Python3 class methods 20200511_083708

cSubnetType___CodeChallenge__Nr01 20200511_091829

SDNScottie

Here we want to learn the difference between the python class methods.

We have the easier (**static methods**), then the more complex (**class methods**)

First we'll concentrate on the (static methods). For our example, we'll use
for our implementation the IP Addresses (Class A, Class B, Class C).

We wanto to know the differences and then use Python3 to automate the decision making.

So first read :

#================
class method vs static method in Python
#================

https://www.geeksforgeeks.org/class-method-vs-static-method-python/

Their Example using a **Person** :

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    # a class method to create a Person object by birth year.
    @classmethod
    def fromBirthYear(cls, name, year):
        return cls(name, date.today().year - year)

    # a static method to check if a Person is adult or not.
    @staticmethod
    def isAdult(age):
        return age > 18

person1 = Person('mayank', 21)
person2 = Person.fromBirthYear('mayank', 1996)

print person1.age
print person2.age

# print the result
print Person.isAdult(22)
```

Now, we want a Python3 CodeChallenge !

Our Example : We want to make an Example checking if a subnet is class A, class B or class C

as a reference use : https://de.wikipedia.org/wiki/Netzklasse

https://github.com/SDNScottie/pyprofi/blob/master/subnetting/subnetting__m__CodeChallenge.py

---

```
class cSubnetType___CodeChallenge__Nr01:
    def __init__(self, ip_address, slash_mask):
        self.name = ip_address
        self.age = slash_mask      # slash_mask can be ( 8 thru 32 )
        #== perform test :
        #=1) take the ip_address and break it up into a ( List )
        List_ip_address = ip_address.split(".")
        #== Now, send the ( List ) into the individuals ( methods )
        #== and determine if class A, class B or class C IP address :-D
        if self.check_if_class_A(List_ip_address, slash_mask) == True:
            print("IP is a class A")
        elif self.check_if_class_B(List_ip_address, slash_mask) == True:
            print("IP is a class B")
        elif self.checkt_if_class_C(List_ip_address, slash_mask) == True:
            print("IP is a class C")
    def check_if_class_A(self, List_ip_address, slash_mask):
        print("check if class A : " + str(List_ip_address))
        result = True
        #== Tip: Here the goal is break compare the List members and
        # determine if IP is class A
        return result
    def check_if_class_B(self, List_ip_address, slash_mask):
        print("check if class A : " + str(List_ip_address))
        result = True
        #== Tip: Here the goal is break compare the List members and
        # determine if IP is class A
        return result
    def checkt_if_class_C(self, List_ip_address, slash_mask):
        print("check if class A : " + str(List_ip_address))
        result = True
        #== Tip: Here the goal is break compare the List members and
        # determine if IP is class A
        return result
# ============================================================
# == MAIN
# ============================================================
def main_cSubnetType():
    cSubnetType___CodeChallenge__Nr01("192.168.0.0","16")
if __name__ == '__main__':
    main_cSubnetType()
```

So take the code and take the Challenge :-D
Just as me questions if need be while trying to complete the implementation.

Upon completion, send your code to me.

Rename your code extension to **.pypro** before you send it per Email.

Example, if your name is (John), then
subnetting__m__CodeChallenge__John**.pypro**

of if your name is (Sussie), then
subnetting__m__CodeChallenge__Sussie**.pypro**

Best of Luck,
Scottie