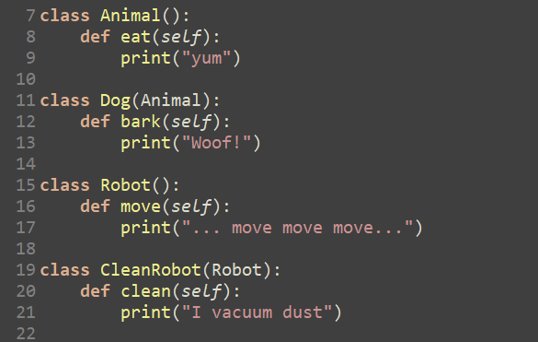
Explaining Association (composition)



**Class (Animal)**

1. We create a class called “**Animal”**.
2. Within that class we define a function called “**eat”**.
3. The function “eat” has a temporary place holder called “**self”.**
4. The functions purpose is to **print “yum”** when it is called.

**Class (Dog)**

1. We create a class called **“dog”**.
2. This class is **inherited** from the **Animal** class. *They are linked*.
3. Within that class we define a function called **“bark”.**
4. The function **“bark”** has a temporary place holder called **“self”**.
5. The functions purpose is to **print “woof”** when it is called.

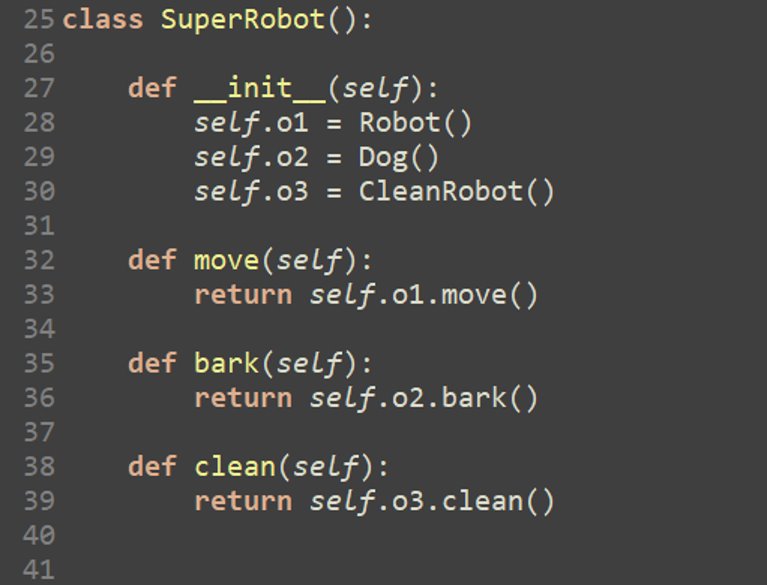
**Class (Robot)**

1. We create a class called **“robot”**.
2. Within that class we define a function called **“move”**.
3. The function **“move”** has a temporary place holder called **“self”**.
4. The functions purpose is to **print “move move move”** when it is called.

**Class (CleanRobot)**

1. We create a class called **“CleanRobot”**.
2. This class is **inherited** from the **“Robot”** class. *They are linked.*
3. Within that class we define a function called **“clean”.**
4. The function **“clean”** has a temporary place holder called **“self”**
5. The functions purpose is to **print “I vacuum dust”** when it is called.

**Class (SuperRobot) - The main Class**



1. We create a class called **“SuperRobot”**.
2. Within the class we create a function called **“ \_\_init\_\_”** which *initiates* the object called **“self”**
3. In that function we can create variables.
4. **“self.o1”** has the value of **“Robot”.** Therefore “**self.o1”** is equal to the class **“Robot”**
5. The same thing applies to the rest of the **“self”** variables.

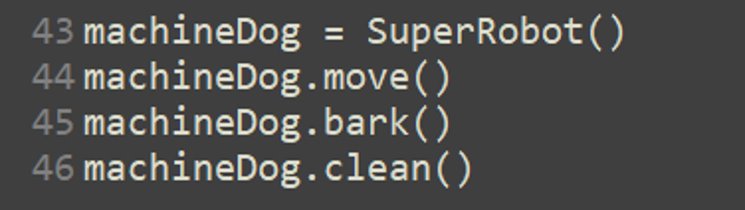
--------------------------------------------------------------------------------------------------------

1. We define a function called **“move”,** that has the **object(self)** as the *argument*.
2. When the function is called, it **returns** 
   * 1. “...move move move...”
     2. **“self.o1**” links to the function **“Robot”**.
     3. Within **“Robot”** it then links to the method **“move”**
     4. It can only do this because we’ve linked it whilst initiating the object.

**Association (composition) is simply creating a link between the classes.**

**We don’t have to create a class within a class.**

**The whole purpose of a class is to logically separate code.**



On the first line of code, we are linking the **object** to the **class.**

All we are doing here is printing the function to the console.

1. We name our object **“machineDog”** and tell it to do something **"move"**.

On the line **“machineDog.move()”**

It will get information from the move function... 

This function will get information from robot class:

