GKTCS INNOVATIONS

Modules and Packages

Index

GKTCS INNOVATIONS

1. Python Modules

GKTCS INNOVATIONS

A python module can be defined as a python program file which contains a python code including python functions, class, or variables. In other words, we can say that our python code file saved with the extension (.py) is treated as the module. We may have a runnable code inside the python module.

Modules in Python provides us the flexibility to organize the code in a logical way.

GKTCS INNOVATIONS

Example:

In this example, we will create a module named as file.py which contains a function func that contains a code to print some message on the console.

Let's create the module named as file.py.

```
#displayMsg prints a message to the name being passed.

def displayMsg(name)
    print("Hi "+name);
```

2. Loading the module in our python code

GKTCS INNOVATIONS

- The import statement
- The from-import statement

The import statement

The import statement is used to import all the functionality of one module into another. Here, we must notice that we can use the functionality of any python source file by importing that file as the module into another python source file.

GKTCS INNOVATIONS

We can import multiple modules with a single import statement, but a module is loaded once regardless of the number of times, it has been imported into our file.

The syntax to use the import statement is given below.

```
import module1,module2,…….. module n
```

GKTCS INNOVATIONS

Example:

```
import file;
name = input("Enter the name?")
file.displayMsg(name)
```

Output:

```
Enter the name?John
Hi John
```

GKTCS INNOVATIONS

The from-import statement

Instead of importing the whole module into the namespace, python provides the flexibility to import only the specific attributes of a module. This can be done by using from? import statement. The syntax to use the from-import statement is given below.

```
from < module-name> import <name 1>, <name 2>..,<name n>
```

GKTCS INNOVATIONS

calculation.py:

```
#place the code in the calculation.py
def summation(a,b):
    return a+b
def multiplication(a,b):
    return a*b;
def divide(a,b):
    return a/b;
```

GKTCS INNOVATIONS

Main.py:

```
from calculation import summation
#it will import only the summation() from calculation.py

a = int(input("Enter the first number"))
b = int(input("Enter the second number"))
print("Sum = ",summation(a,b))

'''we do not need to specify the module name while
accessing summation()'''
```

GKTCS INNOVATIONS

Output:

```
Enter the first number10
Enter the second number20
Sum =  30
```

GKTCS INNOVATIONS

The from...import statement is always better to use if we know the attributes to be imported from the module in advance. It doesn't let our code to be heavier. We can also import all the attributes from a module by using *.

Consider the following syntax.

```
from <module> import *
```

3. Renaming a module

GKTCS INNOVATIONS

Python provides us the flexibility to import some module with a specific name so that we can use this name to use that module in our python source file.

The syntax to rename a module is given below.

```
import <module-name> as <specific-name>
```

GKTCS INNOVATIONS

Example:

```
'''the module calculation of previous example is imported in this example as cal.'''
import calculation as cal;
a = int(input("Enter a?"));
b = int(input("Enter b?"));
print("Sum = ",cal.summation(a,b))
```

Output:

```
Enter a?10
Enter b?20
Sum =  30
```

4. The reload() function

GKTCS INNOVATIONS

As we have already stated that, a module is loaded once regardless of the number of times it is imported into the python source file. However, if you want to reload the already imported module to re-execute the top-level code, python provides us the **reload()** function. The syntax to use the **reload()** function is given below.

```
reload(<module-name>)
```

5. Scope of variables

GKTCS INNOVATIONS

In Python, variables are associated with two types of scopes. All the variables defined in a module contain the global scope unless or until it is defined within a function.

All the variables defined inside a function contain a local scope that is limited to this function itself. We can not access a local variable globally.

GKTCS INNOVATIONS

Example:

```
name = "john"
def print_name(name):
    print("Hi",name) #prints the name that is local to this
function only.
name = input("Enter the name?")
print_name(name)
```

Output:

Hi David

6. Python packages

GKTCS INNOVATIONS

The packages in python facilitate the developer with the application development environment by providing a hierarchical directory structure where a package contains sub-packages, modules, and sub-modules. The packages are used to categorize the application level code efficiently.

Let's create a package named Employees in your home directory. Consider the following steps.

1. Create a directory with name Employees on path /home.

GKTCS INNOVATIONS

2. Create a python source file with name ITEmployees.py on the path /home/Employees.

ITEmployees.py

```
def getITNames():
    List = ["John", "David", "Nick", "Martin"]
    return List;
```

GKTCS INNOVATIONS

3. Similarly, create one more python file with name BPOEmployees.py and create a function getBPONames().

GKTCS INNOVATIONS

4. Now, the directory Employees which we have created in the first step contains two python modules. To make this directory a package, we need to include one more file here, that is __init__.py which contains the import statements of the modules defined in this directory.

__init__.py

```
from ITEmployees import getITNames
from BPOEmployees import getBPONames
```

GKTCS INNOVATIONS

5. Now, the directory Employees has become the package containing two python modules. Here we must notice that we must have to create __init__.py inside a directory to convert this directory to a package.

6. To use the modules defined inside the package Employees, we must have to import this in our python source file. Let's create a simple python source file at our home directory (/home) which uses the modules defined in this package.

Test.py

```
import Employees
print(Employees.getNames())
```

GKTCS INNOVATIONS

6. To use the modules defined inside the package Employees, we must have to import this in our python source file. Let's create a simple python source file at our home directory (/home) which uses the modules defined in this package.

Test.py

```
import Employees
print(Employees.getNames())
```