Standards for language encoding

Tomaž Erjavec
Dept. of Knowledge Technologies
Jožef Stefan Institute

ESSLLI 2011

A few words about me

- Tomaž Erjavec
  Department of Knowledge Technologies
  Jožef Stefan Institute
  Ljubljana
- http://nl.ijs.si/et/, tomaz.erjavec@ijs.si
- Areas of work: compilation and annotation of corpora and other language resources, encoding standards, digital libraries (text-critical editions)
- Web page for this course:
  http://nl.ijs.si/et/teach/esslli11/
  and, of course, Moodle

Overview of the course

1. Introduction, character sets
2. Structuring data: XML
3. Encoding for the humantities: TEI
4. Standards for LRs: ISO
5. Semantic Web: W3C standards

I. Introduction

What are standards?

- dictionary:
  1. an obligatory uniform regulation for measurement, quantity or quality
  2. that which specifies how something can or must be
- consensually accepted regulations, which are public and contain explicit definitions
- the main purpose is to harmonise industrial practice in various fields in order to enable interchange

History of standardisation

- XVIII century: in France each region (village) has its own units of measurement; also, different objects (say a field or forest) are measured differently
- how to define a uniform system of measurements: search for a single unit from which it would be possible to derive all other measures
- meter: one ten-millionth of the length of the meridian through Paris, from the North Pole to the equator
- the importance of standardisation grows with the industrial revolution: mechanical and electrical engineering, construction work…
- today, standards encompass even such “soft” fields as the organisation of business (ISO 9000)
- big business: companies that check compliance with standards

Standards and best practices

- National standard bodies: DIN, ANSI, SIST
- International standard bodies:
  - IEC: International Electrotechnical Commission
  - ISO: International organisation for standardisation
  - IETF: Internet Engineering Task Force
  - W3C:World Wide Web Consortium
  - Unicode consortium
- Initiatives:
  - MUFI: Medieval Unicode Font Initiative
  - TEI: the Text Encoding Initiative
- Best practices:
  - Penn Treebank PoS tags
  - TIGER annotation scheme

Language resources

- Corpora
  - monolingual and multilingual
  - general and domain specific
  - raw text or annotated
  - text or speech
- Lexica
  - monolingual and multilingual
  - words and lemmas
  - entities (names)
  - phrases (terms)

Annotations

- Morphological level: lemmas/stems, coarse (PoS) or fine (MSD) grained tags
- Syntax: syntactic trees or dependencies
- Semantics: word senses, semantic roles
- Named entities: names, dates, numeric expressions
- Terms, time & space expressions and relations
- Anaphora: anaphoric links
- Parallel corpora: sentence and word/phrase alignments
- Meta-data: information about the resource

Utility of LRs

A basis for:

- HLT development
  - Training: datasets for inducing language models
  - Testing: datasets for evaluating performance
- Empirically driven (applied) linguistics:
  - Corpus linguistics
  - Lexicography, Terminography
  - Language teaching

Why standards for encoding of digital data?

Traditionally, each developer made LRs to work with their particular software and for their particular needs

Problems:

- *longevity:* advances in technology make programs soon obsolete and data bound to these programs becomes unreadable
- *interchange*: difficult to use data on other platforms or pass it between programs
- *exploitation:* difficult to re-use the data for other purposes
- *intelligibility:* no public and stable specifications of the format
- *validation:* we don't know whether certain data is written according to the specification or not

Standardisation of LRs

- LRs are expensive to produce – so, a good idea if they are reusable and long-lasting
- LRs are becoming larger and with more complex annotations – no good for everyone to reinvent the wheel
- Familiarity with standards in this area helps to produce good resources and to be able to use the resources already produced:
  - freely available resources: Google, (MetaShare)
  - LDC, ELRA
  - corpora@uib.no

Levels of standardisation

- Characters: the basic building blocks
- XML: structuring the data and assigning annotations
- TEI: a large vocabulary of XML elements: “encoding text for scholarly purposes”
- ISO standards: some basic things like dates and languages; and recent attempts to standardise many different types of LRs
- Semantic Web: Meta-data and ontologies

II. Character sets

- Characters are the “atoms” of textual resources
- It still often happens that characters are garbled in processing, resulting in useless text
- Currently, Unicode is gaining ground but is still not the only character set in use
- Unicode is relatively complex

Character encoding

- Digital computers store data as (binary) numbers
- There is no a priori connection between these numbers and characters (of an alphabet)
- If there are no conventions for this mapping or if there are too many → chaos
- Standards and quasi standards:
  ASCII, ISO 8859, (Windows, Mac), Unicode

Basic concepts I.

character

- abstract concept (An „A“ is something like a Platonic entity: it is the idea of an „A“ and not the „A“ itself)
- a character does not by itself have a mapping to a number or a specific graphical representation
- usually it is descriptivelly defined, e.g. „Greek letter lower-case alpha “, and the graphical representation is given only as a suggestion, „α“

Basic concepts II.

