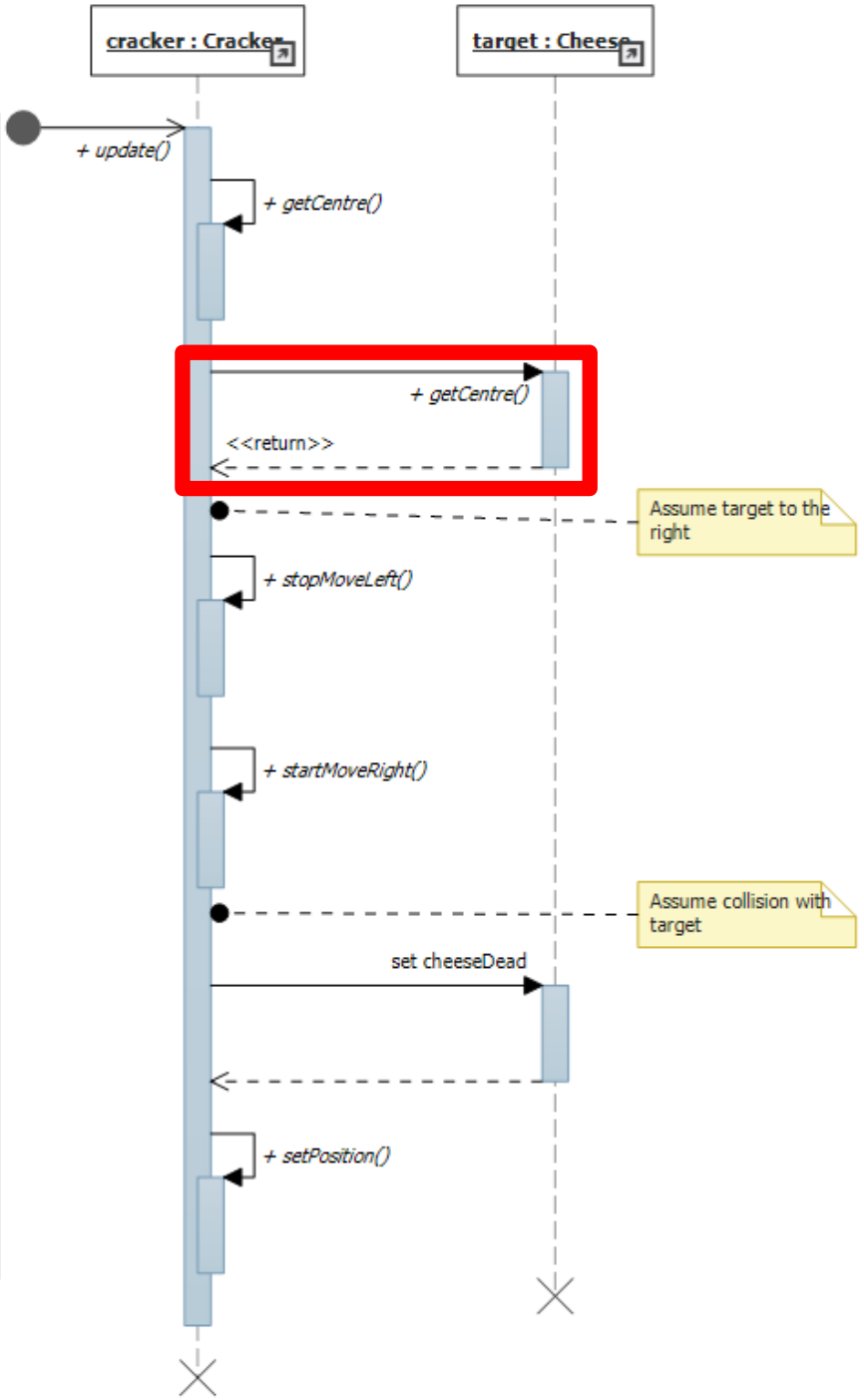
Updating Cracker

- The cracker asks itself where it is



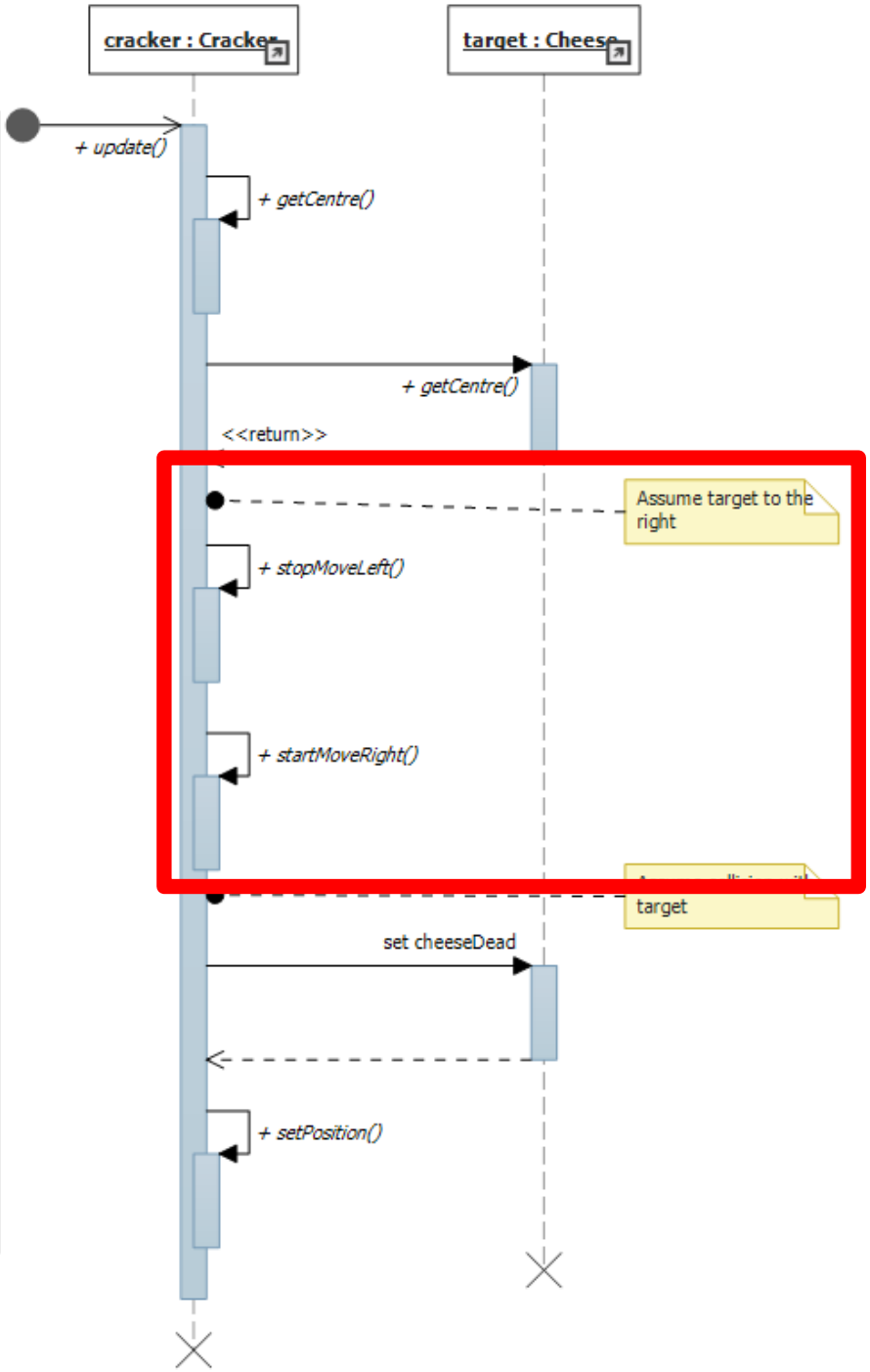
Updating Cracker

- Next the cracker asks the Target (in this case the cheese) where it is



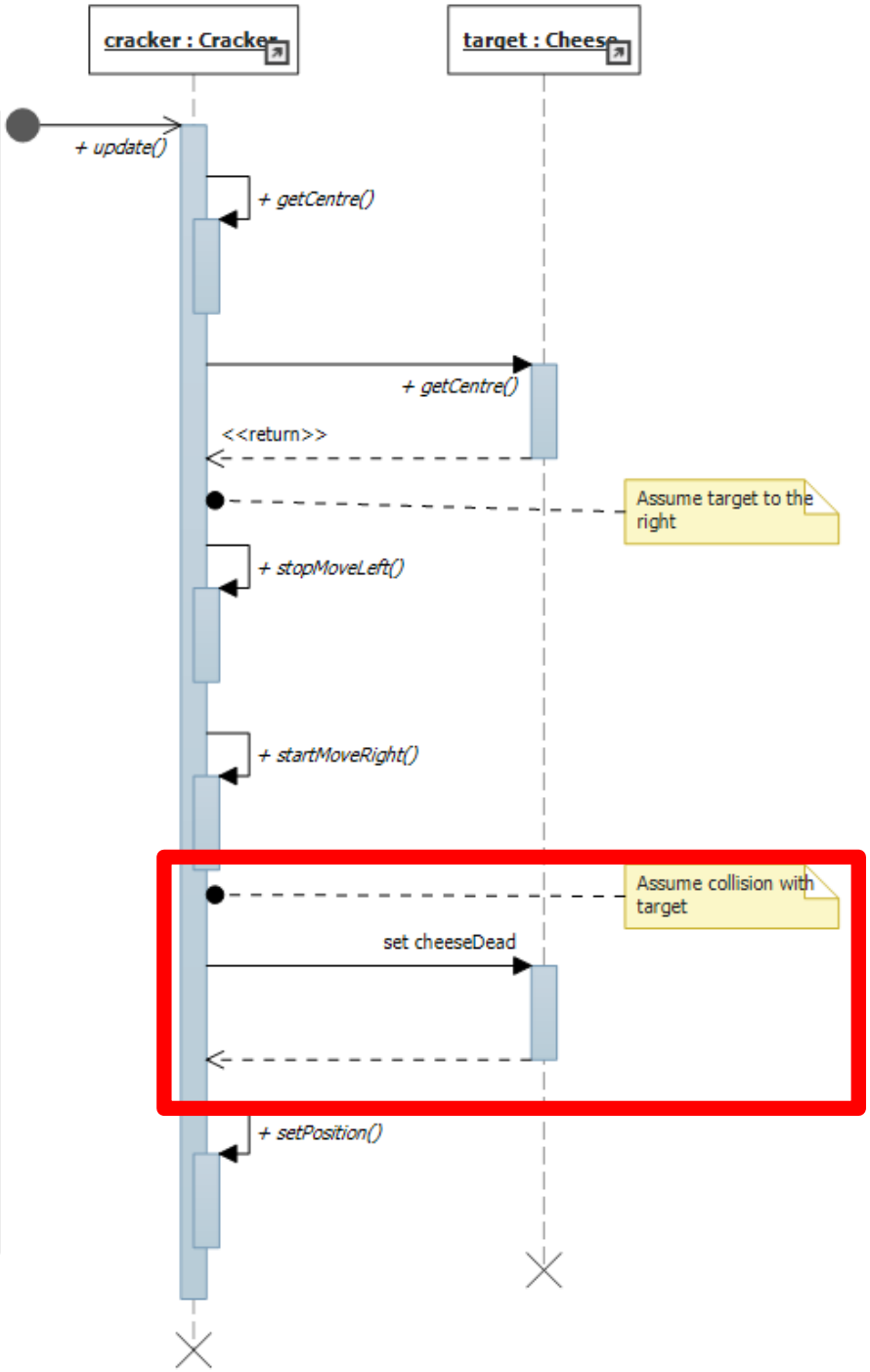
Updating Cracker

- Then the cracker starts moving in the appropriate direction
