

# Character Sets, Internationalization (I18N), and Localization (L10N)

[Web Architecture and  
Information Management](#) [./]  
Spring 2009 — INFO 190-02  
(CCN 42509)

[Erik Wilde, UC Berkeley School of  
Information](#)  
2009-03-18



<http://creativecommons.org/licenses/by/3.0/>

This work is licensed under a [CC Attribution 3.0 Unported License](http://creativecommons.org/licenses/by/3.0/) [http://creativecommons.org/licenses/by/3.0/]

## Contents

• Abstract	2
• 1 Characters and Character Sets	
◦ Characters and Computers	4
◦ Characters	5
◦ Glyphs	6
◦ Character Set Identification	7
◦ History of Character Sets	8
◦ ASCII 1967	9
◦ Beyond ASCII	10
◦ ISO 8859	11
◦ ISO 8859-1 (Latin-1) & ISO 8859-15 (Latin-9)	12
• 2 Unicode	
◦ ISO 8859 Problems	14
◦ Unicode	15
◦ Unicode Character Count	16
• 3 Internationalization (I18N)Internationalization (I18N)	
◦ What is Language?	18
◦ Beyond Language	19
◦ Directionality and Screen Layout	20
◦ Definition	21
◦ I18N Tasks	22
• 4 Localization (L10N)	
◦ Definition	24
◦ L10N Tasks	25
• Conclusions	26

## Abstract (2)

---

Every character-based document is based on some model of which characters are available, and how they are encoded. *Unicode* is the most popular character set today and provides a variety of encoding schemes, each of them being a *Unicode Transformation Format (UTF)*. Many publishing environments need to support multiple languages. *Internationalization (I18N)* is the approach to design systems which can adapt to different locales. *Localization (L10N)* is the activity to identify, define, and encode locales, based on internationalized software.

# Characters and Character Sets

---

## Characters and Computers (4)

---

- *American Standard Code for Information Interchange (ASCII)*
  - for the first time a basic set of characters had a universally accepted encoding
  - many Internet protocols (such as [HTTP](#) [Web Foundations (URI and HTTP); Hypertext Transfer Protocol (HTTP) (1)]) encode their information in ASCII commands
- ASCII is a very limited repertoire of characters
  - basic ASCII contains 128 characters (7 bit) with a number of control chars
  - no variants of characters (german umlauts, french accents) are supported
  - various code pages extending ASCII to 8 bit exist and are hard to distinguish
- *Character* is not a trivial concept when regarded globally
  - european languages have writing systems based on a small number of "atoms"
  - other languages and writing systems have different ideas of "language atoms"

## Characters (5)

Character. (1) The smallest component of written language that has semantic value; refers to the abstract meaning and/or shape [...]

[The Unicode Standard, Version 4.0, Addison-Wesley, 2003](http://dret.net/biblio/reference/unicode4) [http://dret.net/biblio/reference/unicode4]

- Different languages or writing systems use different characters
  - english: A B C D E F G H I J K L ...
  - german: A Ä B C D E F G H I J K L ...
  - russian: А Б В Г Д Е Ё Ж З И Й К ...
  - greek: Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ ...
  - hiragana: いろはにおえどちりぬるを ...
  - kanji: 阪熊奈岡鹿梨卓埼茨栃媛 ...

## Glyphs (6)

[A Glyph is] a recognizable abstract graphic symbol which is independent of a specific design.

[ISO/IEC 9541:1991, Information Technology – Font Information Interchange](http://dret.net/biblio/reference/iso9541) [http://dret.net/biblio/reference/iso9541]

- *Visual rendering* introduces the notion of a glyph.
- There is *not* a one-to-one correspondence between characters and glyphs
  - A A : a single character can be represented by multiple glyphs
  - ffi : a single glyph may represent a sequence of characters
  - a character may be rendered with different glyphs ([old german script](http://www.tr62.de/script/letter4.html))
  - A : a single glyph may represent different characters (e.g. latin, greek, and cyrillic)

## Character Set Identification (7)

- HTTP, XML, and HTML support character set identification
  - [HTTP](#) [Web Foundations (URI and HTTP); Hypertext Transfer Protocol (HTTP) (1)] supports the `Content-Type` [header field](#) [../services-fall06/web1#(17)]  
`Content-Type: text/html; charset=utf-8`
  - XML encodes the character set [in the XML declaration](#) [../xml-fall08/basics#(8)]  
`<?xml version="1.0" encoding="utf-8"?>`
  - [HTML](#) [Hypertext Markup Language (HTML)] supports the meta element in the document's head  
`<meta http-equiv="Content-Type" content="text/html; charset=utf-8">`