- *character set*
  - a set of characters
  - each character has an associated character code
- *character code*
  - 1-1 relation between the character from a character set and a number, e.g. A = 26, B = 27, ...
  - Note:
    character codes are often written in hexadecimal:
    0 → 0, 1 → 1, 2 → 2, ... 9 → 9,
    10 → A, 11 → B, ..., 15 → F,
    16 → 10, 17 → 11, ...,
    254 → FE, 255 → FF, 266 → 100

Example: the ASCII character set

Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	_
48	0	64	@	80	P	96	`	112	p		

e.g.

in the ASCII character set

the character lower case Latin a

has the character code 97

Basic concepts III.

- *glyph*
  - a graphical representation of a character
  - one character can have more than one glyph
    e.g. the character “upper-case Latin A” ↔ glyphs 
  - sometimes one glyph can be associated with more than one character, e.g. the glyph P ↔ characters “upper-case Latin P”, “upper-case cyrillic R”, “upper-case Greek Rho”
- *font*
  - a set of glyphs (for some character set):
    
  - sometimes a font does not cover the complete character set!

Some character sets

- ASCII - oldest, contains only the letters of the English alphabet + punctuation, numbers
- Family of characters sets ISO 8879
- The Windows family of character sets („code pages“)
- Unicode

ASCII

- American Standard Code for Information Interchange (1950')
- 7-bit encoding: character codes 0-127
- 0-31 – control characters + formatting characters:
  Esc, Line Feed, tab, space,...
- 32-126 – punctuation and special characters, numbers, upper- and lower-case letters:

  ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ;
  < = > ? @ A B C D E F G H I J K L M N O P
  Q R S T U V W X Y Z [\ ] ^ _ ` a b c d e f g h
  i j k l m n o p q r s t u v w x y z { | } ~

The ISO 8859 family

- need for extra characters for national (European) alphabets:
  - 80‘s
  - 8 bits, so twice as many chars as in ASCII
  - first half = ASCII, second half = new characters
- International Standards Organisation publishes character sets for particular groups of European languages: **ISO 8859 (-1 .. -12)**
- ISO 8859-1 (ISO Latin 1) – Western European languages:

¡ ¢ £ ¤ ¥ ¦ § ¨ © ª « ¬ ® ¯ ° ± ² ³ ´ µ ¶ · ¸ ¹ º » ¼ ½
¾ ¿ À Á Â Ã Ä Å Æ Ç È É Ê Ë Ì Í Î Ï Ð Ñ Ò Ó Ô Õ
Ö × Ø Ù Ú Û Ü Ý Þ ß à á â ã ä å æ ç è é ê ë ì í î ï
ð ñ ò ó ô õ ö ÷ ø ù ú û ü ý þ ÿ