  - The sequence shown would run if the target was on the right



Killing Cheese

- If the cracker detects a collision it will tell the cheese it is dead
- This is delivered as another method call into the cheese



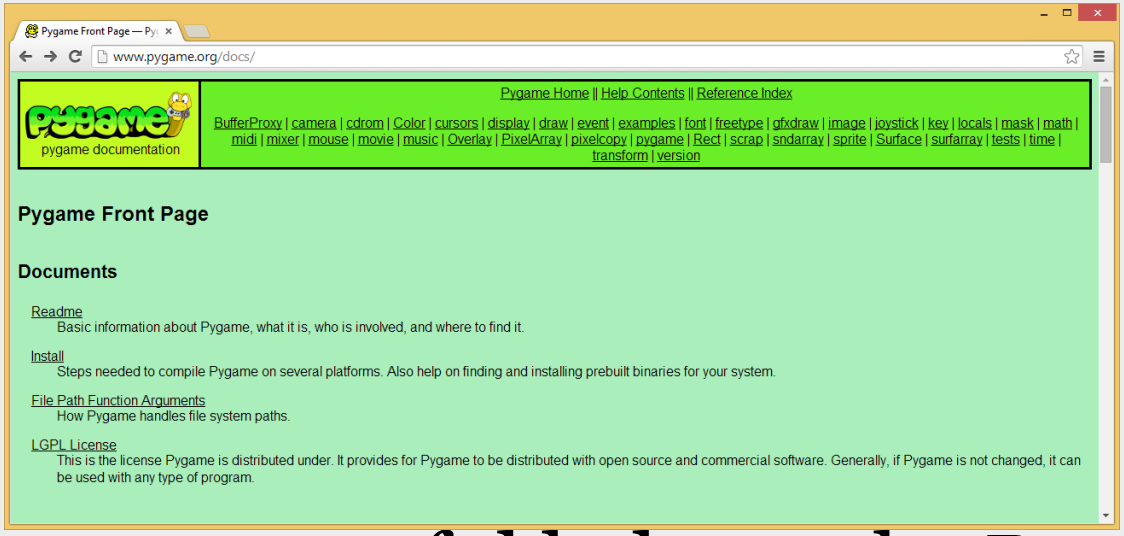
Diagrams and Tools

- The diagrams were produced by a special program that is used during software design
- But you can draw them on pieces of paper and get equally useful results

Programs and Messages

- When we design a program out of components we need to decide
  - The role of each component
  - The data that it manages
  - The messages that it exchanges with other components

Pygame help



- You can get useful help on the Pygame framework from their help pages:

`http://www.pygame.org/docs/`

Working with Components

PRACTICAL BREAK 1