Digital Technologies and Value Creation

Dr. Philippe Blaettchen
Bayes Business School (formerly Cass)

Learning objectives of today

Goals: Understand some key technologies behind information exchange on the internet

- What protocols are there to access data online?
- How are webpages built and how can they be retrieved in Python?
- How is data represented on the internet?

How will we do this?

- We will walk through the most important concepts and tools
- We will see how Python can be used to work with internet protocols and technologies

Note

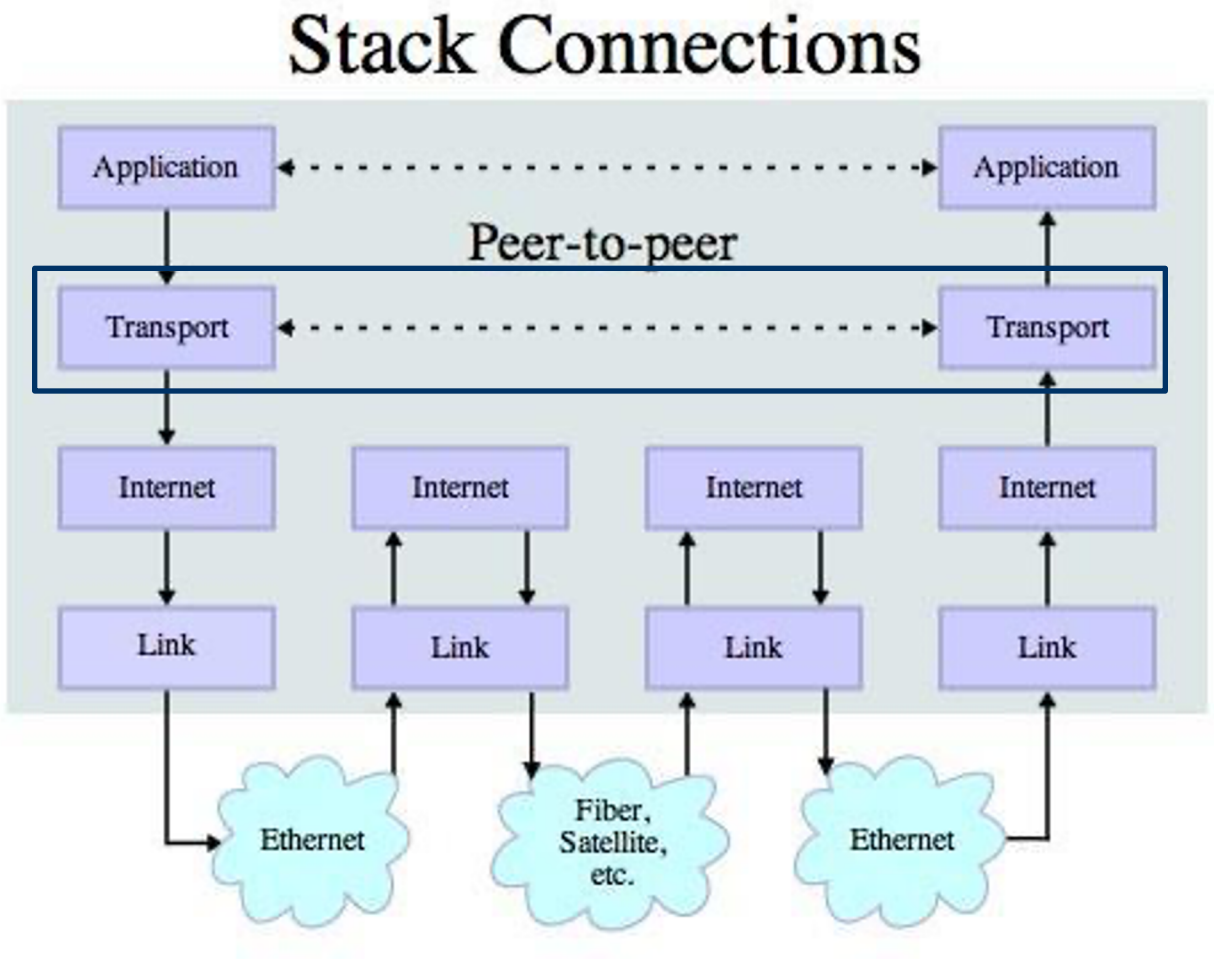
The slides and code for the online part of the lecture draw heavily from materials freely provided by Charles Severance: https://www.py4e.com/lessons

TCP and Sockets

Protocols

- A set of rules that all parties follow so we can predict each other's behavior
- And not bump into each other
  - On two-way roads in USA, drive on the right-hand side of the road
  - On two-way roads in the UK, drive on the left-hand side of the road

Transmission Control Protocol (TCP)