ISO 8859-2
C&E European languages

A–	NBSP 00A0 ***160***	Ą 0104 ***161***	˘ 02D8 ***162***	Ł 0141 ***163***	¤ 00A4 ***164***	Ľ 013D ***165***	Ś 015A ***166***	§ 00A7 ***167***	¨ 00A8 ***168***	Š 0160 ***169***	Ş 015E ***170***	Ť 0164 ***171***	Ź 0179 ***172***	SHY 00AD ***173***	Ž 017D ***174***	Ż 017B ***175***
B–	° 00B0 ***176***	ą 0105 ***177***	˛ 02DB ***178***	ł 0142 ***179***	´ 00B4 ***180***	ľ 013E ***181***	ś 015B ***182***	ˇ 02C7 ***183***	¸ 00B8 ***184***	š 0161 ***185***	ş 015F ***186***	ť 0165 ***187***	ź 017A ***188***	˝ 02DD ***189***	ž 017E ***190***	ż 017C ***191***
C–	Ŕ 0154 ***192***	Á 00C1 ***193***	Â 00C2 ***194***	Ă 0102 ***195***	Ä 00C4 ***196***	Ĺ 0139 ***197***	Ć 0106 ***198***	Ç 00C7 ***199***	Č 010C ***200***	É 00C9 ***201***	Ę 0118 ***202***	Ë 00CB ***203***	Ě 011A ***204***	Í 00CD ***205***	Î 00CE ***206***	Ď 010E ***207***
D–	Đ 0110 ***208***	Ń 0143 ***209***	Ň 0147 ***210***	Ó 00D3 ***211***	Ô 00D4 ***212***	Ő 0150 ***213***	Ö 00D6 ***214***	× 00D7 ***215***	Ř 0158 ***216***	Ů 016E ***217***	Ú 00DA ***218***	Ű 0170 ***219***	Ü 00DC ***220***	Ý 00DD ***221***	Ţ 0162 ***222***	ß 00DF ***223***
E–	ŕ 0155 ***224***	á 00E1 ***225***	â 00E2 ***226***	ă 0103 ***227***	ä 00E4 ***228***	ĺ 013A ***229***	ć 0107 ***230***	ç 00E7 ***231***	č 010D ***232***	é 00E9 ***233***	ę 0119 ***234***	ë 00EB ***235***	ě 011B ***236***	í 00ED ***237***	î 00EE ***238***	ď 010F ***239***
F–	đ 0111 ***240***	ń 0144 ***241***	ň 0148 ***242***	ó 00F3 ***243***	ô 00F4 ***244***	ő 0151 ***245***	ö 00F6 ***246***	÷ 00F7 ***247***	ř 0159 ***248***	ů 016F ***249***	ú 00FA ***250***	ű 0171 ***251***	ü 00FC ***252***	ý 00FD ***253***	ţ 0163 ***254***	˙ 02D9 ***255***
	–0	**–1**	**–2**	**–3**	**–4**	**–5**	**–6**	**–7**	**–8**	**–9**	**–A**	**–B**	**–C**	**–D**	**–E**	**–F**

Confusion

- Microsoft also developed their own code pages:
  - Windows CP1252 v.s. ISO-8859-1
  - Windows CP1250 v.s. ISO-8859-2
- Also other „standards“: IBM, Apple, …
- Problems with Web pages

8-bit character sets (ISO 8859, Windows)

- advantages to ASCII:
  - we can directly write the characters for national alphabets (*slovenščina*)
- disadvantages:
  - we cannot write multilingual texts in the same character set
  - confusion due to competing character sets
  - no coverage for East Asian languages or more complex characters: punctuation, math operators, diacritics, historical characters, Klingon…
  - the file gives no indication which character set it uses:

    © Global publishing ~ Ž Global publishing

The final solution

- Need a character set that would be universal, i.e. would contain all the world's characters
- Must be well documented and open
- Has to be consensually developed and maintained
- Still needs some room for “private” characters

