Software Engineering Bootcamp

Hyperiondev

Object-Oriented Design with Class Diagrams

Lecture – Housekeeping

- ❑ The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
- ❑ No question is daft or silly - **ask them!**
- ❑ There are Q/A sessions midway and at the end of the session, should you wish to ask any follow-up questions.
- ❑ You can also submit questions here: http://hyperiondev.com/sbc4-se-questions
- ❑ For all non-academic questions, please submit a query: www.hyperiondev.com/support
- ❑ Report a safeguarding incident: http://hyperiondev.com/safeguardreporting
- ❑ We would love your feedback on lectures: https://hyperionde.wufoo.com/forms/zsgv4m40ui4i0g/

Objectives

1. The basics of class diagrams
    a. Layout
    b. Components
2. Relationships
    a. Identifying the relationship between classes
    b. Indicating the relationship
3. Drawing our own class diagrams

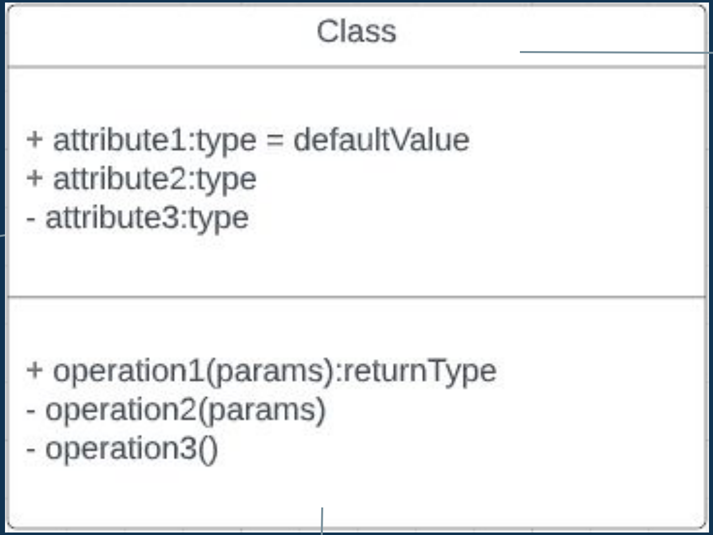
Github Repository - Lecture Examples

https://github.com/HyperionDevBootcamps/C4_SE_lecture_examples

Platform to create class diagrams

https://www.lucidchart.com/pages/uml-class-diagram

The Basics of Class Diagrams



Name of the class

Attributes of the class and the type of data stored

Methods of class objects. Including the parameters and the data type of the return object

Hyperiondev

Class diagrams

A class notation consists of three parts:

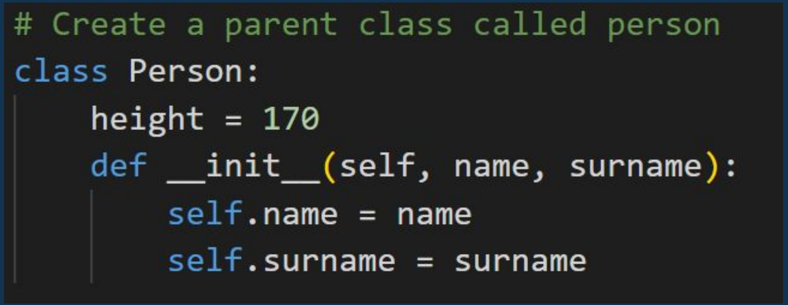
1. **Class Name**
   - The name of the class appears in the first partition.
2. **Class Attributes**
   - Attributes are shown in the second partition.
   - The attribute type is shown after the colon.
   - Attributes map onto member variables (data members) in code.
3. **Class Operations** (Methods)
   - Operations are shown in the third partition. They are services the class provides.
   - The return type of a method is shown after the colon at the end of the method signature.
   - The return type of method parameters is shown after the colon following the parameter name.
   - Operations map onto class methods in code

Relationships Between Classes

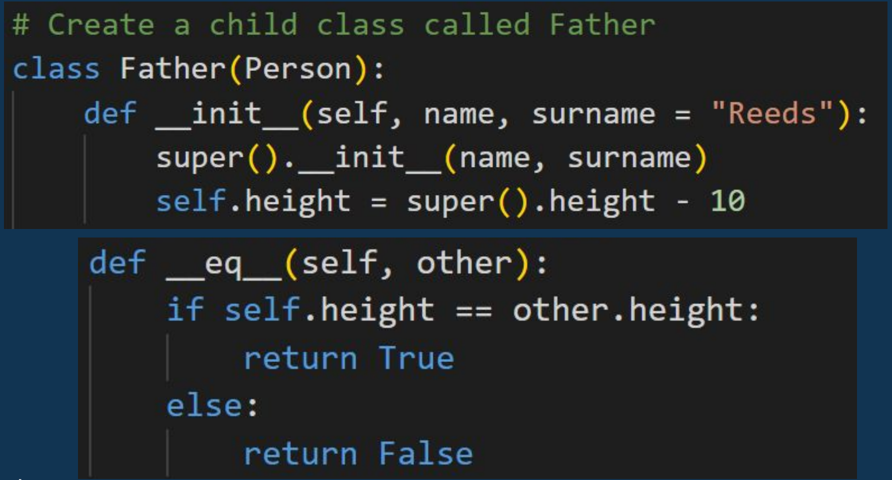
Inheritance	A child class inherits attributes and methods from a parent class.	——▶
Association	A non-dependent relationship just a basic association relationship eg. siblings	——
Aggregation	A specific type of association where the one class can exist without the other.	——◇
Composition	A specific type of association where the one class cannot exist without the other.	——◆

