

Web Internationalization



Abstract

This is an introduction to internationalization on the World Wide Web. The audience will learn about the standards that provide for global interoperability and come away with an understanding of how to work with multilingual data on the Web. Character representation and the Unicode-based Reference Processing Model are described in detail. HTML, including HTML5, XHTML, XML (eXtensible Markup Language; for general markup), and CSS (Cascading Style Sheets; for styling information) are given particular emphasis.

Web Internationalization

Slide 2

Objectives

- Describe the standards that define the architecture & principles for I18N on the web
- Scope limited to markup languages
- Provide practical advice for working with international data on the web, including the design and implementation of multilingual web sites and localization considerations
- Be introductory level
 - Condense 3 hours to 75-90 minutes.

This presentation and example code are available at:
www.xencraft.com/training/webstandards.html

Web Internationalization – Standards and Practice

Slide 3

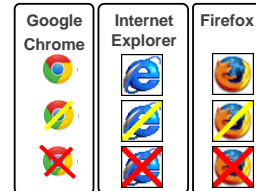
Legend For This Presentation

Icons used to indicate current product support:

Supported:

Partially supported:

Not supported:



CAUTION Caution

Highlights a note for users or developers to be careful.

Web Internationalization

Slide 4

How does the multilingual Web work?

- How does the server know
 - my language?
 - The encodings my browser supports?
- Which Unicode normalization is best?
- Are all Unicode characters useful on the Web?
- Should I use character escapes or text?
- Can CSS designs support multiple languages?

Web Internationalization

Slide 5

A Simple HTML Example Page

Nous espérons que vos applications e-business fonctionneront en français.

Les Français achètent les produits suivants sur internet :

vêtements

produits de beauté

We hope your e-business applications will work in French.

French people buy the following products on the internet:

clothing

beauty products

Don't forget to put prices in Euros (€) not Dollars (\$)

Web Internationalization

Slide 6

Web Internationalization

A Simple HTML Example Page

Here is how the same HTML looks in Japan

Nous esp^望rons que vos applications e-business fonctionneront en fran^蘭is.
 Les Fran^蘭is ach^羅ent les produits suivants sur internet :
 v^黎tements
 produits de beaut[・]/b>

We hope your e-business applications will work in French.

French people buy the following products on the internet:
 clothing
 beauty products

Don't forget to put prices in Euros (€) not Dollars (\$)

Web Internationalization Slide 7

A Simple HTML Example Page

Here is how the same HTML looks in Japan

Nous esp^望rons que vos applic[・] en fran^蘭is.
 Les Fran^蘭is ach^羅ent les pr^黎v^黎tements
 produits de beaut[・]/b>

We hope your e-business ap

French people buy the follow
 clothing
 beauty products

Don't forget to put prices in

The browser has no information about the encoding of the web page. It uses a default value which in this case, is very wrong and even confuses the markup (see beauté).

Web Internationalization Slide 8

A Simple HTML Example Page

Here is how the same HTML looks in Japan

Nous esp^望rons que vos appli en fran^蘭is.
 Les Fran^蘭is ach^羅ent les p^黎v^黎tements
 produits de beaut[・]/b>

We hope your e-business a

French people buy the follo
 clothing
 beauty products

Don't forget to put prices in Euros (€) not Dollars (\$)

Some of the problems may not be obvious to the reader. Changing the euro symbol to a bullet, might cause a significant financial error.

Web Internationalization Slide 9

Character Encodings

- Many character sets exist and in popular use
- Many encoding schemes, even for 1 character set

ISO 8859-1 ≈ IBM 850
 ISO-2022-JP, Shift_JIS, EUC-JP (JIS X-0208-1997)
 UTF-8 = UTF-16 = UTF-32

- Given just bytes, the character set and the encoding scheme can be indeterminate.

How can a browser know how to decode a web page?

Web Internationalization Slide 11

What Character Does 0xFF Represent?

Given just bytes, the character set and encoding are indeterminate.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

FF

West European Code Page ISO 8859-1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

FF

Russian Code Page 1251

Web Internationalization Slide 12

Encoding Identification

Given just bytes, encoding is indeterminate.

- How can an encoding be identified?