Unicode

1991 – Unicode Consortium: http://www.unicode.org/

Unicode Standard / ISO 10646 „Universal Character Set“

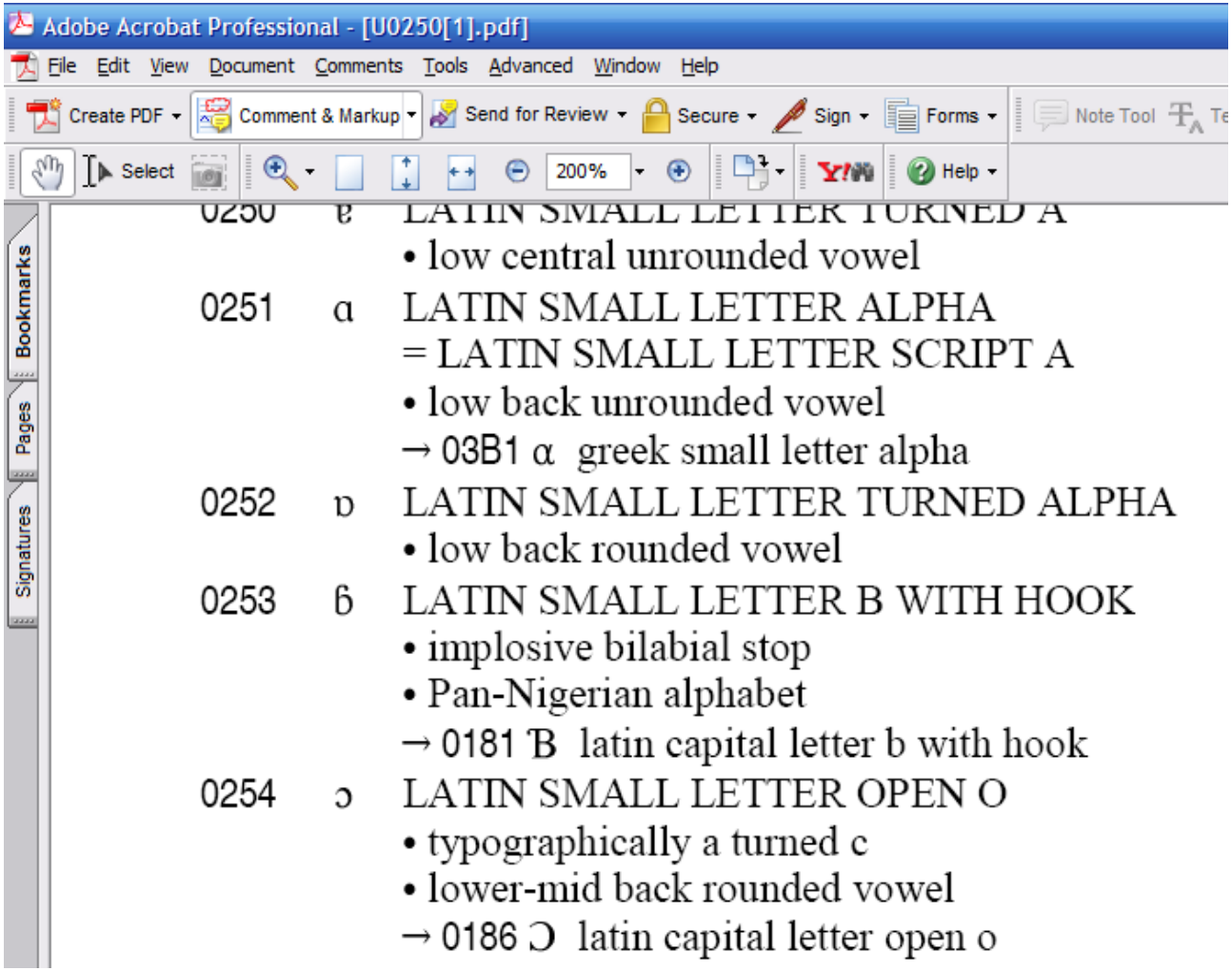
- The most recent major revision is **Unicode 6.0**. (2011): 109,000 characters, 93 scripts
- code charts for visual reference + reference data files
- encoding methodology, character properties, rules for normalization & decomposition, collation, rendering, bidirectional display: complex!
- As yet unrealised ambition:
  completely replace other character sets

Visual reference: Character code charts

Latin Extended-A

Position	Decimal	Name	Appearance
0x0100	256	LATIN CAPITAL LETTER A WITH MACRON	Ā
0x0101	257	LATIN SMALL LETTER A WITH MACRON	ā
0x0102	258	LATIN CAPITAL LETTER A WITH BREVE	Ă
0x0103	259	LATIN SMALL LETTER A WITH BREVE	ă
0x0104	260	LATIN CAPITAL LETTER A WITH OGONEK	Ą
0x0105	261	LATIN SMALL LETTER A WITH OGONEK	ą
0x0106	262	LATIN CAPITAL LETTER C WITH ACUTE	Ć
0x0107	263	LATIN SMALL LETTER C WITH ACUTE	ć
0x0108	264	LATIN CAPITAL LETTER C WITH CIRCUMFLEX	Ĉ
0x0109	265	LATIN SMALL LETTER C WITH CIRCUMFLEX	ĉ
0x010A	266	LATIN CAPITAL LETTER C WITH DOT ABOVE	Ċ
0x010B	267	LATIN SMALL LETTER C WITH DOT ABOVE	ċ
0x010C	268	LATIN CAPITAL LETTER C WITH CARON	Č
0x010D	269	LATIN SMALL LETTER C WITH CARON	č

