Unicode Characters and Strings

ASCII

American Standard Code for Information Interchange.

DEC	HEX	OCT	BIN	CHAR
0	0x00	000	00000000	
⋮	⋮	⋮	⋮	∫
127	0x7F	177	11111111	

Representing Simple Strings

Each character is represented by a number between 0 and 256 stored in 8 bits of memory

The ord( ) function tells us the numeric value of a simple ASCII character

```
print ( ord ('H') ) → 72
print ( ord ('e') ) → 101
print ( ord ('\n') ) → 10
```

小寫的總小於大寫的英文字母

Unicode 的字元含量大於 ASCII 的字含量

Multi-Byte Characters

To represent the wide range of characters computers must handle we represent characters with more than one byte

UTF-16 : Two bytes
UTF-32 : Four bytes

UTF-8 : One ~ Four bytes

- Upwards compatible with ASCII
- Automatic detection between ASCII and UTF-8
- UTF-8 is recommanded practice for encoding data to exchanged between systems

Two kinds of Strings in Python

	x = 'Hello'	x = u'Hello'
Python 2.7	str	str
Python 3.5	str	unicode

in Python 3, all strings are Unicode

Python 2 Versus Python 3

	x = b'Hello'
Python 2.7	str
Python 3.5	bytes

Python 3 and Unicode

In python 3, all strings internally are Unicode.

Working with string variables in Python programs and reading data from files usually "just works".

When we talk to a network resource using sockets or talk to a database we have to encode and decode data (usually to UTF-8)

Python Strings to Bytes

When we read data from an external resource, we must decode it based on

the character set so it is properly represented in Python 3 as a string.

```
While True:                          # Bytes
    data = mysock.recv(512)
    if ( len(data) < 1 ):
        break
    mystring = data.decode()         # utf-8 or ASCII
    print(mystring)                  # bytes to unicode
```

An HTTP Request in Python

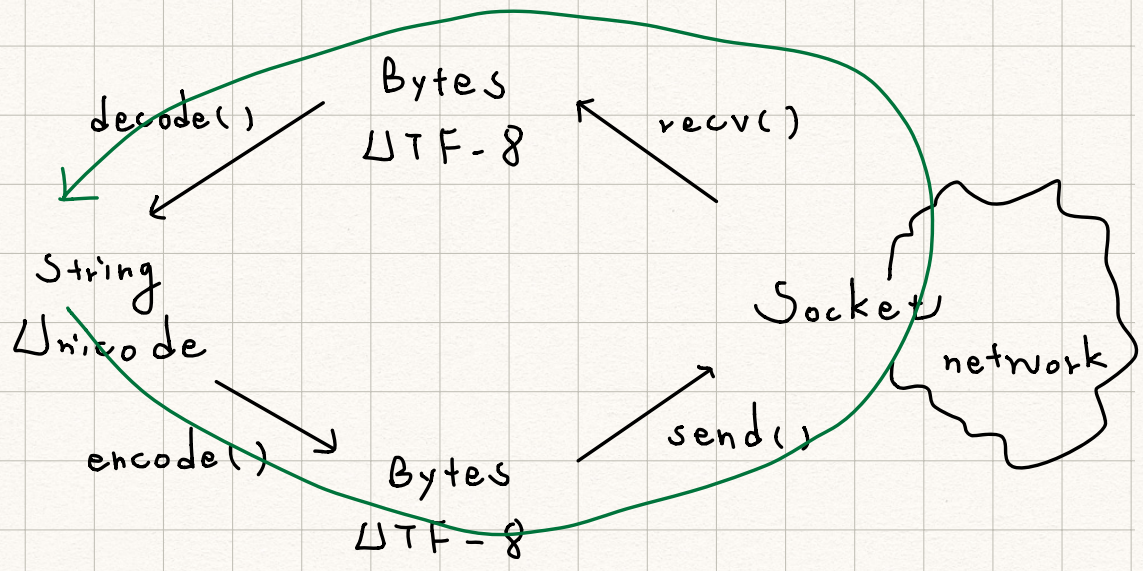
```
import socket
mysock = socket.socket(socket.AF_INET
                       ,socket.SOCK_STREAM
mysock.connect(('data.pr4e.org', 80))

cmd = 'GET http://data.pr4e.org/romeo.txt HTTP
      /1.0\r\n\r\n'.encode()         # utf-8
mysock.send(cmd)                     # Bytes
```

```
while True:
    data = mysock.recv(512)
    if (len(data) < 1):
        break
    print(data.decode())
```

unicode

```
mysock.close()
```



Retrieving Web Pages

Using urllib in Python

Since HTTP is common, we have a library that does all the socket work for us and makes web pages look like a file.

```
import urllib.request, urllib.parse,
       urllib.error

fhand = urllib.request.urlopen
      ('http://data.pr4e.org/romeo.txt')
for line in fhand:
      print(line.decode().strip())
```

file handle

出來的 data 不會有 header

Like a File

```
import urllib.request, urllib.parse,
       urllib.error
fhand = urllib.request.urlopen
      ('http://data.pr4e.org/romeo.txt')

counts = dict()
for line in fhand:
      words = line.decode().split()
      for word in words:
            count[word] = counts.get(word,0)
                                      + 1
print(counts)
```

Reading Web Pages

```
import urllib.request, urllib.parse,
       urllib.error
fhand = urllib.request.urlopen
      ('http://data.pr4e.org/romeo.txt')

for line in fhand:
      print(line.decode().strip())
```

↓ html code

a href 後面接超連結網址

Parsing Web Pages

What is Web Scraping?

When a program or script pretends to be a browser and retrieves web pages, looks at those web pages, extracts information, and then looks at more web pages.

Crawling 爬蟲

Why Scrape?

Pull data - particularly social data
who links to who?

Get your own data back out of some system that has no "export capability"

Monitor a site for new information

Spider the web to make a database for a search engine

Scraping Web Pages

There is some controversy about web page scraping and some sites are a bit snippy about it.

Republishing copyright information is not allowed.

Violating terms of service is not allowed

The Easy Way - Beautiful Soup

You could do string searches the hard way.

Or use the free software library called BeautifulSoup from www.crummy.com

Beautiful Soup Programs

```
import urllib.request, urllib.parse,
       urllib.error
from   bs4 import BeautifulSoup

url = input('Enter')
html = urllib.request.urlopen(url).read()
soup = BeautifulSoup(html, 'html.parser')
```

※ Retrieve all of the anchor tags

超連結的部分

```
tags = soup('a')
for tag in tags:
      print(tag.get('href', None))
```

Summary

The TCP/IP gives us pipe/sockets between applications

We designed application protocals to make use of these pipes

HyperText Transfer Protocal (HTTP) is a simple yet powerful protocal

Python has good support for sockets, HTTP, and HTML parsing