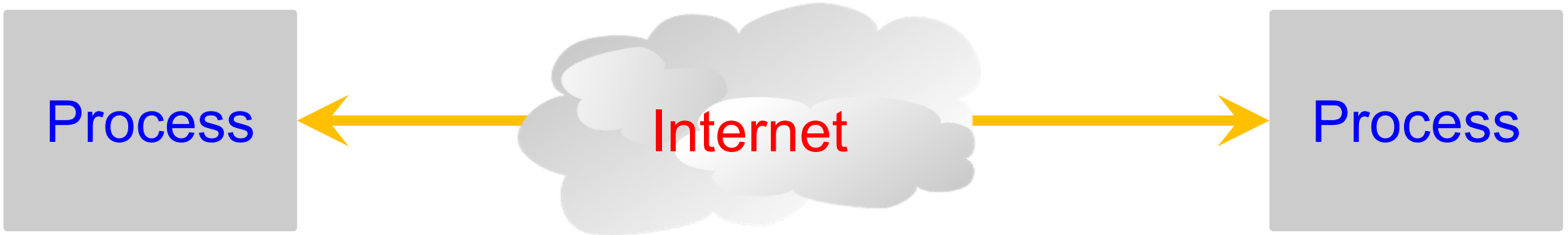


- Built on top of IP (Internet Protocol)
- Assumes IP might lose some data - stores and retransmits data if it seems to be lost
- Handles “flow control” using a transmit window
- Provides a nice reliable pipe

Source: http://en.wikipedia.org/wiki/Internet_Protocol_Suite

TCP Connections – Sockets

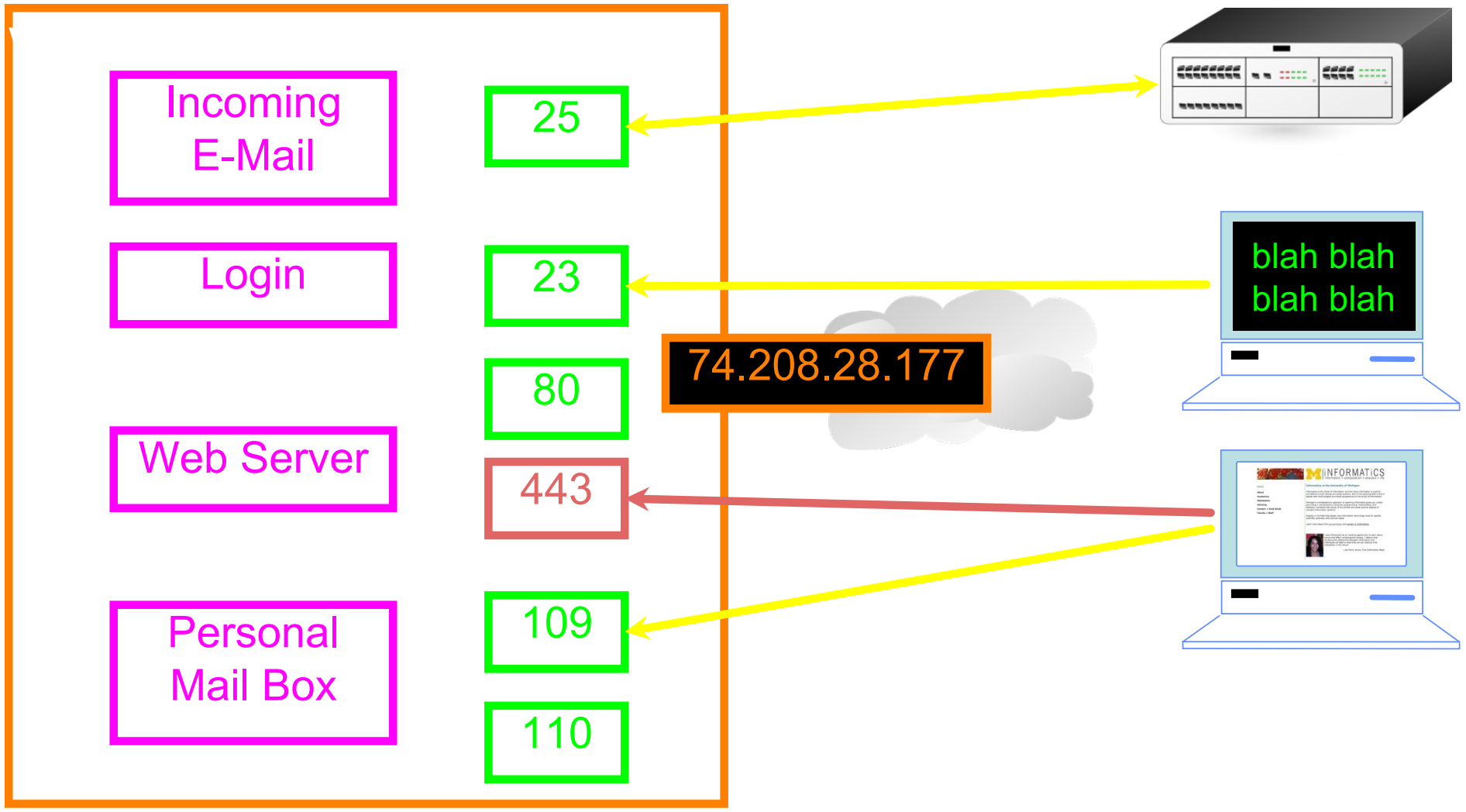
Internet/network socket: endpoint of communication flow across a TCP/IP based computer network