There are 2 requirements:

- Agreement on **names** for encodings
- Mechanisms for **labeling** text with encoding

Web Internationalization Slide 13

Web Internationalization

Character Encoding Names

IANA Internet Assigned Numbers Authority

- Maintains registry of official names for character sets (actually encodings) used on the internet and in MIME (mail)

Registry Names

- ASCII, printable characters
- Case-insensitive
- Maximum length 40 characters
- Aliases (alternative names) also registered
- The preferred name is indicated

www.iana.org/assignments/character-sets

Web Internationalization

Slide 14

Unregistered Encoding Names

- Conventions for Unregistered Character Encoding Names
 - Name begins with "x-"
 - Example: "x-**Tex-Yves-encoding**"
 - Useful for private encodings or very new encodings



Not useful on the web, except for private exchange

Web Internationalization

Slide 15

Character Encoding Names

- IANA Name and Alias Examples
 - ISO_8859-1:1987 (ISO_8859-1, ISO-8859-1, latin1, L1, IBM819, CP819, csISOLatin1)
 - Windows-1252, GB2312, BIG5, BIG5-HKSCS
 - SHIFT_JIS, HP-Legal
 - Extended_UNIX_Code_Packed_Format_for_Japanese
 - Adobe-standard-encoding
 - **UTF-8, UTF-16, UTF-16BE, UTF-16LE, UTF-32**



- Registry contains many useless names
- Preferred names indicated. Use them.

Web Internationalization

Slide 16

Encoding W3C Candidate Recommendation

- Lists encodings and labels user agents must support.
 - Much more restrictive than IANA.
- User agents must not support any others.
- Authors must use UTF-8 and the ASCII case-insensitive "utf-8" label.
- New protocols and formats, and existing formats deployed in new contexts, must use UTF-8 exclusively.

Message: Use UTF-8 going forward!

www.w3.org/TR/encoding/

Web Internationalization

Slide 17

Markup and Encoding Names

- Now we have "official" names for encodings. What do we do with them?
 - HTTP
 - HTML
 - XML
 - CSS
 - Links
 - HTML <LINK>
 - HTML <... HREF>
 - XML <... HREF>

Web Internationalization

Slide 18

HTTP and Encoding Names

HTTP Response

```
200 OK HTTP/1.1
Content-Type: text/html; charset=UTF-8
--- Blank Line
document
...
```

- HTTP Default is ISO-8859-1.
 - Set servers to declare UTF-8.

For information on setting HTTP encoding:

www.w3.org/International/articles/http-charset/index

Web Internationalization – Standards and Practice

Slide 19

Web Internationalization

HTML Encoding Names

HTML (4 and lower versions)


```
<META HTTP-EQUIV="Content-type"
  CONTENT="text/html; charset=UTF-8">
```

- HTML does not specify a default

HTML 5

```
<meta charset="UTF-8">
```

- Must be in the first 1024 bytes of the page
- Use Byte Order Mark instead for UTF-16

 Put encoding declarations as early as possible

- Or prior statements may be decoded incorrectly

Web Internationalization – Standards and Practice Slide 20

CSS and Encoding Name

CSS

- Must be in **first line** of external style sheets

```
@charset "UTF-8";
```

- Byte Order Mark (BOM, U+FEFF) is supported.
- Encoding is unspecified if BOM and @charset conflict.

Web Internationalization – Standards and Practice Slide 21

HTML5: LINKs and Encoding Name

Declaring the charset of a LINKed document

- Deprecated in HTML5
- Link Charset can be incorrect if the document's encoding changes

```
<LINK title="Arabic text"
  type="text/html" charset="ISO-8859-6"
  rel="alternate" href="arabic.htm">
```

```
<A href="http://www.unicode.org" charset="UTF-8">
Unicode</A>
```

Web Internationalization Slide 22

If encodings are specified in more than one way, and possibly with different values, which one has priority?

HTTP? HTML? BOM? Other?

ENCODING PRIORITIES

Web Internationalization Slide 23

HTML5 Encoding Priorities

HTML5 uses the first encoding of:

1. User override for charset
2. HTTP "Content-Type" charset
3. Byte Order Mark
4. Either (Must only be one)
 - <META http-equiv "Content-Type" charset...>
 - <META charset= "UTF-8">
5. Character set-detecting heuristics

Web Internationalization Slide 24

HTML4 Encoding Priorities

HTML uses the first encoding of the following:

1. HTTP "Content-Type" charset
2. <META http-equiv "Content-Type" charset>
3. LINK or other syntax for external documents
4. Charset-detecting heuristics

- Many user agents (browsers) support a user override for charset (highest priority)

Web Internationalization Slide 25

Web Internationalization

CSS Encoding Priorities

CSS external style sheets use the first encoding of:

1. HTTP "Content-Type" charset
2. **BOM**/**@charset** rule in the style sheet
3. LINK or other syntax in referencing document
4. Charset of the referencing document
5. **Assume UTF-8**

Web Internationalization Slide 26

Language and Encoding Negotiation

Web Internationalization Slide 27

Typical Browser-Server HTTP Sequence

Web Internationalization Slide 28

Typical Browser-Server HTTP Sequence

1. Browser issues GET URL
2. Server sends RESPONSE
3. Browser displays document in RESPONSE
4. Browser POSTs Form with user data
5. Web Server receives data, database application stores values.

