Class Inheritance

> Prerequisite: Classes

Reference: https://docs.python.org/3.1/tutorial/classes.html#inheritance

It is possible for a class to "inherit" its properties and functionality from its "parent" class while at the same time retaining its own specific characteristics.

To setup this example, create a new file in the `class-time` directory called something like `inheritance_example.py` or `team.py`. Inside it, place the following contents:

Definition

```
#
# PARENT CLASS DEFINITION
#

class Team():

  def __init__(self, params):
    self.city = params["city"]
    self.name = params["name"]
    self.sport = params["sport"]
    self.league = params["league"]
    self.players = params["players"]

  def full_name(self):
    return self.city + " " + self.name

#
# CHILD CLASS DEFINITION
#

class BaseballTeam(Team):

    def __init__(self, params):
        params["sport"] = "baseball"
        super().__init__(params) # equivalent to: `Team.__init__(self,
params)`
```

Normally, each class definition would exist inside its own file, but we are defining both in the same file for example purposes.

Initialization

Normally we would reference the class from another file by importing it, but for example purposes, place the following contents at the bottom of the script you used to set up your inheritance example:

```
attributes = {
    "city": "New York",
    "name": "Yankees",
    "league": "major",
    "players": ["Jeter", "Mariano", "Mantle", "Babe"]
}

bt = BaseballTeam(attributes)

type(bt) #> __main__.BaseballTeam

bt.name #> 'Yankees'

bt.city #> 'New York'

bt.players #> ['Jeter', 'Mariano', 'Mantle', 'Babe']

bt.full_name() #> 'New York Yankees'
```

In this inheritance example, the reason why an instance of the `BaseballTeam` class has the `full_name()` property is because that property was defined in the parent class (`Team`).