Mutable and Immutable objects in Python

Python is Object Oriented Programming Language. All data in python is represented by objects. Each object has an identity, a value and a type.

Object's id (identity) can think as memory address. This is unique number which never changes once the object is created. In Python built-in id() function returns the identity of the object.

Mutable Objects

Dictionary meaning of mutable is changeable.

Objects that can change in-place are considered as mutable objects. Mutable objects can change its content.

Example:

```
>>> no1list = [1, 2, 3]
>>> no2list = no1list
>>> id(no1list)
2800427196360
>>> id(no2list)
2800427196360
>>>
```

Notice **id of both no1list and no2list are the same**. It means they are the label of same objects.

If you update the no1list then the changes will be reflected in no2list also. It happens only in mutable objects.

Example:

```
no1list = [7, 8, 9]
no2list = no1list
print(id(no1list))
print(id(no2list))
print(no1list is no2list)
no1list+= [22, 24, 25]
print(no2list)
```

Output

```
2980955419528

2980955419528

True

[7.8, 9, 22, 24, 25]
```

Following are the Mutable objects,

1. list
2. dictionary
3. set
4. byte array
5. user defined classes

Immutable Objects

When we creates objects that cannot be change it is called immutable

Example

```
x = 10

>>> id(x)

140724193846384

>>> x += 2

>>> id(x)

140724193846448

>>> x

12
```

Notice that id of x when value is been assign and after the updating is **different.** It has created new x that is why the id is different.

Following are the immutable objects

1. int
2. float
3. decimal
4. complex
5. bool
6. string
7. tuple

8. range
9. frozenset
10. bytes

In Nutshell:

- Immutable object is an object whose state cannot be modified after it is created.
- Mutable object is an object which can be modified after it is created.