Data on the Web

- The web started with static content and became gradually dynamic over the years.
- Now, almost every page is a Web Application or has some attributes of a Web Application.
- Reminder : A Web Application is HTML + CSS + JS + resources + server-side support
- Web Applications can process and display data
  - In e-commerce applications : catalog items and prices, stock information, . . .
  - In Social Networks/Blog applications : messages, photos, . . .
  - In Data Science applications : numerical data, graphs . . .
- Web data can be of different types :
  - Text content : real text (e.g. messages, comments), numbers (e.g. graph data, prices, . . .),
  - Non-textual content : images, videos, sounds
- Each type of data may have different server-side and client-side processing
  - How is the data stored server-side ?
  - What server-side processing is applied ?

Representing World Wide Web Resources

Source :
https ://en.wikipedia.org/wiki/Languages_used_on_the_Internet
(Feb 2020)

W3Techs estimated percentages of the top 10 million websites on the World Wide Web using various content languages

Rank	Language	Percentage
1	English	58.5%
2	Russian	8.1%
3	Spanish	4.4%
4	German	3.4%
5	French	3.0%
6	Persian	2.6%
7	Turkish	2.6%
8	Japanese	2.6%
9	Portuguese	2.3%

Internet Users

Source :
https ://en.wikipedia.org/wiki/Languages_used_on_the_Internet
(Feb 2020)

Rank	Language	Internetusers	Percentage
1	English	1,105M	25.2%
2	Chinese	863M	19.3%
3	Spanish	344M	7.9%
4	Arabic	226M	5.2%
5	Portuguese	171M	3.9%
6	Indonesian/ Malaysian	170M	3.9%
7	French	145M	3.3%
8	Japanese	119M	2.7%
9	Russian	109M	2.5%
10	German	92M	2.1%
1-10	Top 10 languages	3,346M	76.3%

The internationalization (i18n) problem

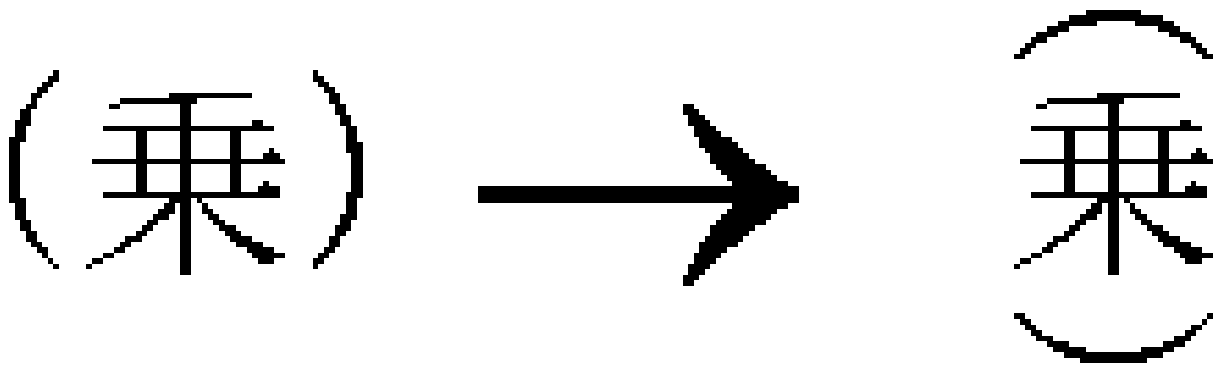
- Web resources are mostly text-based resources
- What is text ?
  - A sequence of character : what is a character ?
    - in English, in French, in Chinese, in Arabic . . .
    - what about symbols (e.g €), punctuation (., spanish reverse question mark) . . .
    - Difference character/character code (used for storage/transfer)
    - Difference character/graphical representation (used for display)
- Need for a text representation
  - Working for all languages
  - Including alphabets, ideograms, writing modes, . . .
  - Efficient for storage and network transfer
  - Efficient for display, editing, text selection
- Fundamentals
  - Unicode : Character Set
  - UTF-8 : Encoding