TCP Connections – Port Numbers

- A port is an application-specific or process-specific software communications endpoint
- It allows multiple networked applications to coexist on the same server
- There is a list of well-known TCP port numbers

Port numbers



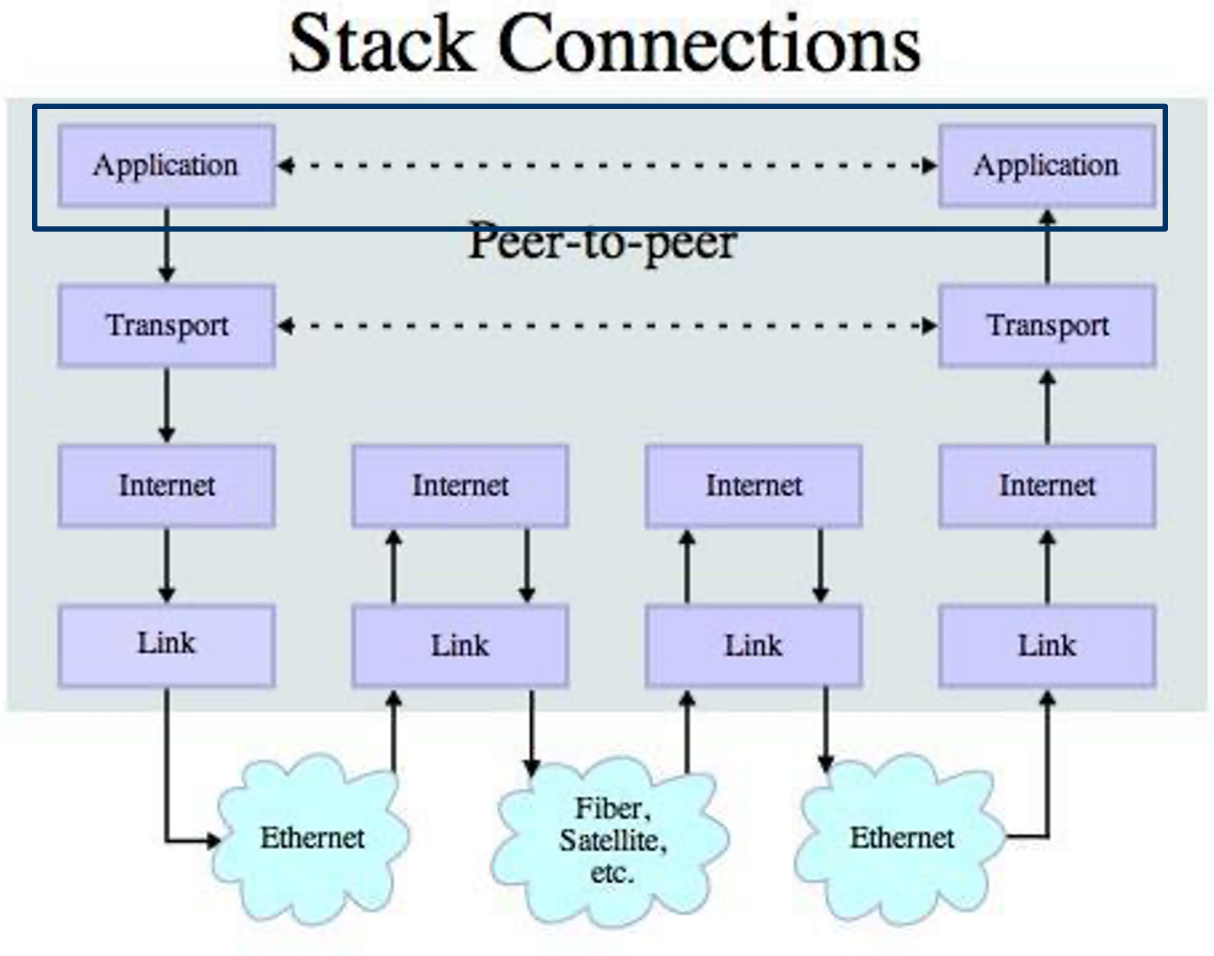
BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Let’s talk to a server in Python

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

HTTP

Application Protocols in general



- We got the socket from TCP
- Now, need to decide what to do with that socket
- Each type of socket brings its own application protocols

Source: http://en.wikipedia.org/wiki/Internet_Protocol_Suite

Hypertext Transfer Protocol Secure (HTTPS)

- Extension of HTTP (technically speaking, use of HTTP on encrypted SSL/TLS connection)
- Used for secure communication
- Most sites use this as the standard today

HTTP(S) example

`https://docs.python.org/3/installing/index.html`

protocol host path document

Getting data from the server

- Each time the user clicks on a link (in HTML: an anchor tag with an href= value), the browser makes a connection to the web server and issues a “GET” request
  - This signals that the browser wants to GET the document at the specified URL

- The server returns the (HTML) document to the browser, which formats and displays the document to the user

