**Python 2 to 3 Changes**

**Numbers**

* Python 3: Long and int are the same
* Integer literals “L” does not exist anymore
* Octal literals: use “0o” everywhere instead of “0”
* 2to3 does this

**Strings**

* Always convert text to Unicode. Convert string literals that are text to u’abc’. **Label text as u and data as b**
* Py2: Ascii (default encoding) Py3: utf-8 default encoding.
* str.encode() converts Unicode to bytes. Str.decode() converts bytes to Unicode. Use them while combining Unicode and byte strings.
* Find everywhere you use binary data, File I/O, string literals, Arrays. If it’s a binary literal use b’abc’ syntax.
* Python 3’s -bb flag will help you track down places in your code where you are comparing byte strings and Unicode strings.
* Text and binary data are different.
* Basestring no longer exists in python3. Identify if your data is text or binary and treat accordingly
* I/O
* Python2
  + Open(…).read() always returns byte string
  + Text strings are byte strings also
* Python 3
  + Open(…,’r’).read() always returns Unicode
  + Open(…,’rb’).read() always returns byte string
* Interning
  + Python2: intern(‘abc’)
  + Python3: sys.intern(‘abc’)

In Python 2, implicit str type is ASCII. But in Python 3.x implicit str type is Unicode.

print(type('default string '))

print(type(b'string with b '))

'''

Output in Python 2.x (Bytes is same as str)

<type 'str'>

<type 'str'>

Output in Python 3.x (Bytes and str are different)

<class 'str'>

<class 'bytes'>

'''

Python 2.x also supports Unicode

print(type('default string '))

print(type(u'string with b '))

'''

Output in Python 2.x (Unicode and str are different)

<type 'str'>

<type 'unicode'>

Output in Python 3.x (Unicode and str are same)

<class 'str'>

<class 'str'>

'''

**Exception Handling**

* Py2: except ValueError, e: print e. Py3: except ValueError as e: print e
* Py2: raise Exception, ‘argument’. Py3: raise Exception(‘argument’)
* 2to3 helps with both.

Classes

* Py2: Class A:

\_\_metaclass\_\_=Singleton

Py3: Class A(metaclass=Singleton):

Pass

Iterators

* Py2: def next(self):

Py3: def \_\_next\_\_(self):

* Py2: itr.next()

Py3: next(itr)

* Py2: Range() returns a list, xrange returns an iterator.

Py3: range() returns an iterator, xrange is no longer there.

* 2to3 does all of these
* Dictionaries:
  + Py2: keys(), values(), items() all returns lists.

Iterkeys(), itervalues(), iteritem() for iterators.

* + Py3: keys(), values(), items() for iterators.

Str and BOOL methods

* Remove \_\_unicode\_\_
* Change \_\_str\_\_ to return Unicode
* 2to3 does this
* Bool(obj) calls obj.\_\_nonzero\_\_() in py2, obj.\_\_bool\_\_ in py3. Rename nonzero to bool. 2to3 does this.

Files: open and read

* Use io.open instead of open
* Libraries won’t help much

Input

* Raw\_input() has been renamed to input()
* Py2’s input() doesnot exist anymore.
* 2to3 does this

Execfile

* Doesn’t exist in Python3.
  + To get this behavior exec(open(path).read())
* 2to3 does this

StringIO

* Py3: StringIO has been reorganized into Io.BytesIO for binary, Io.StringIO for Unicode text
* 2to3 won’t help

Collections

* Some classes moved to collections
* 2to3 covers most of this

DBM

* Database modules reorganized
* 2to3 will port for you

Commands and subprocess

* Commands module replaced with subprocess module
* 2to3 will do this

Web Libraries

* http, html and xmlrpc libraries are restructured. Modules changes names, contents didn’t.
* 2to3 will do this

Itertools

* Function names changed inside itertools
  + Itertools.ifilterfalse-> Itertools.ifilterfalse
  + Itertools.izip\_longest->itertools.zip\_longest
* 2to3 does this

Miscellaneous Libraries

* Tkinter, socketserver, configparser, queue, copyreg, repr->reprlib, libraries have been renamed.reorganized
* 2to3 does this

Print

* Print is now a function
* 2to3 does this

Repr

* `x` , backtick syntax for repr has been removed in python3
* 2to3 does this

Relative imports and reload

* Import one, from one import name-> from . import one, from .one import name
* 2to3 does this
* Py2: reload is a builtin function , Py3: from importlib import reload, use importlib.reload
* No automated help for this

Dictionaries

* Dictionary behavior has changes, returns iterators instead of list
* 2to3 does this

Range

* Py3: returns iterator, Py2: returns list
* L=range(5) -> l=list(range(5))
* Xrange() is gone now

Functional Operators

* Map(), zip() and filter() all returns iterators.
* 2to3 does this
* Cmp() function replaced with key()
* \_\_cmp\_\_() replaced with \_\_lt\_\_()
* No automated tools for this