*Which encoding is sent by the server?
Which encoding is returned by the browser?
Which languages can the user use?*

Web Internationalization Slide 29

Language and Encoding Negotiation

Accept Charset x,y,z Browser Get URL Server HTML Pages

```
GET / HTTP/1.1
Accept-Language: en-us,en,hr;q=0.5
Accept-Charset: iso-8859-1,utf-8;q=0.75,*;q=0.5
```

The browser's HTTP GET request can list the languages and the encodings it can make use of, to guide the server.

- "q" is a relative measure of the usefulness (quality) of an entry.

The above example indicates:

- US English preferred, other English, Croatian are also ok.
- ISO 8859-1 preferred, then UTF-8, then anything else.

Web Internationalization Slide 30

Language and Encoding Negotiation

- Most browsers let you set your language preferences and priorities
- Encoding capabilities are not settable (since they are software dependent).
- **New in 2012: Browsers, except Google Chrome, have stopped sending accept-charset.**
- **UTF-8 preferred.**

Web Internationalization Slide 31

Web Internationalization

Language and Encoding Negotiation

```

200 OK HTTP/1.1
Content-Type: text/html; charset=UTF-8
Content-Language: en
--- Blank Line
HTML document
    
```

The server returns a document with language and encoding declared in the RESPONSE header.

- Web administrators or content authors need to configure the server with document properties.
- Use HTML `<lang="xx">`. Don't rely on HTTP header.

Web Internationalization Slide 32

Language and Encoding Negotiation

The browser adapts the document for operating system display.

Web Internationalization Slide 33

Now that you have the Web page and a form to fill out, there is:

FORM DATA SET SUBMISSION

Web Internationalization Slide 34

Form Data Set Submission

The browser also accepts user data in HTML `<FORM>` and can send it to the server as a Form Data Set.

A Form Data Set is a series of control name/current value pairs, for "successful" controls.

There are 3 ways browsers submit form data sets.

Web Internationalization Slide 35

Form Data Set

```

<form name="input" method="GET"
action="http://www.xencraft.com/cgitest"
enctype="application/x-www-form-urlencoded">
Name: <input type="text" name="Name" size="10" />
<input type="radio" name="sex" value="m"> Male
<input type="radio" name="sex" value="f"> Female
<input type="submit" value="Send">
</form>
    
```

Name: Male Female

Form Data Set = Control Name/Current Value Pairs
 Name/Tex
 sex/m

Web Internationalization Slide 36

Form Data Set Submission

3 Submission Methods

GET + HTTP URI
 Form Data Set appended to URI + "?"
 encoded as "application/x-www-form-urlencoded"

POST + HTTP URI
 Form Data Set sent in body, encoded as

- 1) "application/x-www-form-urlencoded", or
- 2) "multipart/form-data" (MIME, RFC 2045)

Web Internationalization Slide 37

Web Internationalization

Form Data Set- GET Method Submission

```
<form name="input" method="GET"
action="http://www.xencraft.com/cgitest"
enctype="application/x-www-form-urlencoded">

Name: <input type="text" name="Name" size="10" />
<input type="radio" name="sex" value="m"> Male
<input type="radio" name="sex" value="f"> Female
<input type="submit" value="Send">
</form>
```

Name: Male Female

This simple form will submit a an HTTP GET with:
<http://www.xencraft.com/cgitest?Name=Tex&sex=m>

Web Internationalization Slide 38

Application/x-www-form-urlencoded Encoding

Name=Value&Name2=Value2&Name3=Value3

- Pairs of control names and current values
- Names separated from values by =
- Name/value pairs separated by &
- Spaces replaced by +
- Line breaks represented as CR LF: %0D%0A
- Non-alphanumeric and non-ASCII characters and '+', '&', '=', have their byte values replaced by %HH
- **Browsers map byte values of the current charset**

If the server doesn't know browser's character encoding, it may decode form data incorrectly.

Web Internationalization Slide 39

Form Data Set Encoding

Application/x-www-form-urlencoded

Name: Male Female

Example comparing two character encodings:

Charset=ISO-8859-1
Name=Fran%E7ois+Ren%E9+Strau%DF

Charset=UTF-8
Name=Fran%C3%A7ois+Ren%C3%A9+Strau%C3%9F

Web Internationalization Slide 40

Form Data Set Submission

Accept Charset x,y,z

Browser → Get URL → Server ← HTML Pages

Browser ← Response → Server ← CHARSET=x

Submit Form (GET or POST)

O/S Charset =z Browser → CHARSET=x → Server

encoding=x-www-form-urlencoded

Modern browsers send x-www-form-urlencoded data to the server in the CHARSET that was determined to be that of the *form*, however that determination was made (HTTP, <meta>, default, user override).

Web Internationalization Slide 41

Form Data Set Submission

Accept Charset x,y,z

Browser → Get URL → Server ← HTML Pages

Browser ← Response → Server ← CHARSET=x

Submit Form (POST)

O/S Charset =z Browser → CHARSET=x → Server

multipart/form-data (MIME)

Each control name/current value pair is a separate part. Each part can be a different charset or content-type encoding.

