Python_r_[Logging & Debugger]_[Day 24](DATA MINDS)

September 23, 2023

TOPIC - LOGGING

LOGGING - IT GENRALLY CREATES A FILE IN WHICH IT HOLDS ALL THE DATA THAT WE WANT TO USE WHENEVER WE PERFOR THE INVESTIGATION IN FUTURE .

LIKE IF WE PRINT SOMETHING SO OBIOUSLY IT PRINT ON THE CONSOLE BUT AFTER USING THE LOGGING IT DOES NOT PRINT ON CONSOLE IT PRINT OR STORE IN A FILE, IN PRODUCTION GRADE CODE WE ALWAYS TRY TO WRITE OR PUT CODE IN LOGGING MANNER WITHOUT USING THE PRINT ( ) FUNCTION.

WE SHOULD REPLACE THE PRINT ( ) FUNCTION BY USING LOGGING MODULE.

NOTE - BY USING THE LOGGING WE CREATE THE FILE OF OUTPUT THAT PRINT IN CONSOLE AND IT STORED IN HARD DISK FOR A VERY VERY LONG TIME

[5]:
```
import logging
```

FIRST WE DESCRIBE THE BASIC CONFIGURATION OF LOGGING -

[6]:
```
logging.basicConfig(filename="test.log" , level=logging.INFO)
```

[7]:
```
logging.info("This is Virat Tiwari")
```

[8]:
```
logging.info("I read and write Data Science Code ")
```

[9]:
```
logging.info("Data Science is the art of uncovering hidden patterns behind the␣
↪data")
```

NOTE - IN PRINT ( ) FUNCTION WE HAVE ONLY A PRINT FUNCTION THAT SIMPLY PRINT THE OUTPUT IN CONSOLE BUT IN LOGGING MODULE WE HAVE DIFFERNT KINDS OF LEVELS OR WE SAY TEMPLATES ALSO - :

1) NOTESET,

2) DEBUG,

3) INFO,

4) WARNING,

5) ERROR,

6) CRITICAL

[10]:
```
logging.warning("This is my warning message ")
```

[11]:
```
logging.error("We found error in the name of Virat")
```

[12]:
```
logging.critical("This is critical case of logging")
```

[ ]:
```
logging.shutdown()
```

[ ]:
```
import logging
```

[ ]:
```
logging.basicConfig(filename="Data.log",level=logging.DEBUG,format="%(asctime)s␣
↪%(message)s")
```

[ ]:
```
logging.info("This is logging info file")
logging.error("This is my error file")
logging.warning("This is my warning")
logging.critical("This is critical file")
```

[ ]:
```
logging.shutdown()
```

[14]:
```
import logging
```

[15]:
```
logging.basicConfig(filename="New.log",level=logging.DEBUG,format="%(asctime)s␣
↪%(levelname)s %(message)s")
```

[16]:
```
logging.info("This is logging info file")
logging.error("This is my error file")
logging.warning("This is my warning")
logging.critical("This is critical file")
```

PRACTICAL APPLICATION OF LOGGING - :

[25]:
```
l = [1,2,3,[9,8,7],"Virat Tiwari","Data Science"]
```

[26]:
```
l
```

[26]:
```
[1, 2, 3, [9, 8, 7], 'Virat Tiwari', 'Data Science']
```

[30]:
```
l1_int=[]
l2_str=[]
for i in l:
    logging.info("We are iterating through our list and local varables is␣
↪i"+str(l))
    if type(i)==list:
        logging.info("I am inside if statement and i am trying to check list␣
↪type")
        for j in i:
            logging.info("I am in another for loop for list inside element")
            if type(j)==int:
                logging.info("I am inside the if statement")
                l1_int.append(j)
    elif type(i)==int:
        l1_int.append(i)
    else:
        if type(i)==str:
            l2_str.append(i)
```

[31]:
```
l1_int
```

[31]:
```
[1, 2, 3, 9, 8, 7]
```

[29]:
```
l2_str
```

[29]:
```
['Virat Tiwari', 'Data Science']
```

[32]:
```
type(l)
```

[32]:
```
list
```

THANK YOU SO MUCH !!

YOURS VIRAT TIWARI :)