HTTP(S) specification of a request

```
5 Request

   A request message from a client to a server includes, within the
   first line of that message, the method to be applied to the resource,
   the identifier of the resource, and the protocol version in use.

        Request       = Request-Line              ; Section 5.1
                        *(( general-header        ; Section 4.5
                         | request-header         ; Section 5.3
                         | entity-header ) CRLF)  ; Section 7.1
                        CRLF
                        [ message-body ]          ; Section 4.3

5.1 Request-Line

   The Request-Line begins with a method token, followed by the
   Request-URI and the protocol version, and ending with CRLF. The
   elements are separated by SP characters. No CR or LF is allowed
   except in the final CRLF sequence.

        Request-Line   = Method SP Request-URI SP HTTP-Version CRLF
```

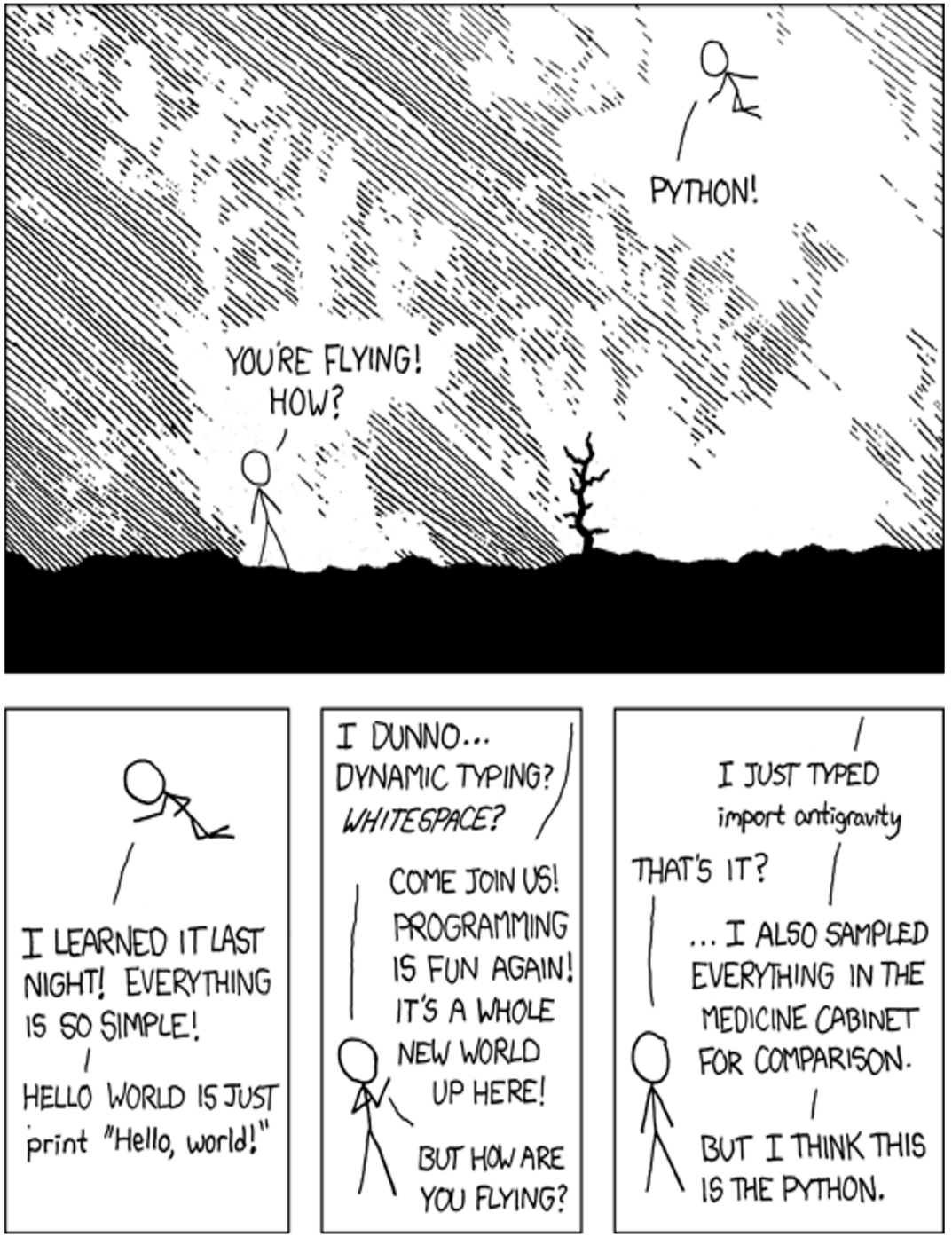
Source: http://www.w3.org/Protocols/rfc2616/rfc2616.txt

HTTP Requests in Python

Retrieving and parsing web pages – Regular Expressions and BeautifulSoup

Simplified retrieval

- Of course, we could do everything ourselves, sending out GET requests, listening to the response, etc.
- Luckily, **requests** already does most of the tedious work for us (the beauty of Python…)



Source: xkcd.com

Parsing HTML – also known as Web Scraping

- **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)

- When moving from web page to web page based on the links found, we usually call this “spidering the web” or “web crawling” – more on this in class

Why do we scrape?