Supports file uploading (RFC1867).

Web Internationalization Slide 42

Form Data Set Submission

Multipart/form-data

- More efficient than x-www-form-urlencoded for non-ASCII data, binary data, and files
- Does not have the length limit that browsers impose on URLs (can be as low as 250 for some devices)
- Is well supported
- Recommended for POST of all form data

Web Internationalization Slide 43

Web Internationalization

Form Data Set Submission

- **TIP:** Use with older browsers:
- Hidden fields containing encoding name or carefully chosen values
 - Server application can analyze and detect unexpected changes or determine browser encoding
 - Microsoft sets hidden field labeled `_CHARSET_` with browser encoding.

Web Internationalization Slide 44

DATATYPES

Web Internationalization Slide 45

HTML5 INPUT Data types

Input data types and formats defined (for code and exchange) including:

- Date, Time and Timezone related Input using RFC 3339 / ISO8601 formats and subsets
 - yyyy-mm-ddThh:mm:ss.s±hh:mm or yyyy-mm-ddThh:mm:ss.z**
 - 2016-12-19T16:39:57-08:00
 - 2016-12-31T23:59:60Z
- Numeric and Range Input in floating point format and subsets
 - dddd.ddddE±dd

Web Internationalization Slide 46

Input Types

Keyword	Data type
datetime	Date and time (year, month, day, hour, minute, second, fraction of second) with time zone set to UTC
date	Date (year, month, day) with no time zone
month	Date consisting of year and month with no time zone
week	Date consisting of week-year number and week number with no time zone
time	Time (hour, minute, seconds, fractional seconds) with no time zone
datetime-local	Date and time (year, month, day, hour, minute, second, fraction of a second) with no time zone
number	Numerical value
range	Numerical value, with the extra semantic that the exact value is not important

Date Time Example

```
<p>Schedule Appointment:</p>
<form action="example-date.html">
  Date: <input type="date" name="day">
  Time: <input type="time" name="time">
  <input type="submit" value="Schedule">
</form>
```

Submits values like:

example-date-simple.html?day=2016-11-01&time=19%3A44

Web Internationalization Slide 48

Date and Time User Interface Examples

Schedule Appointment: (From Microsoft Edge 2016)

Controls used Windows regional settings and not HTML lang. Setting DIR=RTL was confusing. Other browsers didn't support or behaved differently.

Web Internationalization Slide 49

BCP47

LANGUAGE IDENTIFIERS

Web Internationalization Slide 50

Language Identification

- Many standards groups involved:
 - IETF, ISO TC37, SIL, W3C, et al
- BCP47 leverages language, region, script standards to define a language identifier

`language-extlang-script-region-variants-extensions-privateuse`

`en-US` `ar-OM`

- Don't confuse language & locale identifiers
See www.w3.org/International/articles/language-tags/

Web Internationalization Slide 51

BCP47 Language Identifiers

`language-extlang-script-region-variants-extensions-privateuse`

Subtag	Standard	Syntax	Examples
Language	ISO 639	2 or 3 letter code	en, yue
Extlang	ISO 639-2	3 letter code	(Legacy only) zh-yue
Script	ISO 15924	4 letter code	Latn, Cyrl, Hans, Hant
Region	ISO 3166 UN M49	2 letter code 3 digit code	US, GB 419
variants			
extensions			
privateuse			

<http://www.iana.org/assignments/language-subtag-registry>

Web Internationalization Slide 53

Example Language Identifiers

Tag	Language	Tag	Language
en	English	zh	Chinese
en-US	American English	zh-Hant	Traditional Chinese
es-US	Spanish as spoken in U.S.	zh-Hans	Simplified Chinese
en-CA	Canadian English	cmn	Mandarin
fr-CA	Canadian French	yue	Cantonese
fr-FR	French French	cmn-Hans-CN	Mandarin for China in Simplified Chinese
es-ES	Iberian Spanish	cmn-Hant	Mandarin in Traditional Chinese
es-419	Latin American Spanish	pt-BR	Brazilian Portuguese
es-MX	Mexican Spanish	zh-yue	retired, use yue instead
		zh-CN	Chinese spoken in China

Web Internationalization Slide 54

Language Declaration

Declaring language aids browsers, search engines, et al to correctly categorize, render, hyphenate, and process your content.

- HTTP: Content-Language header
 - HTTP-EQUIV **no longer recommended** in HTML
- HTML: lang attribute (**Recommended!**)
- XML: xml:lang attribute
- XHTML 1.0: Both lang and xml:lang


```
<p xml:lang="es" lang="es">Hola</p>
```
- XHTML 1.1: xml:lang attribute

Web Internationalization Slide 55

Language Selection– CSS

Lang attribute enables style based on language

lang pseudo-class:

```
*:lang(zh) { font-family: SimSun }
```

- Matches values beginning with tag including inherited

attribute selector:

- Matches explicitly set attributes
- |= matches values that begin with. = matches exactly

```
*[lang|=fr] { font-weight:bold }
```

```
*[lang=fr] { font-weight:bold }
```

Both use the same matching mechanism as the lang() function in XPath.

