Boolean (True/False) Values

Python 2

home (../handouts.html)

Introduction: Booleans (True/False) Values

Every object can be converted to boolean (True/False): an object is **True** if it is "non-zero".

```
counter = 5
if counter:                              # True
    print 'this int is not zero'
else:
    print 'this int is zero'


mystr = ''                               # empty string

if mystr:                                # False
    print 'this string has characters'
else:
    print 'this string is empty'


var = ['a', 'b', 'c']
if var:                                  # True
    print 'this container has elements'
else:
    print 'this container is empty'
```

Objectives for the Unit: Booleans

- Read any object in *boolean context* (i.e., with **if** or **while**) to see if it is empty

Summary for Object: Boolean

A *boolean* object can be True or False. All objects can be converted to boolean, meaning that all objects can be seen as True or False in a

boolean context.

```
print type(True)              # <type 'bool'>
print type(False)             # <type 'bool'>
```

bool() converts objects to boolean.

```
print bool(['a', 'b', 'c'])    # True
print bool([])                # False
```

if and *while* induce the bool() conversion

So when we say **if var:** or **while var:**, we're testing if **bool(var) == True**, i.e., we're checking to see if the value is a **True** value

```
print bool(5)                 # True
print bool(0)                 # False
```

if and **while** induce the boolean conversion -- an object is evaluated in *boolean context*

```
mylist = [0, 0]

if mylist:                            # not empty, so True in boolean conte
    print 'that list is not empty'


yourlist = [1, 2, 3]

while yourlist:             # True as long as yourlist has elements
    x = yourlist.pop()      # remove element from the end of yourlist
    print x
```

Boolean quiz

Quiz yourself: look at the below examples and say whether the value will test as

True or **False** in a boolean expression. Beware of tricks!

Remember the rule: if it represents a 0 or empty value, it is False. Otherwise, it is **True**.

```
var   = 5
var2  = 0
var3  = -1
var4  = 0.0000000000000001
varx  = '0'
var5  = 'hello'
var6  = ""
var7  = '    '
var8  = [   ]
var9  = ['hello', 'world']
var10 = [0]
var11 = {0:0}
var12 = {}
```

==
<=H>Booleans: quiz answers

```
var   = 5                       # bool(var):   True
var2  = 0                       # bool(var2):  False
var3  = -1                      # bool(var3):  True (not 0)
var4  = 0.0000000000000001      # bool(var4):  True
varx =  '0'                     # bool(varx):  True (not empty string)
var5  = 'hello'                 # bool(var5):  True
ver6  = ""                      # bool(var6):  False
var7  = '    '                  # bool(var7):  True (not empty)
var8  = [   ]                   # bool(var8):  False
var9  = ['hello', 'world']      # bool(var9):  True
var10 = [0]                     # bool(var10): True (has an element)
var11 = {0:0}                   # bool(var11): True (has a pair)
var12 = {}                      # bool(var12): False
```

any() all() and in

any() with a sequence checks to see if any elements are True; **all()** asks whether they are all True; **in** can be used to see if a value exists in a sequence.

any(): are any of these True?

```
mylist = [0, 0, 5, 0]

if any(mylist):
    print 'at least one item is True'
```

all(): are all of these True?

```
mylist = [5, 5, 'hello', 5.0]

if all(mylist):
    print 'all items are True'
```

in with a list: is this value anywhere within this list?

```
mylist = ['highest', 'lowest']

user_choice = 'lowest'

if user_choice not in mylist:
    print 'please enter "higest" or "lowest"'
```