Drawing you own class diagram

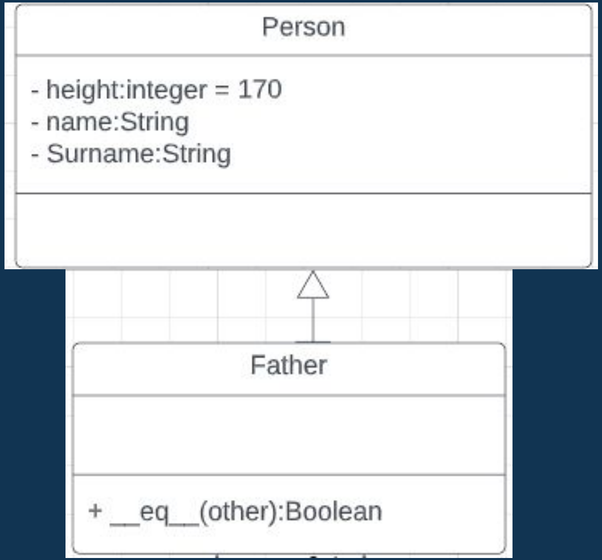
- Let’s have a look at one of our previous inheritance examples.



```python
# Create a parent class called person
class Person:
    height = 170
    def __init__(self, name, surname):
        self.name = name
        self.surname = surname
```

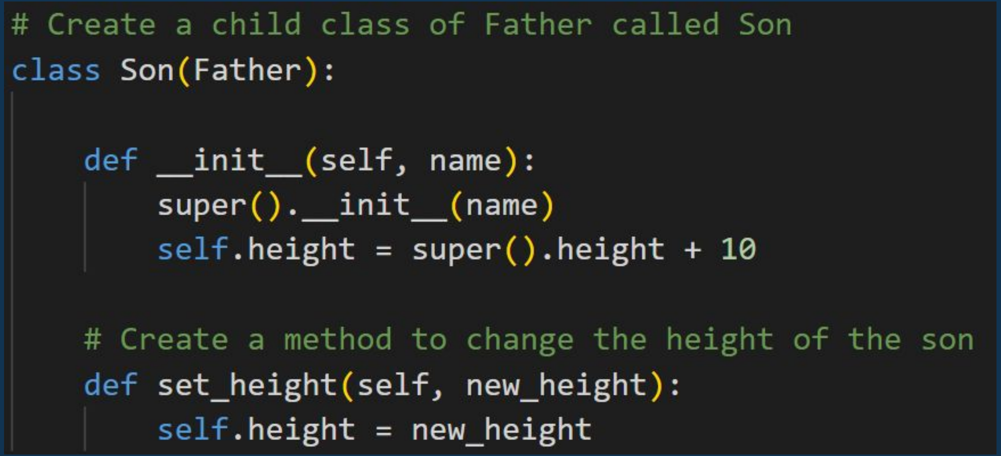


```python
# Create a child class called Father
class Father(Person):
    def __init__(self, name, surname = "Reeds"):
        super().__init__(name, surname)
        self.height = super().height - 10
    def __eq__(self, other):
        if self.height == other.height:
            return True
        else:
            return False
```