Web Internationalization Slide 56

Web Internationalization

CSS lang example code

```
<style>
  *:lang(en-us) { font-weight: bold; color: green;}
  *[lang|=fr] { font-style: italic; color: red; }
</style>
</head>
<body>

<table>
<caption>Demonstration</caption>
<tr><th>Lang Tag</th> <th> Example</th></tr>
<tr><td>en</td> <td> Text in generic English.</td></tr>
<tr><td>en-US</td> <td lang="en-US"> Text in American English.</td></tr>
<tr><td>fr</td> <td lang="fr"> Texte en français.</td></tr>
<tr><td>fr-CA</td> <td lang="fr-CA"> Texte en québécois.</td></tr>
<tr><td>en-GB</td> <td lang="en-GB"> Text in British English.</td></tr>
</table>
```

Web Internationalization

Slide 57

CSS lang example code result

Style Settings

lang(en-us) is bold green
[lang|=fr] is italic red
everything else is normal black

Demonstration

Lang Tag	Example
en	Text in generic English.
en-US	Text in American English.
fr	<i>Texte en français.</i>
fr-CA	<i>Texte en québécois.</i>
en-GB	Text in British English.

Web Internationalization

Slide 58

Acquiring User Language Preferences

- User explicitly declares by:
 - Via a language control (e.g. button, menu, etc)
 - The page's domain name or URL, etc.
 - E.g. es.xencraft.com, xencraft.mx, xencraft.com/a?lng=es
 - Preferences declared in User Profile
 - Product default settings applied to user
- Heuristics
 - Device settings, IP Geolocation, et al
 - Browser settings (Accept-language)
 - Desktop settings are unreliable. Mobile may be better.

Web Internationalization

Slide 59

Language Tag Matching

- HTML: truncate language subtags until match found
- RFC 4647 provides more complex matching rules
- Other languages (JavaScript, et al, use other rules)
 - E.g. Java fallback truncates language tags. If no match found, uses a default language, truncates again.
 - *UI->fr-ca truncate-> fr default-> en-us truncate -> en*

Web Internationalization

Slide 60

A Model for Designing and Implementing Textual Software Independent of Encodings

REFERENCE PROCESSING MODEL

Web Internationalization

Slide 61

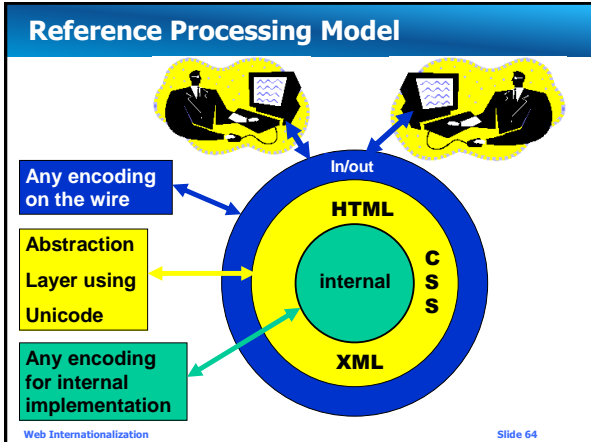
Reference Processing Model

- Logically, characters are Unicode characters
 - Specifications are in terms of Unicode characters
 - Implementations do NOT have to use Unicode, only **behave as if** they did
- Benefits
 - Removes ambiguity, simplifies specifications
 - Allows flexibility for common local encodings
 - Backward compatible for older HTML browsers
 - Supports internationalization (large character set)
 - Removes dependencies/orientation on byte values

Web Internationalization

Slide 63

Web Internationalization



Reference Processing Model Examples

- HTML declares Unicode as its SGML Document Character Set
- CSS "sequence of characters from UCS"
- XML "A character is an atomic unit of text as specified by ISO/IEC 10646"
- XML requires parsers to accept UTF-8 and UTF-16
- DOM requires UTF-16
- Any encoding can be used internally, but Unicode makes the most sense.