I18N Handling

- Correct processing of accents and other special characters

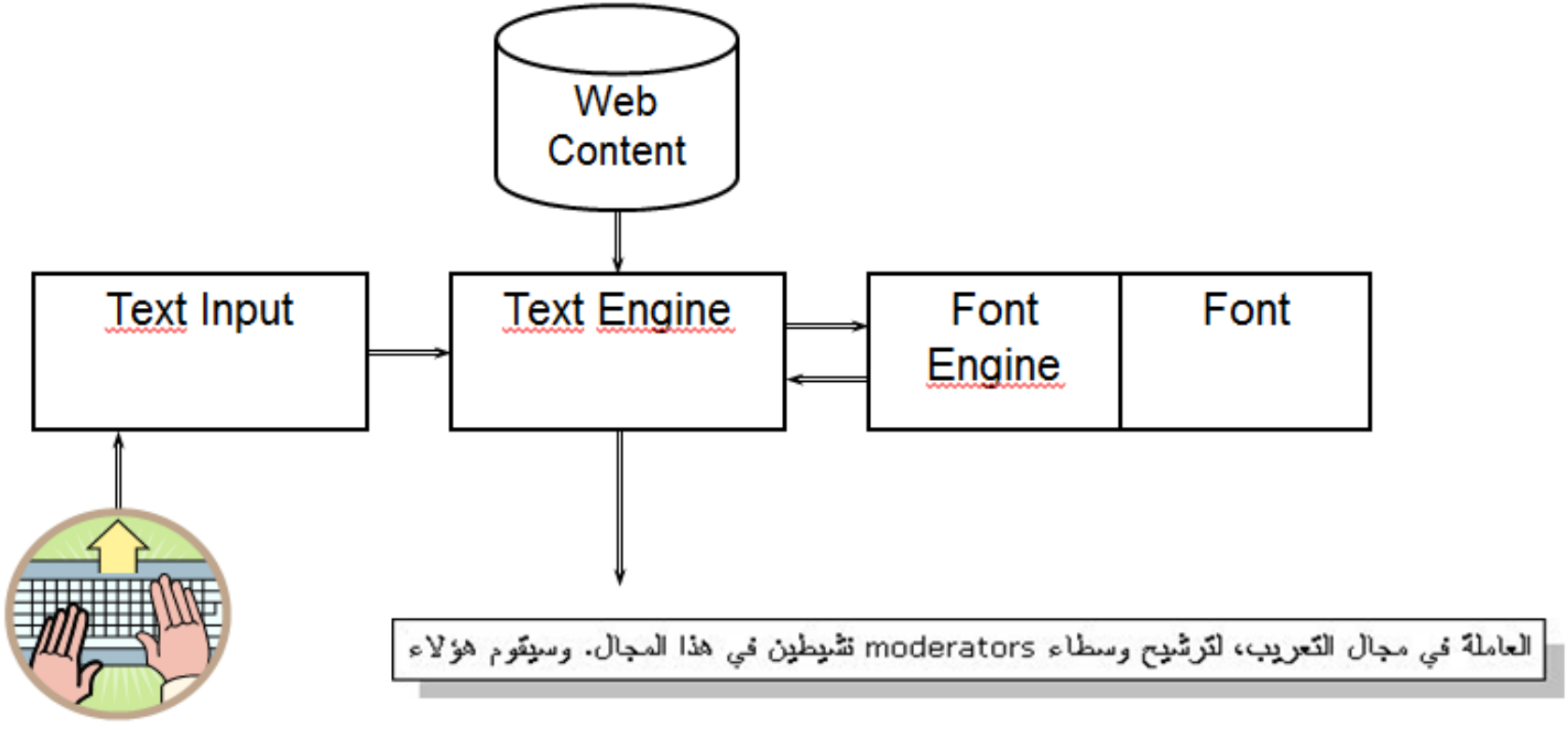
Ï Óýëëïãïò ÅëëÞíùí Ößëùí ôïõ TeX, ãíùóôùò é
LUG (*Local Users Group*). Ï ÅÖÔ, ðïõ éäñ
âíùôåñéêü. Åðßóçò ðñùôåéóáé íá óõíáñÙìåé
åñãáëåßùí (ð.÷. ôïõ ÙìÝãá) ðïõ èá äéåõêïëýíé
äéáÔñçóç êáé åöáñìïãÞ ôùí áíéþí ôçò ðáñáãù

