With modern Python Exceptions, you don't need to abuse .message, or override .\_\_str\_\_() or .\_\_repr\_\_() or any of it. If all you want is an informative message when your exception is raised, do this:

class MyException(Exception):

pass

raise MyException("My hovercraft is full of eels")

That will give a traceback ending with MyException: My hovercraft is full of eels.

If you want more flexibiilty from the exception, you could pass a dictionary as the argument:

raise MyException({"message":"My hovercraft is full of animals", "animal":"eels"})

However, to get at those details in an except block is a bit more complicated; they are stored in the args attribute, which is a list. You would need to do something like this:

try:

raise MyException({"message":"My hovercraft is full of animals", "animal":"eels"})

except MyException as e:

details = e.args[0]

print(details["animal"])

It is still possible to pass in multiple items into the exception, but this will be deprecated in the future. If you do need more than a single piece of information, then you should consider fully subclassing Exception.