Web Internationalization Slide 65

CHARACTER ESCAPES

Web Internationalization Slide 66

Character Escaping

Mechanisms to represent characters

- Numeric Character References (NCRs)
 - HTML and XML
 - Hexadecimal: `&#xhhhhh;`
 - Decimal `&#dddd;`
 - CSS2
 - "`\hh`" (note terminating space), `hhhhh`
- Named Character References (HTML5)
 - aka Character Entity References in HTML4
 - `å`; `Å` (note case-sensitivity)

Web Internationalization Slide 67

Character Escaping

Useful for:

- syntax-significant characters
 - e.g. `<`; (`<`), `>`; (`>`), `&`; (`&`), `"`; (`"`)
- characters outside current encoding
- eliminating visual or other ambiguity
 - `­` (soft-hyphen),
 - `-` (hyphen-minus)

 - ` ` (space)
 - ` ` (no-break space)

Web Internationalization Slide 68

Character Escaping

- Relies on Reference Processing Model
 - Always references Unicode scalar value
 - Same value regardless of encoding
 - Same value for UTF-8, UTF-16, UTF-32
 - One value for supplementary characters, not two
 - E.g. `𒍅` not `��`
 - Simplifies transcoding (no parsing or conversion)
 - Allows any Unicode character in any document (if it is legal in the language of the document)

Web Internationalization Slide 69

String Indexing

How long is this string? 丌 ≠ Å

Web Internationalization Slide 80

String Indexing

- Which units should be used for counting?


Graphemes 3	丌	≠	Å
Characters 4	U+233B4	U+2260	U+0041 U+030A
Code units 5	D84C DFB4	2260	0041 030A
Bytes 10	D8 4C DF B4	22 60	00 41 03 0A

Web Internationalization Slide 81

String Indexing

Character Model recommendations

- Character counting is recommended for most programming interfaces (e.g. XML Path)
- Code unit counting may be used for internal efficiency (e.g. DOM)
- Graphemes may be useful for user interaction, once a suitable definition exists

 Avoid creating API with single unit arguments
e.g. "SS" = Uppercase("ß")

Web Internationalization Slide 82

Unicode Normalization

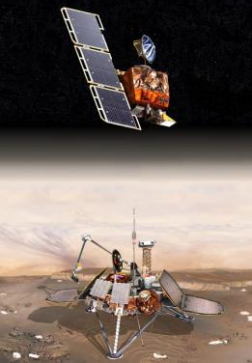
W3C Full Normalization

NORMALIZATION

Web Internationalization Slide 83

Normalization

- Representing data in more than 1 way leads to errors
- E.g. The Mars Climate Orbiter mission was disastrous. Information expected to be metric, was sent in English units
- Solution- Adopt a standard representation- **Normalize**



Web Internationalization Slide 84

Canonical & Compatibility Normalization

Unicode characters can have more than 1 representation

- Canonical equivalence
 - Indistinguishable, fundamental equivalence
 - E.g. combining sequences, singletons
 - “Å” U+00C5 (A-ring pre-composed)
 - “A+°” U+0041 + U+030A (A + combining ring above)
 - “Å” U+212B (Angstrom)
- Compatibility equivalence
 - E.g. Formatting differences, ligatures
 - “カ” U+FF76 “力” U+30AB (KA half and full width)
 - “fi” U+FB01 (ligature fi)

Web Internationalization Slide 85

Web Internationalization

Unicode Normalization Forms

- Unicode Consortium has defined canonical and compatibility decomposition formats and 4 different sets of rules for normalization:

“Unicode Normalization Forms”

<http://www.unicode.org/unicode/reports/tr15/>

	Composed	Decomposed
Canonical	NFC	NFD
Canonical+ Kompatibility	NFKC	NFKD

Web Internationalization

Slide 86

W3C Normalization

The W3C Character Model recommends **Normalization Form C (NFC)**

- w3c.github.io/charmod-norm/
- Brings canonical equivalences to composed form
- Leaves compatibility forms as distinct
- Most legacy text is composed, so unchanged

	Composed	Decomposed
Canonical	NFC	NFD
Canonical+ Kompatibility	NFKC	NFKD

Web Internationalization

Slide 87

Fully Normalized Text

Authors SHOULD create **Fully Normalized** text

Fully Normalized text is either:

- Unicode text in Normalization Form NFC, and
- Does not contain character escapes or includes that upon expansion would undo point 1, and
- Does not begin with a composing character.

or:

- Legacy encoded text, which transcoded to Unicode satisfies the above.

Web Internationalization

Slide 88

Fully Normalized Text Examples

Fully Normalized Text

`"suçon", "suçon",
"sub,on", "sub̧on"`

Note- Unicode does not have a composed b-cedilla.

Not Fully Normalized Text

`"suc,on", "suçon"`

Reason: should use composed character "ç"

`",on", "̧on"`

Reason: should not begin with combining character

Web Internationalization

Slide 89

IRI

INTERNATIONAL RESOURCE IDENTIFIERS

Web Internationalization

Slide 94

Resource Identifiers: URI, IRI

`<scheme>://<authority><path>?<query>#<fragment>`

URIs encode bytes, not characters

- Most ASCII bytes expressed as ASCII
- Non-ASCII are %HH, which is ambiguous
 - Character encoding not taken into consideration

IRI-Internationalized Resource Identifiers

- Use Unicode NFC (not NFKC) Normalization
- Transcode to UTF-8, then encode as URI

<http://Xencraft.com/Fran%C3%A7ois/Ren%C3%A9?s=m>

- <http://www.w3.org/International/O-URL-and-ident.html>

Web Internationalization

Slide 95

Web Internationalization

IRI Query, Fragment

<scheme>://<authority><path>?<query>#<fragment>

- The standard could not insist on UTF-8 for reasons of backward compatibility and semantics
 - The values can be used for example by cgi programs or database queries or as identifiers within documents, that use other encodings.