- Using writing modes
  - Left-to-right/right-to-left/Vertical text



  - Text selection
- Handling language specificities
  - Arabic substitutions

I18N Processing



Character Set

- A set of ordered characters (aka Repertoire)
  - from one or more languages
  - closed (ASCII) or open (Unicode)
- Universal Character Set
  - Each character is only present once in the set
  - Characters are defined independently of their graphical representation or position in a text
- Each character is identified by its position (code position, code point)
- Characters from a set are encoded to store/transmit text : codec character set, character encoding

ASCII

- American Standard Code for Information Interchange
  - Invented in 1965 in the USA, standardised in 1983 as ISO 646
  - Derived with many variants
  - Widely used
- Set of 128 characters
  - 33 command characters (ex CR)
  - 95 printable character
  - 83 characters common to all ASCII variants
    - small, capital roman letters
    - digits
    - punctuation : (! " % & ‘ * + , - . / : ; < = > ? _) and space
  - 2 symbols : # or £ et $ or ¤
  - 10 variable characters (per country)
- Associated encoding on 7-bits

ASCII

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	a
002	☻	STX	034	"	066	B	098	b
003	♥	ETX	035	#	067	C	099	c
004	♦	EOT	036	$	068	D	100	d
005	♣	ENQ	037	%	069	E	101	e
006	♠	ACK	038	&	070	F	102	f
007	(beep)	BEL	039	'	071	G	103	g
008	◘	BS	040	(	072	H	104	h
009	(tab)	HT	041	)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	,	076	L	108	l
013	(carriage return)	CR	045	-	077	M	109	m
014	♫	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	o
016	►	DLE	048	0	080	P	112	p
017	◄	DC1	049	1	081	Q	113	q
018	↕	DC2	050	2	082	R	114	r
019	!!	DC3	051	3	083	S	115	s
020	¶	DC4	052	4	084	T	116	t
021	§	NAK	053	5	085	U	117	u
022	▬	SYN	054	6	086	V	118	v
023	↨	ETB	055	7	087	W	119	w
024	↑	CAN	056	8	088	X	120	x
025	↓	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[	123	{
028	(cursor right)	FS	060	<	092	\	124	|
029	(cursor left)	GS	061	=	093	]	125	}
030	(cursor up)	RS	062	>	094	^	126	~
031	(cursor down)	US	063	?	095	_	127	⌂

ASCII Variants

ISO-8859

- 8-bit extension to ASCII
- Same 128 first characters as ASCII
- 32 additional characters
- 96 language-specific characters
- ISO/IEC 8859-n, n=1…16 (aka Latin-1, Latin-2 …)

	008	009	00A	00B	00C	00D	00E	00F
0	XXX	DCS	NBSP	°	À	Đ	à	ð
1	XXX	PU1	¡	±	Á	Ñ	á	ñ
2	BPH	PU2	¢	²	Â	Ò	â	ò
3	NBH	STS	£	³	Ã	Ó	ã	ó
4	IND	CCH	¤	´	Ä	Ô	ä	ô
5	NEL	MW	¥	µ	Å	Õ	å	õ
6	SSA	SPA	¦	¶	Æ	Ö	æ	ö
7	ESA	EPA	§	·	Ç	×	ç	÷
8	HTS	SOS	¨	¸	È	Ø	è	ø
9	HTJ	XXX	©	¹	É	Ù	é	ù
A	VTS	SCI	ª	º	Ê	Ú	ê	ú
B	PLD	CSI	«	»	Ë	Û	ë	û

The Unicode Standard

- Universal Character Set
  - More than 1 million of representable characters
- Latest version
  - Unicode 8.0 - 06/2015
  - Over 120 000 characters defined
- Grouped in 17 planes de 2ˆ16 characters
  - Base Multilingual Plane (BMP)
  - Supplementary Multilingual Plane (SMP)
  - ...