Unicode definitions for IPA

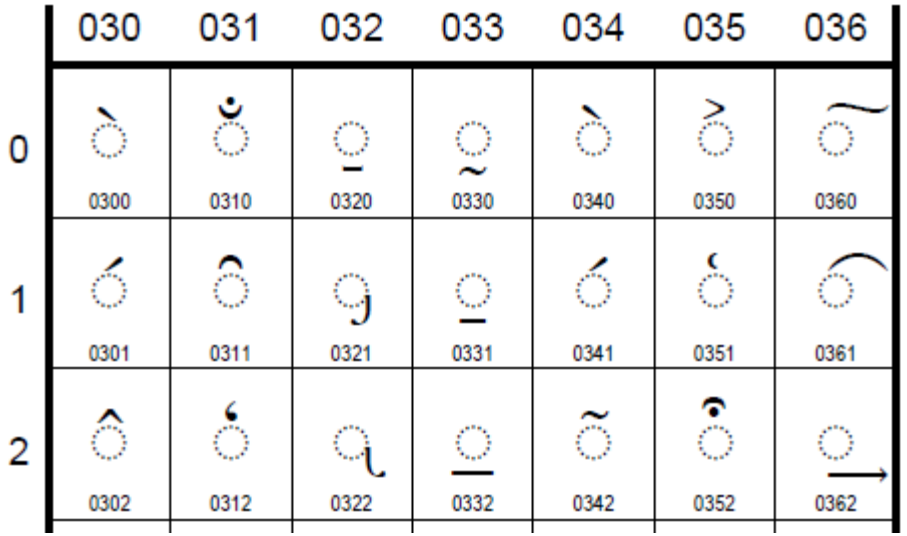


Reference data

- Unicode Character Database
- e.g. http://www.unicode.org/Public/UNIDATA/NamesList.txt
- CSV files, XML, PDFs
- Various software uses this information:
  Java classes, Perl modules, e.g.
  - m/[[:upper:][:punct:]]/;
    matches any upper case letter or punctuation symbol
  - charnames::viacode(4532)
    returns: LATIN CAPITAL LETTER U WITH DOUBLE GRAVE

Unicode and diacritics

- many letters are available with diacritics as individual characters: áâãäåǎāǻȧ
- but diacritics also exist as combining characters (combining diacritical marks)
- eg.: a + ˆ + ¨ = a̽̈
- although problems with display of complex combinations, e.g. a + ˆ + ˚ = a̽̊
- solved by specialised fonts



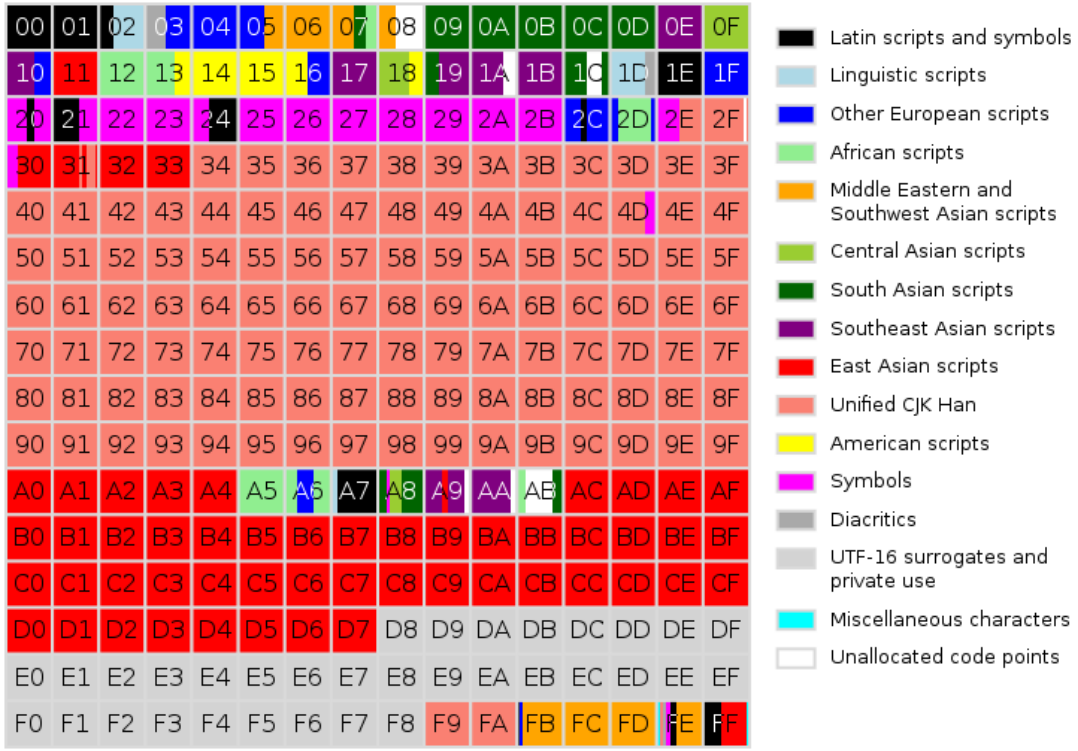
Private Use Area

- Not all characters are (or could be) included in Unicode
- Unicode allows for addition of new characters, but only after an extended process
- For extra characters, a Private Use Area (PUA) is designated
- Fonts are free to use PUA, but with the understanding that these characters are not portable