Be careful globally changing text that includes URIs to UTF-8, and don't inadvertently change query or fragments that require other encodings.

Web Internationalization

Slide 96

<http://globalização.biz/>

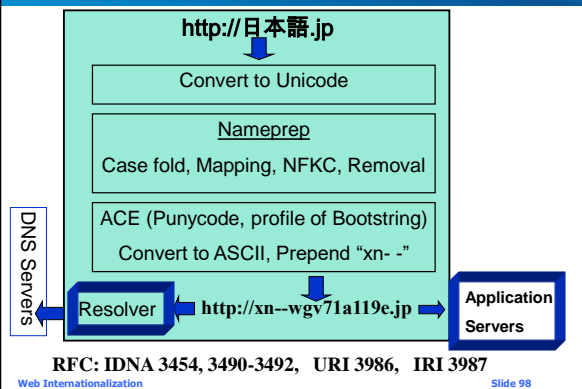
<http://xn--globalizao-n5a1c.biz>

**INTERNATIONALIZED
DOMAIN NAMES**

Web Internationalization

Slide 97

Domain Names: IDNA Architecture



Web Internationalization

Slide 98

Internationalized Domain Names

- Considerations for IDN
 - Upgrade network libraries or convert to ASCII domains before calling them
 - Normalize domains for listing, comparison, etc.
 - May want to present users both forms for ease of recognition
 - ASCII form may be more useful for copy/paste to older software

<http://globalização.biz/>

<http://xn--globalizao-n5a1c.biz>

Web Internationalization

Slide 99

E-mail Addresses

- In addition to international domain names, the local part (aka user name) of e-mail addresses support international characters
- Local part:
 - UTF-8
 - Normalization Form NFC
 - Additional restrictions may be imposed by the server

RFC 5890, 6530, 6531, 6532, 6533, 6855, 6856, 6857, 6858

Web Internationalization

Slide 100

Bidi

**BIDIRECTIONAL LANGUAGE
SUPPORT**

Web Internationalization

Slide 105

Web Internationalization

Bidirectional (Bidi) Language Support

- HTML 4 DIR attribute
 - `dir="ltr" | dir="rtl"`
 - Sets base direction
 - Direction is inherited
- Direction affects alignment and flow
 - Ordering of text and table columns
 - Text alignment, Alignment of overflowing blocks
- Control Characters
 - Right to Left and Left to Right Marks `‏` & `&rlrm;`
 - Useful for correct positioning of neutrals

Web Internationalization

Slide 106

Bidirectional (Bidi) Language Support

- HTML 5 – Isolates
 - `<bdi dir=rtl> </bdi>`
 - Flow doesn't change with container changes!
- DIR=AUTO
 - Detects direction, based on first strong character
- CSS Selectors
 - `:dir(rtl)` for rtl elements
 - `:dir(ltr)` for ltr elements

Web Internationalization

Slide 107

Bidi Example

Bidi Example

The source contents of each table are identical except for DIR attributes of LTR vs RTL on the table element.

Rows 3 and 4 set bdir dir=ltr in each cell.

`dir(RTL) { color:blue; }`, everything else is normal black

LTR			RTL		
First	Second	Third	Third	Second	First
1	John has \$1,234.00.	10 - 5 = 3	3 = 5 - 10	John has \$1,234.00	1
2	Joe has (1,234.00).	10 - 5 = 3	3 = 5 - 10	Joe has (1,234.00)	2
3 bdi dir=ltr	John has \$1,234.00.	10 - 5 = 3	10 - 5 = 3	John has \$1,234.00	bdi dir=tr 3
4 bdi dir=ltr	Joe has (1,234.00).	10 - 5 = 3	10 - 5 = 3	Joe has (1,234.00)	bdi dir=tr 4
5 show alignment	John has \$1,234.00 and Jill has \$9,876.54.	10 - 5 = 3	3 = 5 - 10	John has \$1,234.00 and Jill has \$9,876.54	show alignment 5