## History of Character Sets (8)

- Text documents need ways to represent characters
  - computers handle bits, not characters
  - to handle characters, computers need a mapping from characters to bits
- For a long time, computers were doing their work in a very isolated way
  - "I think there is a world market for maybe five computers." (→ [T. J. Watson](#) [http://en.wikipedia.org/wiki/Thomas\_J.\_Watson#Famous\_misquote])
- With more computers being used, more data is exchanged between computers
- Standardization of character sets started in the 1960's
  - ASCII was the first generally accepted character set
  - EBCDIC was invented and marketed by IBM (more than 128 characters)
  - ISO 8859 was the first attempt to better support character sets beyond ASCII
  - *asian scripts* were always a problem because of the number of characters they need

## ASCII 1967 (9)

	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
000	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
020	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
040		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
060	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
120	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
140	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
160	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

## Beyond ASCII (10)

- ASCII is called ASCII for a reason
  - it works well for english-speaking countries
  - the majority of other languages cannot be represented
- Character sets and the 8 bit computer start to collide
  - ASCII is very convenient because characters and bytes correspond 1:1
  - every character set expanding ASCII will make this more complicated
  - complications can occur within and/or outside of the character set
- Introducing a character set beyond 8 bit is a fundamental change
  - dealing with and counting bytes is a seductively simple idea
- Introducing several 8 bit character sets saves the 8 bit world
  - by introducing several character sets, each of them can remain 8 bit
  - the complexity has now been shifted to the handling of various character sets

## ISO 8859 (11)

- A family of character sets rather than a single character set
  - each ISO 8859 family member is an 8 bit character set (256 characters)
  - the lower half (128 characters) are always the same (ASCII)
  - the upper half is supporting different user groups and changes between versions
- ISO 8859 files cannot be identified by inspection
  - ASCII characters can always be safely interpreted (identical on all ISO 8859 code pages)
  - the upper half can only be interpreted if the code page is well-known
- ISO 8859 environments must carefully track the code pages being used
  - failure to do so results in misinterpretation of characters

It also shows the Euro sign € which is part of ISO 8859-15 (Latin-9), but not included in ISO 8859-1 (Latin-1).

It also shows the Euro sign ¤ which is part of ISO 8859-15 (Latin-9), but not included in ISO 8859-1 (Latin-1).

## ISO 8859-1 (Latin-1) & ISO 8859-15 (Latin-9) (12)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3		0	1	2	3	4	5	6	7	8	9	:	;	<	=	>
4		@	A	B	C	D	E	F	G	H	I	J	K	L	M	N
5		P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^
6		`	a	b	c	d	e	f	g	h	i	j	k	l	m	n
7		p	q	r	s	t	u	v	w	x	y	z	{		}	~
8																
9																
A		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	®	¯	°
B		±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C		À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î
D		Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E		à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î
F		ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Latin-1 (Western European)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3		0	1	2	3	4	5	6	7	8	9	:	;	<	=	>
4		@	A	B	C	D	E	F	G	H	I	J	K	L	M	N
5		P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^
6		`	a	b	c	d	e	f	g	h	i	j	k	l	m	n
7		p	q	r	s	t	u	v	w	x	y	z	{		}	~
8																
9																
A		¡	¢	£	€	¥	¦	§	¨	©	ª	«	¬	®	¯	°
B		±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
C		À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î
D		Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
E		à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î
F		ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Latin-9

# Unicode

---

## ISO 8859 Problems (14)

- One document can only contain characters from one character set
  - mixing characters from different sets is impossible

In Latin-9, Latin-1's currency symbol ¤ has been replaced with the Euro sign €.

- An increasing number of character sets does not make life easier
  - in particular, if they sometime differ only slightly (e.g., Latin-1 vs. Latin-9)
- For bigger character sets, the 8 bit approach is not working at all
  - the ISO 8859 approach allows only 128 special characters (the lower half is ASCII)
- ISO 8859 is as good as it gets with 8 bit
  - to improve this approach, the 8 bit philosophy must be abandoned

## Unicode (15)

- Generalization takes characters beyond one and even two bytes
  - Unicode has been designed to cover all characters of the world
  - Unicode recently added its 100'000<sup>th</sup> character
  - for handling this character set, a more structured approach is required
- Unicode cleanly separates various conceptual steps
  - characters are collected and are then part of the *character repertoire*
  - characters are then identified by a unique *code point* (written as U+0041)
  - a *Character Encoding Scheme (CES)* then maps the *Coded Character Set (CCS)* based on a *Character Encoding Form (CEF)*
- “[XML is ASCII for the 21<sup>st</sup> century](#) [../xml-fall08/basics#(6)]”
  - purists sometimes consider Unicode too big or dangerous
  - Unicode is well-established and is necessary in a globalized economy