Basic Multilingual Plane

A Unicode code point

- Each character is assigned
  - A unique code point (code position) :
    - U+xxxx (BMP) Ex : U+0044
    - Ex : U+yyxxxx (other planes)
  - A name : ex Capital latin letter D
  - A direction : « left – right » or « right – left »
  - A possible decomposition : é=e + ‘
  - Some language information
- The graphical shape is not associated
  - see Font information
- The byte representation on the wire is not defined in Unicode
  - see Character Encoding (fixed length, variable length)

Fixed-length Character Encoding

- Mostly defined by ISO
- ASCII
  - Not capable of encoding the Unicode Character Set
- UCS-2 (deprecated)
  - 16 bits - PMB
  - Not ASCII-compatible
- UCS-4 (deprecated)
  - 31 bits (+ leading 0 bit)
  - Designed for 32-bits machines
  - Restricted to [0x0..0x10FFFF] for UTF-16 compatibility
  - Not ASCII-compatible

Variable Length Character Encodings

- Mostly defined by IETF (RFC 2279, 1998)
- UTF-8 : Universal Transformation Format
  - Most popular format
  - 1-Byte alignment (no multi-byte problem)
  - ASCII-compatible (0..127)
    - An ASCII file transcoded in UTF-8 is identical to the original file
    - Bytes with the most-significant bit set to 1 are ignored by ASCII processors
  - Efficient conversion into UTF-16 & UTF-32
  - Used on the web
- UTF-16
  - Alignment on 2-bytes
  - BMP=2 bytes
  - Other planes=2 (indirection) + 2
  - Use of Byte Order Mark (BOM) to detect Endianness
  - Used on Windows and in Java
- UTF-32=UCS-4

Universal Transformation Format

Code Position Unicode

UTF-16

UTF-8 1st byte

UTF-8 2nd byte

UTF-8 3rd byte

UTF-8 4th byte

0000 0000 0xxx xxxx

0000 0000 0xxx xxxx

0xxx xxxx

0000 0yyy yyxx xxxx

0000 0yyy yyxx xxxx

Unicode & encodings : example and counter-examples

Character	Unicode Code	UTF-8	UTF-8 in ASCII	UTF
A	U+0041	41	A	004
space	U+0020	20		002
é	U+00C9	C3 A9	Ã©	00E
greek delta	U+03B4	CE B4	Î´	03B
Å	U+00C5	C3 85	Ã…	00C
Å	U+212B	E2 84 AB	â„«	212
A + °	U+0041 + U+030A	41 CC 8A	AÌŠ	004

Other encodings

- ISO-8859-1 : Western Europe
- ISO-8859-6 : Arabic
- ISO-8859-11 : Thai
- Windows-1252 : Western languages
- Shift-JIS : Japanese
- GB-2312 : Chinese Guobiao
- Big-5 : Taïwan
- ISO-2022-KR : Korean
- . . .

Declaring character encoding

- In HTTP Headers (default is ISO-8859-1)

```
Content-Type: text/html; charset=utf-8
```

- XML Declaration

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

- In HTML Documents

```
<meta charset='utf-8'>
<meta http-equiv="Content-Type" content="text/html;charset=UTF
```

Escape codes in Web Content

Character(s)
é
Å
greek delta
±
space
Text
HTML Escaping (a.k.a. entity names or entity numbers)
&acute ; / &#x00C9 ;
&Aring ; / &#x212B ;
&delta ; / &#x03B4 ;

Structured Text Data

- Text data that is structured, with a specific syntax to relate pieces of text :
  - CSV (Comma Separated Values, exported from Spread Sheets (Excel, . . .))
  - XML (syntax inspired by HTML)
  - JSON (syntax inspired by JavaScript), JSONP
- Data is often stored in databases
  - Possibly exported in one of these formats
  - Or directly integrated into the HTML content (e.g. via HTML Templates))

CSV

- Example

```
city,state,population,landarea
seattle,WA,652405,83.9
new york,NY,8405837,302.6
boston,MA,645966,48.3
kansas city,MO,467007,315.0
```

- Be careful of :
  - absence of comments,
  - difficult use of ", line break, spaces or commas in the content. . .
- How to process it in a Web Browser ?
  - Example with D3.js `d3.csv("/data/cities.csv", function(data) { console.log(data[0]); });` → `{city: "seattle", state: "WA", population: 652405, landarea: 83.9}`
  - Other examples : jQuery, . . .
- Limits :
  - When the number of columns is variable in each line

