Python Dunder Methods

Dunder methods are also called special techniques that are a set of predefined methods in Python that can be used to improve your classes. Because they begin and conclude with double underscores, such as \_init\_ or \_str\_, they are easily identifiable. In Python, these “dunders” or “special methods” are sometimes known as “magic methods.” However, adopting this phrase can make things appear more sophisticated than they are. These techniques should be handled as if they have built-in language aspects.

1-init

This is a method you must have already used if you have worked with classes. The init method is used to create an instance of the class.

def \_init\_(self,names):

if names:

self.names = names.copy()

for name in names:

self.versions[name] = 1

else:

raise Exception("Please Enter the names")

2-str

The str method is useful when we want to use instances of our class in a print statement. As discussed earlier, it usually returns a memory object. But we can override the str method to meet our requirements.

def \_str\_(self):

s ="The current softwares and their versions are listed below: \n"

for key,value in self.versions.items():

s+= f"{key} : v{value} \n"

return s

3-setitem

When assigning values in a dictionary, the setitem method is invoked.

d = {}

d['key'] = value

def \_setitem\_(self,name,version):

if name in self.versions:

self.versions[name] = version

else:

raise Exception("Software Name doesn't exist")

4-getitem

The getitem method is like the setitem method, the major difference being that the getitem method is called when we use the [] operator of a dictionary.

d = {'val':key}

print(d['val'])

def \_getitem\_(self,name):

if name in self.versions:

return self.versions[name]

else:

raise Exception("Software Name doesn't exist")

5-len

In a dictionary, the len method returns the number of elements in a list or the number of key-value pairs in a dictionary.

We can define a len method for our class as well.

def \_len\_(self):

return len(self.names)

6-contains

The contain method is used when the operator is used. Return value must be Boolean

def \_contains\_(self,name):

if name in self.versions:

return True

else:

return False

if 'S2' in p:

print("Software Exists")

else:

print("Software DOESN'T exist")