Web Internationalization

Slide 108

Bidi Example Code

```

<dir(RTL) { color:blue; }

<style>
</style>
</head>
</body>

<table class="inner" dir="LTR">
<caption>LTR</caption>
<tr><th>First</th><th>Second</th><th>Third</th></tr>
<tr><td>1</td> <td>John has $1,234.00.</td><td>10 - 5 = 3</td></tr>
<tr><td>2</td> <td>Joe has (1,234.00).</td><td>10 - 5 = 3</td></tr>
<tr><td>3 bdi dir=ltr</td> <td>John has $1,234.00.</td><td>bdi dir=tr 10 - 5 = 3</td></tr>
<tr><td>4 bdi dir=ltr</td> <td>Joe has (1,234.00).</td><td>bdi dir=tr 10 - 5 = 3</td></tr>
<tr><td>5 show alignment</td> <td>John has $1,234.00 and Jill has $9,876.54.</td><td>10 - 5 = 3</td></tr>
</table>

<table class="inner" dir="RTL">
<caption>RTL</caption>
<tr><th>First</th><th>Second</th><th>Third</th></tr>
<tr><td>1</td> <td>John has $1,234.00.</td><td>10 - 5 = 3</td></tr>
<tr><td>2</td> <td>Joe has (1,234.00).</td><td>10 - 5 = 3</td></tr>
<tr><td>3 bdi dir=ltr</td> <td>John has $1,234.00.</td><td>bdi dir=tr 10 - 5 = 3</td></tr>
<tr><td>4 bdi dir=ltr</td> <td>Joe has (1,234.00).</td><td>bdi dir=tr 10 - 5 = 3</td></tr>
<tr><td>5 show alignment</td> <td>John has $1,234.00 and Jill has $9,876.54.</td><td>10 - 5 = 3</td></tr>
</table>
    
```

Web Internationalization

Slide 109

Bidi References

- Presentation by Erika J. Etemad (fantasai) Mozilla Project W3C CSS Working Group
 - <http://fantasai.inkedblade.net/style/talks/bidi/>
- Unicode Standard Annex #9
 - <http://www.unicode.org/reports/tr9/>
- Additional Requirements for Bidi in HTML
 - <http://www.w3.org/TR/html-bidi/>
- W3C Bidi Tutorial
 - www.w3.org/International/tutorials/bidi-xhtml/

Web Internationalization

Slide 110

Example Code and Tests

Richard Ishida and W3C I18n feature tests for numerous browsers and versions (X)HTML: www.w3.org/International/tests/

More content and example code are available at: www.xencraft.com/training/webstandards.html

Feature
Lang()
Lang pseudo-class
Lang attr selector
Quote:qo
Text-transform
CSS list-style-type
Xsl number

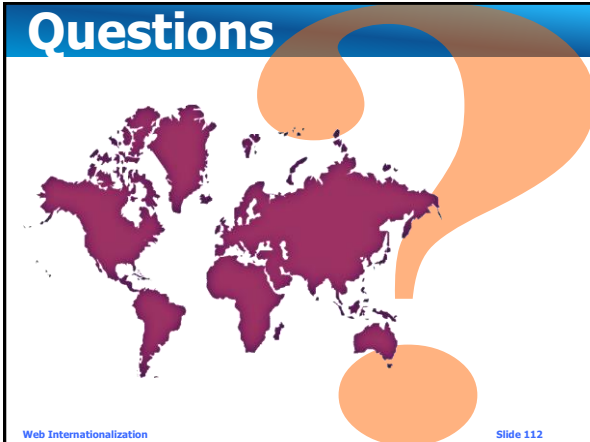
Feature
Xsl format-number
Html bi-directional text
CSS bi-directional text
Vertical text (SVG losing ground)
Ruby annotation
CSS3 combined sort
Xsl:sort

Web Internationalization

Slide 111

Web Internationalization

Questions



Web Internationalization Slide 112

Tex Texin

"XenCraft", "TexTexin" and "118nGuy" are Trademarks of Tex Texin.

TexTexin@Xencraft.com

Tex is an industry thought leader specializing in business and software globalization services. His expertise includes global product strategy, Unicode and internationalization architecture, and cost-effective implementation and testing. Over the past two decades, Tex has created numerous global products, led internationalization development teams, and guided companies in taking business to new regional markets.

Tex is a contributor to internationalization standards for software and on the Web.

Tex is a popular speaker at conferences around the world and provides on-site training on Unicode, internationalization, and globalization QA worldwide.

Tex is the author of the popular, instructional web site www.118nGuy.com

Tex is founder and Chief Globalization Architect for XenCraft. XenCraft provides global business consulting and software design, implementation, test and training services on globalization product strategy and software internationalization architecture.



Copyright © Tex Texin 2017 All Rights Reserved.

Acknowledgements

- Parts of this presentation are based on "Weaving the multilingual Web: Standards and their implementations" by François Yergeau and Martin Dürst, given at previous Unicode conferences.
- Yves Savourel (ENLASO Corporation) created the test programs and the initial versions of the best practices content.
- Thanks to Richard Ishida and Martin Dürst for their extensive review.

Web Internationalization Slide 114