XML

- Example

```
<data>
<sensor time="0" type="3D" x="0" y="12" z="33"/>
<sensor time="0" type="temperature" value="10"/>
<sensor time="10" type="3D" x="0" y="22" z="33"/>
<sensor time="20" type="2D" x="0" y="12"/>
</data>
```

- Highlights
  - Can be flat, similar to CSV, with a markup syntax
  - Variability in the number and type of data per “line”
  - Possible validation of the data (3D requires z)
  - Can represent more complex data structure
  - Verbosity

XML continued

- How to process it in a Web Browser ?

```
var xhttp = new XMLHttpRequest();
xhttp.onload = function() {
    if (this.status == 200) {
        console.log(this.responseXML);
   }
};
xhttp.open("GET", "http://server.com/data", true);
xhttp.send();
```

JSON

- Example

```
[
{ "city": "seattle","state": "WA","population": 652405,"landare
{ "city": "new york","state": "NY","population": 8405837,"landa
{ "city": "boston","state": "MA","population": 645966,"landarea
{ "city": "kansas city","state": "MO","population": 467007,"lan
]
```

- Highlights :
  - Similar to XML, with a JS-like syntax but
    - Absence of comments,
    - Need to use " for property names
    - Not tolerant to errors (trailing comma)

JSON continued

- How to process it in a Web Browser ?
  - Example with D3.js `d3.json("/data/cities.json", function(data) { console.log(data[0]); });`
    `→ {city: "seattle", state: "WA", population: 652405, landarea: 83.9}`
  - Other examples : basic XHR, jQuery, ...
- Limit : Cross-origin restrictions

JSONP

- JSON is restricted to Single-Origin requests unless using CORS
- JavaScript is not restricted
- JSON cannot be used as is in a <script> element (no variable name)
- JSONP concepts :
  - Wrap JSON into JS code (variable, function) to make it script-compatible `process({ "city": "seattle","state": "WA","population": 652405,"landarea": 83.9 });`
  - The wrapped JSON can be loaded via a <script> element
  - The actual wrapper can be generated specifically based on the URL `<script type="application/javascript" src="http://server.example.com/City/Seattle?callback=proces </script>`

Databases & the Web

- Database types :
  - Relational databases / Tables / SQL : MySQL, . . .
  - Key-value / Document-oriented : CouchDB, MongoDB, . . .
- APIs :
  - REST
  - SOAP

REST and Web Services

- A Web Service is
  - Software
  - Exposes functions with a communication protocol on the web
  - With a standard way to use it, independent from languages and systems
- This makes possible
  - To make the service accessible on the web
  - To distribute the services
  - To concatenate services into more complex ones
  - To use a well established network infrastructure

Example

Gitlab has a REST interface that I use to gather information about the amount of work that a PACT group is doing :

- number of commits in a project : /projects/ :id/repository/commits
- list members of a project : /projects/ :id/members
- etc

The response is a JSON. The API is quite detailed.

I only have access to these because I am an admin for these projects, and I authenticate with Gitlab. :id is a 4 digits number identifying the repository.

REST : Representational State Transfer

- Neither a protocol, nor a format
- More a style of distributed service
  - You can use the model/style completely or just reuse parts
  - Initial proposal by Roy Fielding
- Basic principles
  - You just need to know the URL of a service to access it
  - HTTP provides everything required :
    GET, PUT, POST, DELETE are used as action commands on the server
  - Stateless : the URL contains all the information required for the server to provide an answer, there is no need for the server to keep any client state
    (there may still be a server state, such as a DB)

REST URL scheme

Typical form-related url :

```
http://server/path?param=value&param2=value2&...
```

Typical URL scheme for a REST service :

```
http://server/path/value/value2
```

where value, value2 are parameter values of the request.

Benefits of using REST

- Simple to implement, at least for developers used to implementing dynamic web services
- Stateless means
  - Server load is smaller, can deal with more clients
  - Easy to debug
  - Easy to balance the load onto a server farm
- Excellent integration into the HTTP universe
- Standard Web Cache works well with the use of URLs

Web APIs

- Web Services accessible on the Web (including REST) are often called Web APIs
- ProgrammableWeb
  - Example of an API directory
  - https ://www.programmableweb.com/category/all/apis
  - Hundreds of referenced APIs, covering mapping, social networks, translation. . .