## Unicode Character Count (16)

- Unicode has the ability to encode  $17 \times 2^{16} = 1'114'112$  characters
  - this means that currently ~10% of the available space is used
- Characters are organized into 17 "planes" of  $2^{16} = 65'536$  characters
- Planes are numbered from "0" to "16"
- Plane 0 is the *Basic Multilingual Plane (BMP)*
  - all characters which in practical use today are part of the BMP (well, [almost ...](http://www.tlg.uci.edu/~opoudjis/unicode/unicode_astral.html) [[http://www.tlg.uci.edu/~opoudjis/unicode/unicode\\_astral.html](http://www.tlg.uci.edu/~opoudjis/unicode/unicode_astral.html)])
- Planes beyond the BMP contain rare and historic characters
  - "[Old Italic](http://unicode.org/charts/PDF/U10300.pdf)" [<http://unicode.org/charts/PDF/U10300.pdf>]", "[Deseret](http://unicode.org/charts/PDF/U10400.pdf)" [<http://unicode.org/charts/PDF/U10400.pdf>]", "[Byzantine Musical Symbols](http://unicode.org/charts/PDF/U1D000.pdf)" [<http://unicode.org/charts/PDF/U1D000.pdf>]"
  - most space within these "astral planes" is empty

# Internationalization (I18N) Internationalization (I18N)

## What is Language? (18)

- Language is more than the encoding of individual words
  - languages and culture are deeply intertwined and inseparable
  - ideally, systems/solutions should adapt to culture, not only to language
  - on a superficial level, adapting to language is useful a first step
- Languages have properties which are beyond character sequences
  - for right-to-left languages, screen layout should be done right-to-left
  - if languages are mixed, one language has to be the preferred language
- Language identification and selection are basic I18N tasks
  - the "one language fits all" assumption is becoming increasingly inappropriate
  - the "just switch the labels" strategy also may be too little for true [L10N](#) [Localization (L10N) (1)]

## Beyond Language (19)

- Directionality is the concept how written language is organized
  - early languages had no inherent directionality (even zigzag writing was possible)
  - the majority of today's languages are *left-to-right* languages
  - Arabic and Hebrew are two popular *right-to-left* languages
  - Chinese can be written *top-to-bottom* and *right-to-left*
  - Mongolian (using the [Uighur alphabet](#) [http://www.linguamongolia.co.uk/]) is written *top-to-bottom* and *left-to-right*



- Icons can be very culture-specific and often need to be localized as well



owls)

- icons are pictorial metaphors (rooted in language and/or culture)
- language-specific metaphors may not work across languages (e.g., [OK gesture](#) [http://customsholidays.suite101.com/article.cfm/international\_ok\_gestures])
- culture-specific metaphors may not work across cultures (e.g.,

## Directionality and Screen Layout (20)



## Definition

(21)

Internationalization is the design and development of a product, application or document content that enables easy localization for target audiences that vary in culture, region, or language.

- I18N starts at the design phase and influences the complete process
- The upfront costs of I18N are considerable (increased complexity)
- The costs of retroactive I18N are much higher than that

## I18N Tasks

(22)

1. UI elements (windows, menus) must be modified to accept translated text
2. Static text must be made configurable
3. Icons and graphics must be changed to be more culturally appropriate
4. Sound files that contain spoken language must be re-recorded
5. Online help must be translated
6. Dynamic text (dates, times) must be formatted using the locale
7. Text handling code must calculate word breaks using the locale
8. Tabular data must be sortable using the locale

# Localization (L10N)

---

## Definition (24)

---

Localization refers to the adaptation of a product, application or document content to meet the language, cultural and other requirements of a specific target market (a "locale").

- Without proper I18N, L10N is a very expensive and risky process
- L10N based on internationalized products should be straight-forward
- during L10N for specific locales, new areas for I18N might be identified
  - some non-internationalized part of the product is identified as inappropriate
- L10N should be regarded as an input for improved I18N

## L10N Tasks (25)

---

1. Create translations for all interface elements
2. Translate all static texts
3. If necessary, create localized icons and graphics
4. Any spoken text must be recorded in the target language
5. Make sure that the localized product uses the localized online help
6. Formatting of data types must be treated locale-specific
7. If necessary, dictionaries and other language tools must be integrated
8. Sorting functions in the code must respect the locale

## Conclusions

**(26)**

- Characters and Character Sets are essential for the Internet
- Unicode solves many problems by supporting many characters
- Internationalization is important for a global economy
- Localization adapts software/content to a specific locale