- **Because we want to get data**
- Some systems don’t have explicit “export” capabilities, and scraping may be faster than manually gathering information
- We may want to monitor sites for new information
- Crawling the web to create a search engine

The legal parts

- The legal situation of scraping can be complicated
- What is never allowed is to republish copyrighted information (unless there is some license granting you the right to, but this usually involves citation)
- It is generally forbidden to violate terms of service
    - When you create an account, you explicitly agree to the terms of service of a site
    - Even without an account, you implicitly agree
    - And even when not illegal, providers may block your account or IP!
- A good first place to check: robots.txt
- Generally, be reasonable: even if there is no problem scraping a site in principle, remember that server resources are finite!

We have retrieved a web page – now what?

We could use regular expressions to analyze the website text:

- regular expression (also “regex” or “regexp”) is a means for matching strings
- A regular expression is written in a formal language
- The formal language can be interpreted by a regular expression processor.

A few notes:

- Very powerful and quite cryptic
- Fun once you understand them
- Regular expressions are a language unto themselves
- A language of “marker characters” - programming with characters
- It is kind of an “old school” language - compact

Regular expressions – the main ones

^	Matches the **beginning** of a line
$	Matches the **end** of the line
.	Matches **any** character
\s	Matches **whitespace**
\S	Matches **any non-whitespace** character
*	Repeats a character **zero or more times**
*?	Repeats a character **zero or more times (non-greedy)**
+	Repeats a character **one or more times**
+?	Repeats a character **one or more times (non-greedy)**
[aeiou]	Matches a **single character** in the listed set
[^XYZ]	Matches a **single character not** in the listed set
[a-z0-9]	The set of characters can include **a range**
(	Indicates **where string extraction is to start**
)	Indicates **where string extraction is to end**

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Regular expressions in Python

So how is this relevant?

Installing Python Modules

Email: distutils-sig@python.org

As a popular open source development project, Python has an active supporting community of contributors and users that also make their software available for other Python developers to use under open source license terms.

This allows Python users to share and collaborate effectively, benefiting from the solutions others have already created to common (and sometimes even rare!) problems, as well as potentially contributing their own solutions to the common pool.

This guide covers the installation part of the process. For a guide to creating and sharing your own Python projects, refer to the distribution guide.

Note: For corporate and other institutional users, be aware that many organisations have their own policies around using and contributing to open source software. Please take such policies into account when making use of the distribution and installation tools provided with Python.

Key terms

- `pip` is the preferred installer program. Starting with Python 3.4, it is included by default with the Python binary installers.
- A *virtual environment* is a semi-isolated Python environment that allows packages to be installed for use by a particular application, rather than being installed system wide.
- `venv` is the standard tool for creating virtual environments, and has been part of Python since Python 3.3. Starting with Python 3.4, it defaults to installing `pip` into all created virtual environments.
- `virtualenv` is a third party alternative (and predecessor) to `venv`. It allows virtual environments to be used on versions of Python prior to 3.4, which either don't provide `venv` at all, or aren't able [redacted] into created environments.
- The Python Package Index is a public repository of open source licensed packages made available for use by other Python users.
- the Python Packaging Authority is the group of developers and documentation authors responsible for the maintenance and evolution of the standard packaging tools and the associated metadata and file format standards. They maintain a variety of tools, documentation, and issue trackers on both GitHub and Bitbucket.

a.reference.external 169.41 × 18

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Luckily, there is an easy way - BeautifulSoup

Generally, it's save to assume that someone has already built a fantastic library for Python!

Data representation in computers and on the web

Text and strings – American Standard Code for Information Interchange (ASCII)