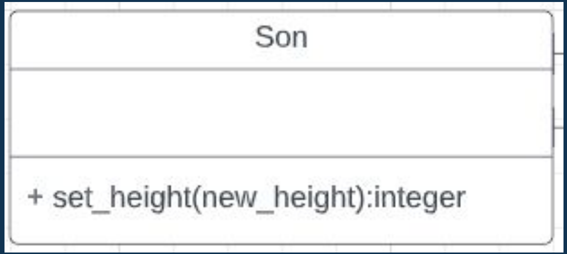


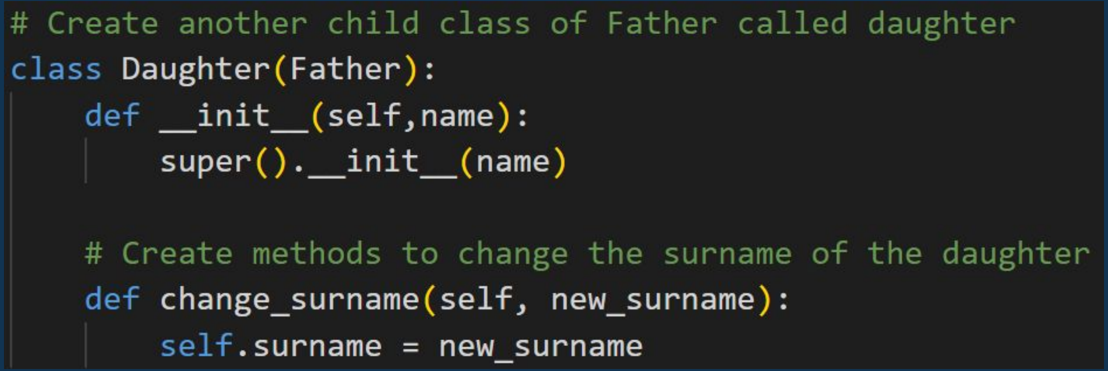
Hyperiondev

- We also had two child classes of Father class

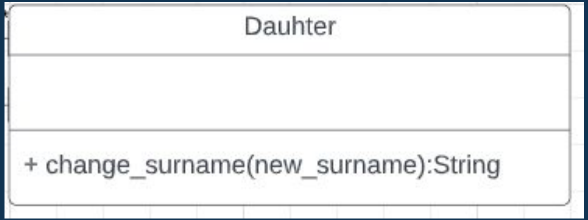


```python
# Create a child class of Father called Son
class Son(Father):

    def __init__(self, name):
        super().__init__(name)
        self.height = super().height + 10

    # Create a method to change the height of the son
    def set_height(self, new_height):
        self.height = new_height
```



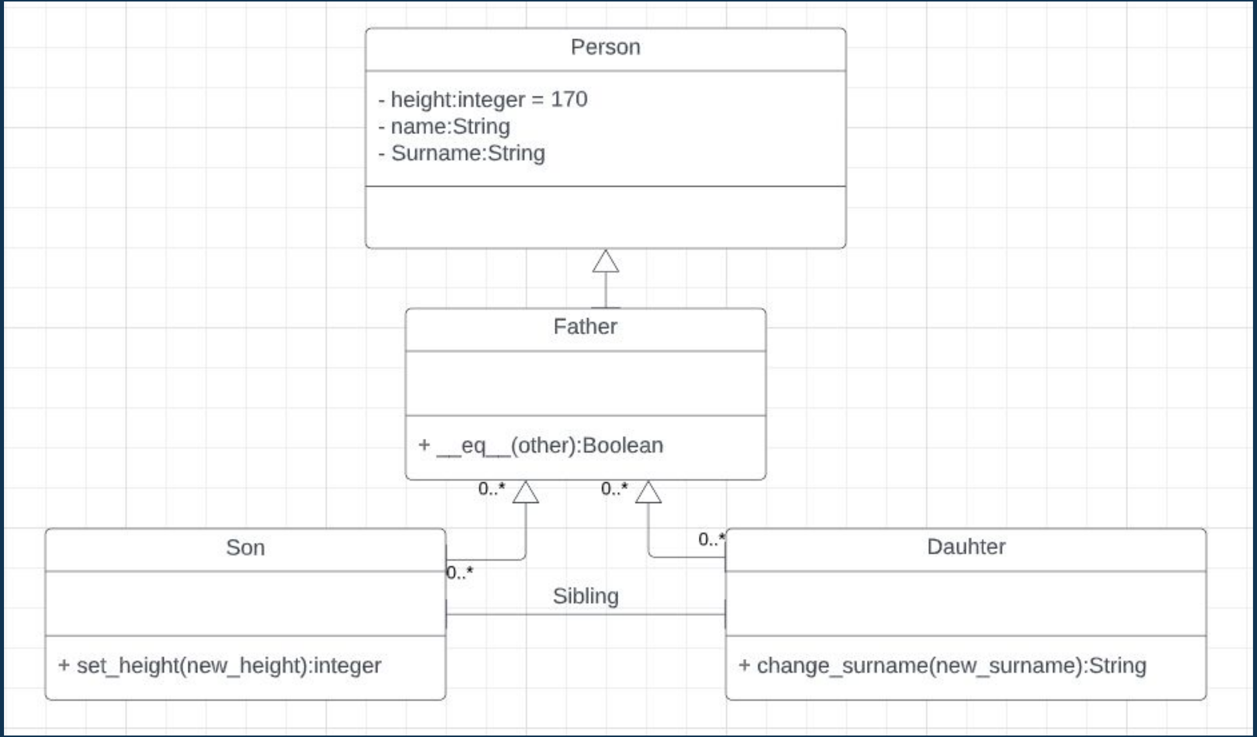


```python
# Create another child class of Father called daughter
class Daughter(Father):
    def __init__(self,name):
        super().__init__(name)

    # Create methods to change the surname of the daughter
    def change_surname(self, new_surname):
        self.surname = new_surname
```

