Exceptions

Informatics 1 for Biomedical Engineers
Tutor Session 8

KTI, Knowledge Technologies Institute

12. Dezember 2016

Today's Topics

1. What are exceptions
2. What types are there
3. When to use them

Student Goals

- Know how to use try-except blocks
- How to detect exception types

What are exceptions

- Errors crash your program
  - “Not good!”
- Exceptions tell your program what to do if things go wrong
- Defined by 2 blocks
  - **try:** - your code that might break
  - **except:** - what to do when things go wrong

What parts of your code can break?

- ALL OF THEM!

What parts of your code can break?

- ALL OF THEM!
- At least whenever you work with one of the following
  - User input
  - Files
  - External resources (like websites or servers)

Simple error example

```
1  # Your name and matr_nr
2  name = "Johnny Bravo"
3  mat_nr = 4396469
4  print("Your name is " + name + " and your matr number is " + mat_nr)
5  # Poof!
```

```
1  >>> TypeError: Can't convert 'int' object to str implicitly
```

Fixed Simple example

```
1   # Your name
2   try:
3       name = "Johnny␣Bravo"
4       mat_nr = 4396469
5       print("Your␣name␣is␣" + name + "␣and␣your␣matr␣number␣is␣" + mat_nr)
6   except:
7       print("Johnny␣never␣learned␣to␣code....")
8       name = "Johnny␣Bravo"
9       mat_nr = str(4396469)
10      print("Your␣name␣is␣" + name + "␣and␣your␣matr␣number␣is␣" + mat_nr)
```

Error Types

- Plain except catches all error types (Except SyntaxErrors)
- Python supports different types of errors (built-in)[1]
- Handle each error individually
- Can be chained
- SyntaxErrors usually can't be caught!

Syntax:

```
1    except <ErrorType>:
```

[1] https://docs.python.org/3/library/exceptions.html

Try different error Types

```
1    # Modify the name to trigger the different errors
2  try:
3      name = "Johnny Bravo" # try Johnny Bravo or Johnny
4      mat_nr = 4396469 # try str(4396469)
5      print("Your name is " + name + " and your matr number is " + mat_nr)
6  except SyntaxError:
7      print("Syntax Error: We can't catch those...")
8  except NameError:
9      print("Name Error: Variable probably isn't defined yet...")
10 except TypeError:
11     print("Type Error: Number should be text to be printable...")
12 else:
13     print("Johnny is a coding god!")
```

Raising Errors

- Manually trigger exceptions
- Feedback for others using your programs
- Stick to standard error types
  - Custom error types exist but are not relevant for now

Syntax:

```
1  # raise <ErrorType>("<ErrorMessage>")
2
3  name = input("Please␣enter␣your␣name;␣")
4
5  if name = "Johnny␣Bravo":
6    raise InputError("Rather␣unlikely...")
```

Student Task

Task: Ask the user for his full name (first and last) and his age

- First ask for the name then the age
- Use input() to ask the user
- Catch errors and inform the user about any mistakes
- Ask until all input is valid
- Raise error(AssertionError) if age is < 5

Student Task: Ask for the name

```
1  while(True):
2    try:
3      first_name, last_name = input("What␣is␣your␣name?␣").split("␣")
4    except ValueError:
5      print("We␣need␣your␣first␣and␣your␣last␣name")
6    except:
7      print("'Something'␣went␣wrong")
8    else:
9      break
```

Student Task: Ask for the age

```
1   while(True):
2     try:
3       age = int(input("What␣is␣your␣age?␣"))
4       if age < 5:
5           raise AssertionError()
6     except ValueError:
7       print("That␣is␣not␣a␣number")
8     except AssertionError:
9       print("You␣can␣already␣read␣and␣write?")
10    except:
11      print("'Something'␣went␣wrong")
12    else:
13      break
```