- Example use: Freising Manuscripts

Unicode planes & BMP

v·d·e	Unicode **planes** and code point (character) ranges				[hide]
Basic	**Supplementary**				
0000–FFFF	**10000–1FFFF**	**20000–2FFFF**	**30000–DFFFF**	**E0000–EFFFF**	**F0000–10FFFF**
Plane 0: Basic Multilingual Plane	**Plane 1:** Supplementary Multilingual Plane	**Plane 2:** Supplementary Ideographic Plane	**Planes 3–13:** Unassigned	**Plane 14:** Supplementary Special-purpose Plane	**Planes 15–16:** Private Use Area
BMP	**SMP**	**SIP**	—	**SSP**	**S PUA A/B**



Basic multilingual plane
0000 - FFFF
(65,535 characters)

E000–F8FF - PUA

Encoding Unicode

- ISO 8859, Windows charaters sets:
  8bit, therefore limited to 256 characters
- Trivial mapping: one char = one byte
- But Unicode codepoints can be huge
- Necessary to use several bytes to encode one character
- All Unicode codepoints (numbers) fit into 4 bytes
- But it is – in general – very wasteful to use 4 bytes for one char
- Is there any better way to do it?
  Yes, several..

Unicode Transformation Format

UTF defines how to map codepoints to bytes (bits), which are then stored or transmitted

- UTF-32
  - 1 character = always 4 bytes
- UTF-16
  - 1 character = 2 bytes in basic multilingual plane
- **UTF-8**
  - if char in ASCII, then in 1 byte (compatibility!)
  - otherwise 1-6 bytes for 1 char
  - cunning system, where not all byte sequences are valid (so, won‘t mix with 8 bit encodings)

Back to ASCII

ASCII is sometimes still the safest:

- problems with input and display of chars
- data transfer (e-mail)

Recoding to ASCII:

- e-mail: - MIME standard
- HTML and XML – character entities:
  š = Š = š

Defining the character set of a document

- **HTML:**

```
<HTML>
  <HEAD>
    <TITLE>Recept za ribano kašo</TITLE>
    <META http-equiv="Content-Type"
                content="text/html; charset=ISO-8859-2">
  </HEAD>
  <BODY>
    …
```

- **XML:**

```
<?xml version="1.0" encoding="utf-8"?>
<recept>
 <naslov>Recept za ribano kašo</naslov>
  …
```

- Some valid character sets:
  - utf-8, iso-8859-X, us-ascii

i18n, l10n

- Internationalisation and localisation:
  enabling programs to work with different languages (and cultures)
- E.g. language of program messages and help; keyboard layout; date and number format
- CLDR - Unicode Common Locale Data Repository
- For language resources:
  - collating sequence: *a,b,c,č,d*, not *a,b,c,d,…č*
- Unix: system variables regulate which locale is selected, e.g. LC_COLLATE = sl_SI.UTF-8

Case study: Cleaning Gigafida

- 1,000,000,000 word tokens
- Unicode + XML + TEI
- Character profile: 1200
- Forbidden chars: 500
- Excel
- Character normalisation

Fixing chars in Gigafida

- Hyphens:
  $s=~s/[\x{0336}\x{0096}\x{2010}]/-/g;
- Spaces:
  $s=~s/[\x{00A0}\x{2002}\x{2008}\x{2009}\x{202F}]/ /g;
- Digraphs:
  $s=~s/ffi/ffi/g; $s=~s/ffl/ffl/g; $s=~s/ff/ff/g; $s=~s/fl/fl/g;
- Non-spacing diacritics:
  $s=~s/ñ/ñ/g; $s=~s/č/č/g; $s=~s/š/š/g;
- Entities:
  $s=~s/&/&/g; $s=~s/ / /g; $s=~s/©/©/g;

An excercise

ʧmə́rəlj

- This is the phonetic spelling of the Slovene word „čmrlj“ (bumbelbee)

1. Write the characters in Word
2. Find which Unicode characters these are http://www.unicode.org/charts/