Python Error Handling

Syntax Errors

A syntax error is incorrect in the context of the language. Code containing syntax errors cannot be executed by the Python interpreter. Here are a few examples:

```
# forgetting :
def my_func()
    print("hello from my_func")
```

```
# not closing a bracket
a = [1, 2, 3
```

```
# missing comma
a = [1, 2 3, 4]
```

Runtime Errors

Runtime errors occur when syntactically correct code does something wrong (like attempt to access a list out of bounds, or divide an integer by zero).

We have seen these before:

```
a = [3, 7]
a[2]
```

```
b = {'cupcakes' : 7, 'brownies' : 2}
b['cookies']
```

Exceptions

- Runtime errors generate exceptions
- Exceptions can potentially be caught
- Uncaught exceptions propagate up to the interpreter, which halts execution and displays the information in a traceback
- Python uses a try/except model for error handling

```
f = open('thisfiledoesntexist.txt')
```

```
try:
    f = open('thisfiledoesntexist.txt')
except IOError:
    print("That filename doesn't exist.")
```

Here, we caught the exception raised when `open` could not find the file. The `try-except` syntax allows us to control what happens when an exception occurs.

Catching multiple exceptions

Specific exceptions can be handled by specifying the exception type after `except`.

```
try:
    5/0
except IOError:
```

```
    print('I/O error')
except ZeroDivisionError:
    print('Zero division error')
except Exception as e:
    # here we get access to the exception object
    print(e)
```

Raising exceptions

From `mymodule5.py`:

```
import types

def summation(a,b):
    """
    Returns the sum of numbers between, and including, a and b.
    """

    if (type(a) != types.IntType or type(b) != types.IntType):
        raise ValueError('Expected integers')

    total = 0
    for n in range(a,b+1):
        total += n
    return total
```

Using:

```
import mymodule4
mymodule4.summation(1,'hello')
```

```
import mymodule5
mymodule5.summation(1,'hello')
```

Recommended Reading

- Python Tutorial: Errors and Exceptions
- Chapter 33: Exception Basics
- Chapter 34: Exception Coding Details