- Each character is represented by a number between 0 and 256 stored in 8 bits of memory
- We refer to "8 bits of memory" as a "byte" of memory – (i.e. my disk drive contains 1 Terabyte 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
>>>
```

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Text and strings – American Standard Code for Information Interchange (ASCII)

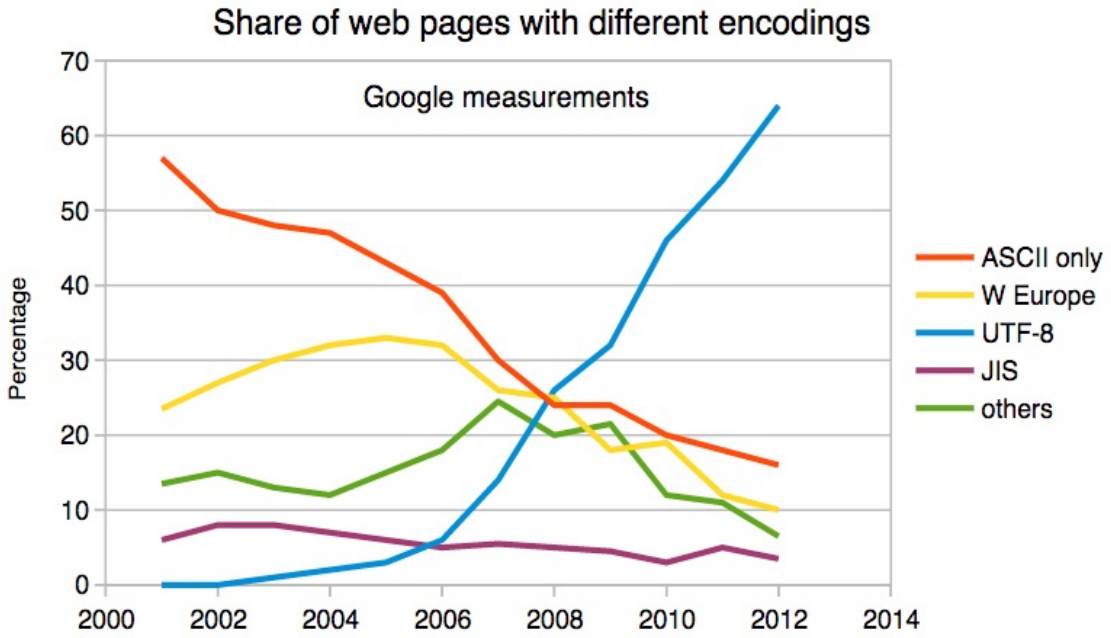
Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char	Dec	Hex	Oct	Bin	Char
0	0x00	000	0000000	NUL	32	0x20	040	0100000	space	64	0x40	100	1000000	@	96	0x60	140	1100000	`
1	0x01	001	0000001	SOH	33	0x21	041	0100001	!	65	0x41	101	1000001	A	97	0x61	141	1100001	a
2	0x02	002	0000010	STX	34	0x22	042	0100010	"	66	0x42	102	1000010	B	98	0x62	142	1100010	b
3	0x03	003	0000011	ETX	35	0x23	043	0100011	#	67	0x43	103	1000011	C	99	0x63	143	1100011	c
4	0x04	004	0000100	EOT	36	0x24	044	0100100	$	68	0x44	104	1000100	D	100	0x64	144	1100100	d
5	0x05	005	0000101	ENQ	37	0x25	045	0100101	%	69	0x45	105	1000101	E	101	0x65	145	1100101	e
6	0x06	006	0000110	ACK	38	0x26	046	0100110	&	70	0x46	106	1000110	F	102	0x66	146	1100110	f
7	0x07	007	0000111	BEL	39	0x27	047	0100111	'	71	0x47	107	1000111	G	103	0x67	147	1100111	g
8	0x08	010	0001000	BS	40	0x28	050	0101000	(	72	0x48	110	1001000	H	104	0x68	150	1101000	h
9	0x09	011	0001001	TAB	41	0x29	051	0101001	)	73	0x49	111	1001001	I	105	0x69	151	1101001	i
10	0x0A	012	0001010	LF	42	0x2A	052	0101010	*	74	0x4A	112	1001010	J	106	0x6A	152	1101010	j
11	0x0B	013	0001011	VT	43	0x2B	053	0101011	+	75	0x4B	113	1001011	K	107	0x6B	153	1101011	k
12	0x0C	014	0001100	FF	44	0x2C	054	0101100	,	76	0x4C	114	1001100	L	108	0x6C	154	1101100	l
13	0x0D	015	0001101	CR	45	0x2D	055	0101101	-	77	0x4D	115	1001101	M	109	0x6D	155	1101101	m
14	0x0E	016	0001110	SO	46	0x2E	056	0101110	.	78	0x4E	116	1001110	N	110	0x6E	156	1101110	n
15	0x0F	017	0001111	SI	47	0x2F	057	0101111	/	79	0x4F	117	1001111	O	111	0x6F	157	1101111	o
16	0x10	020	0010000	DLE	48	0x30	060	0110000	0	80	0x50	120	1010000	P	112	0x70	160	1110000	p
17	0x11	021	0010001	DC1	49	0x31	061	0110001	1	81	0x51	121	1010001	Q	113	0x71	161	1110001	q
18	0x12	022	0010010	DC2	50	0x32	062	0110010	2	82	0x52	122	1010010	R	114	0x72	162	1110010	r