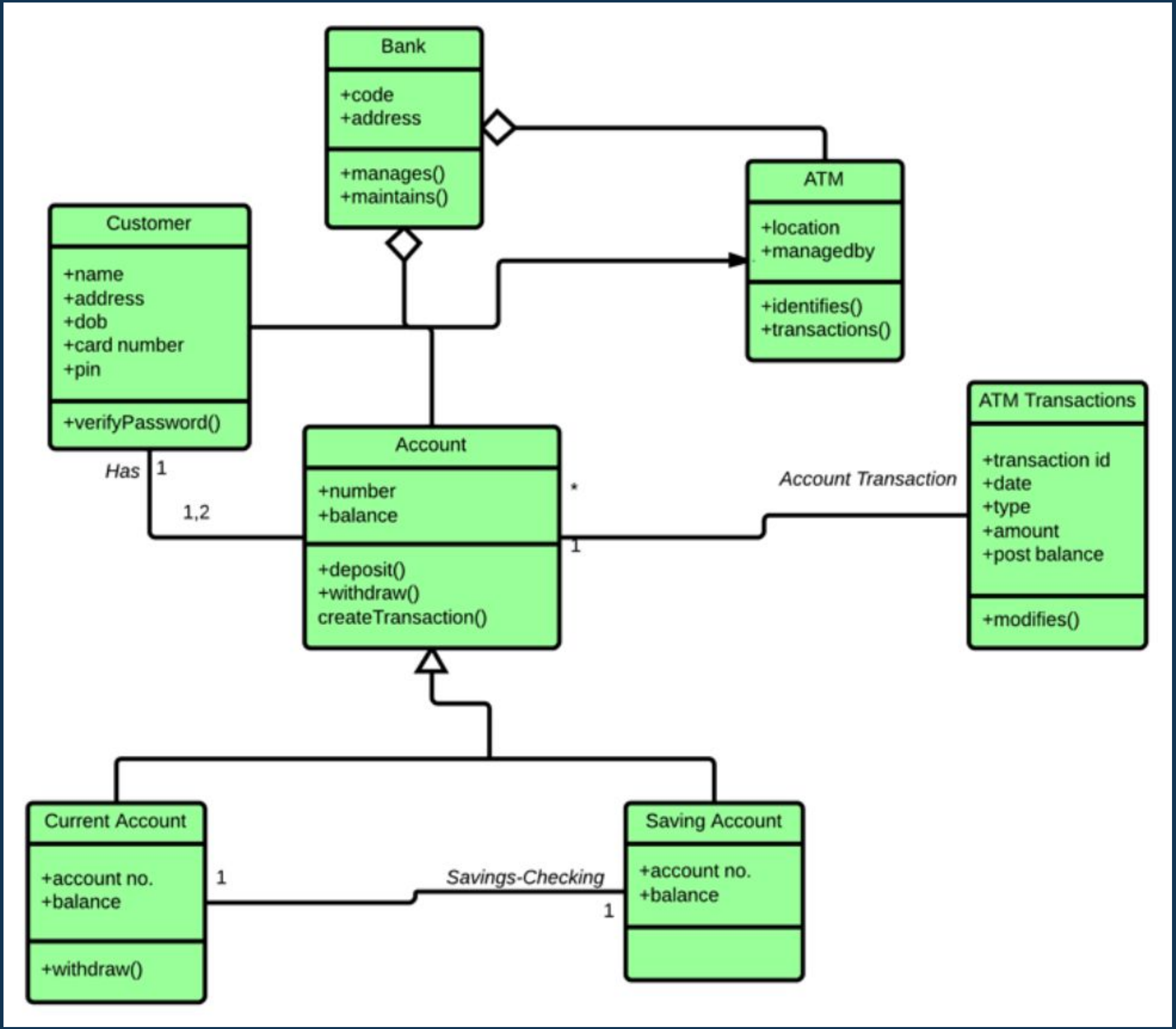


Hyperiondev

The relationship between classes



Hyperiondev


Bank
+code
+address
+manages()
+maintains()
ATM
+location
+managedby
+identifies()
+transactions()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
Has
1
1,2
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
*
1
Account Transaction
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Current Account
+account no.
+balance
+withdraw()
1
Savings-Checking
1
Saving Account
+account no.
+balance

Hyperiondev

Q & A Section

Please use this time to ask any questions relating to the topic explained, should you have any

Hyperiondev

Thank you for joining us