19	0x13	023	0010011	DC3	51	0x33	063	0110011	3	83	0x53	123	1010011	S	115	0x73	163	1110011	s
20	0x14	024	0010100	DC4	52	0x34	064	0110100	4	84	0x54	124	1010100	T	116	0x74	164	1110100	t
21	0x15	025	0010101	NAK	53	0x35	065	0110101	5	85	0x55	125	1010101	U	117	0x75	165	1110101	u
22	0x16	026	0010110	SYN	54	0x36	066	0110110	6	86	0x56	126	1010110	V	118	0x76	166	1110110	v
23	0x17	027	0010111	ETB	55	0x37	067	0110111	7	87	0x57	127	1010111	W	119	0x77	167	1110111	w
24	0x18	030	0011000	CAN	56	0x38	070	0111000	8	88	0x58	130	1011000	X	120	0x78	170	1111000	x
25	0x19	031	0011001	EM	57	0x39	071	0111001	9	89	0x59	131	1011001	Y	121	0x79	171	1111001	y
26	0x1A	032	0011010	SUB	58	0x3A	072	0111010	:	90	0x5A	132	1011010	Z	122	0x7A	172	1111010	z
27	0x1B	033	0011011	ESC	59	0x3B	073	0111011	;	91	0x5B	133	1011011	[	123	0x7B	173	1111011	{
28	0x1C	034	0011100	FS	60	0x3C	074	0111100	<	92	0x5C	134	1011100	\	124	0x7C	174	1111100	|
29	0x1D	035	0011101	GS	61	0x3D	075	0111101	=	93	0x5D	135	1011101	]	125	0x7D	175	1111101	}
30	0x1E	036	0011110	RS	62	0x3E	076	0111110	>	94	0x5E	136	1011110	^	126	0x7E	176	1111110	~
31	0x1F	037	0011111	US	63	0x3F	077	0111111	?	95	0x5F	137	1011111	_	127	0x7F	177	1111111	DEL

Source: http://www.catonmat.net/download/ascii-cheat-sheet.png

Text and strings today - Unicode

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

- UTF-16 – Fixed length - Two bytes
- UTF-32 – Fixed Length - Four Bytes
- UTF-8 – 1-4 bytes
  - Upwards compatible with ASCII
  - Automatic detection between ASCII and UTF-8
  - UTF-8 is recommended practice for encoding
    data to be exchanged between systems



https://en.wikipedia.org/wiki/UTF-8

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Using Unicode in Python 3

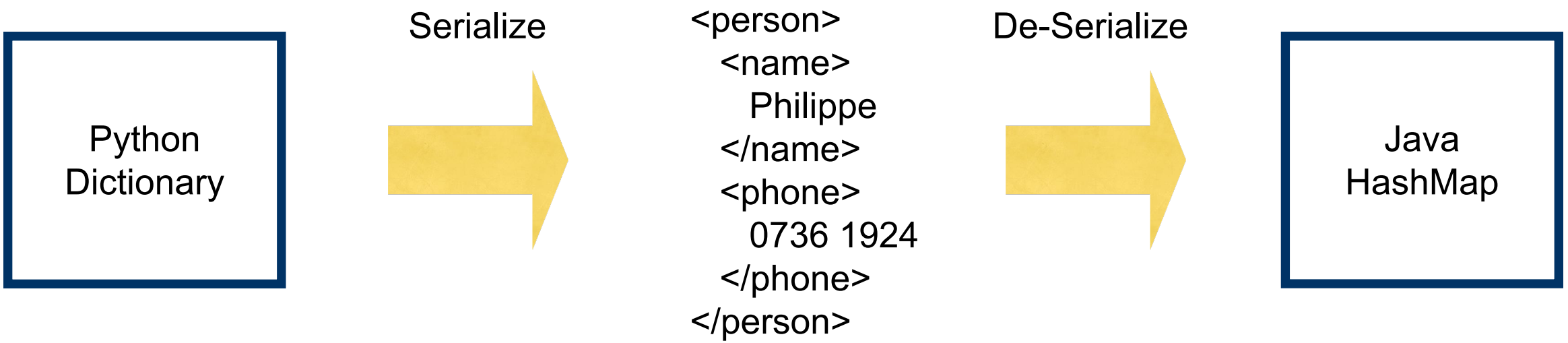
- In Python 3, all strings internally are UNICODE
- Working with string variables in Python programs and reading data from files usually "just works"
- When we talk to a network resource using sockets or talk to a database we have to encode and decode data (usually to UTF-8)
- When we talk to an external resource like a network socket we send bytes, so we need to encode Python 3 strings into a given character encoding
  - ‘this is a string to send’.encode()
- 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
  - ‘these are characters we receive’.decode()

Communicating data between computers

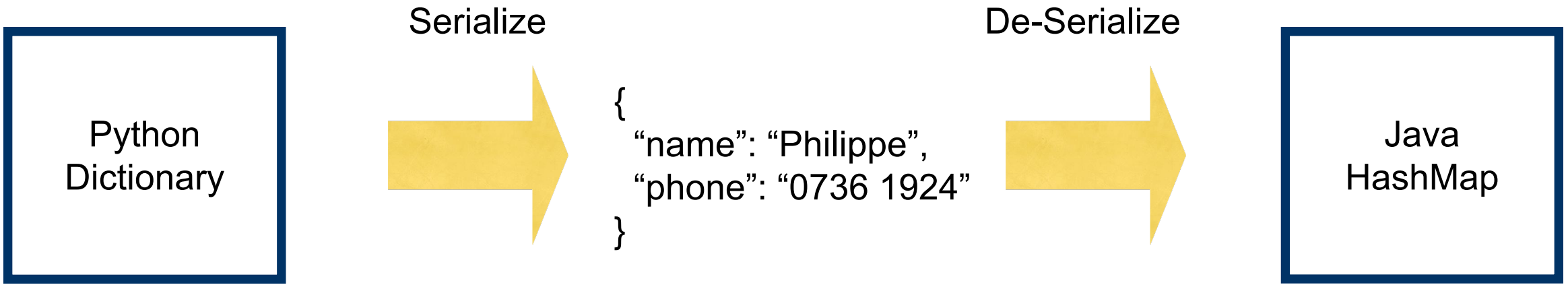
- Given the usefulness of HTTP(S) to transfer HTML document, there was a natural move toward exchanging data between programs using these protocols

  → need agreed way to represent data going between applications and across networks
- There are two commonly used formats: XML and JSON

Sending data across the web – XML



Sending data across the web – JSON



In more detail: eXtensible Markup Language

- Primary purpose: help information systems share structured data
- Started as simplified subset of the Standard Generalized Markup Language (SGML)
- Designed to be relatively human-legible

Start Tag

End Tag

Text Content

Attribute

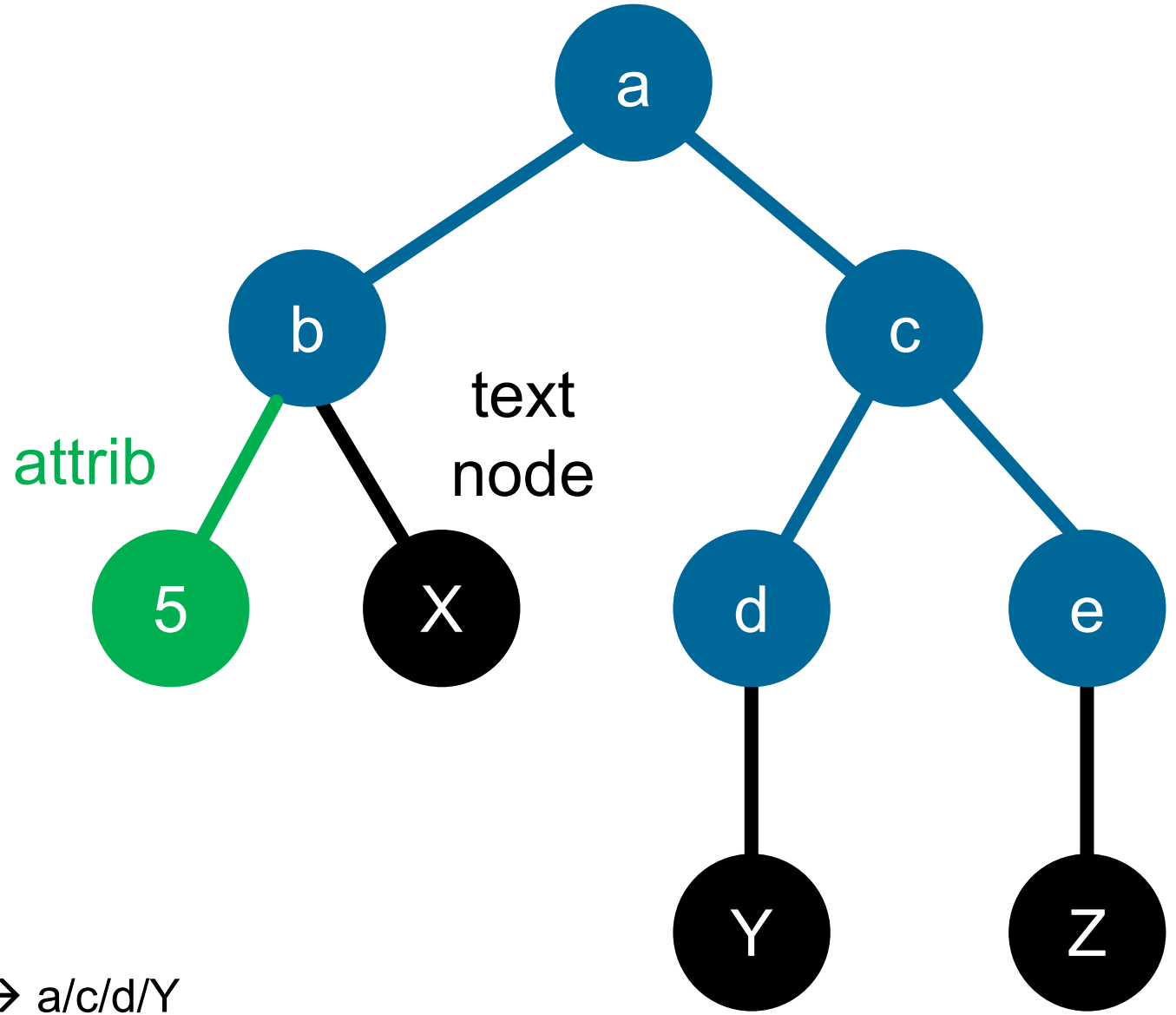
Self Closing Tag

```
<person>
  <name>Philippe</name>
  <phone type="intl">
     +44 736 1924
   </phone>
   <email hide="yes" />
</person>
```

BAYES BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

XML as a tree or a path

```
<a>
  <b w="5">X</b>
  <c>
    <d>Y</d>
    <e>Z</e>
  </c>
</a>
```



How to get to “Y”? → a/c/d/Y

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON

Validating an XML document – XML Schema

- XML Schema describes the the legal format of an XML document
  - Expressed in terms of constraints on the structure and content of documents
  - Often used to specify a “contract” between systems - “My system will only accept XML that conforms to this particular Schema.”
- If a particular piece of XML meets the specification of the Schema - it is said to “validate”
- Note: there are many XML Schema languages, but most common is the standard from W3C

XML in Python

In more detail: JavaScript Object Notation (JSON)

```
{
    “name” : “Philippe”,
    “phone” : {
        “type” : “intl”,
        “number” : “+44 736 1924”
    },
    “email” : {
        “hide” : “yes”
    }
}
```

JSON in Python

BAYES
BUSINESS SCHOOL
CITY, UNIVERSITY OF LONDON


See you in class!