Tech Day: Universal Acceptance

Mark Švančárek
Today’s Objectives

• Definition of Universal Acceptance
• Universal Acceptance Steering Group
• Challenges
• BiDi Stuff
• Conclusion
Definition of Universal Acceptance

*ALL domain names and ALL email addresses should work in ALL Internet-enabled applications, devices and systems*
Universal Acceptance Steering Group (UASG)

• A community-based team
  • ICANN's role is that of supporter, provider of funds
• Formed to identify topline issues and proposed solutions, and disseminate best practices
  • Objective: Help software developers and website owners update systems to keep pace with evolving Internet standards
  • Message: Universal Acceptance will enable the next billion users build and access their own spaces and identities online
• **UASG.tech**
UASG Activities

- **Review**: Popular Websites, Dev Frameworks, Browsers, OS
- **Build**: Use Cases, Test Environments, EAI Community
- **Outreach**: Live Workshops, Panel Discussions, Presentations
- **Writing**: Knowledge Databases, Whitepapers, Quick Guides
Challenges

• **Technical Challenges**
  • Challenging old assumptions
  • Updating old software
  • Managing backward-compatibility

• **Business Challenges**
  • Understanding the opportunity
  • Evaluating return on investment

Today’s discussion

Learn more at UASG.tech
Technical Challenges – Old Assumptions

- Sometimes coders make bad assumptions about domain name strings and email address strings
  - This may be because RFCs have changed (e.g. SMTPUTF8)
  - Or standards may be misleading (e.g. HTML5.3 email input type definition)
  - Or standards may not exist (e.g. “linkification”)
- But mostly assumptions are based on previous state of the ecosystem, rather than RFCs (i.e. they may never have been correct assumptions)
Examples of bad assumptions

- **Bad assumptions about TLDs**
  - Length restrictions, script restrictions, maintaining outdated name lists

- **Bad assumptions about email addresses**
  - All of the above (domain name part)
  - Regular expressions which aren’t EAI-aware
  - Over-aggressive spam-filtering when scripts are mixed within or between labels

- **Bad assumptions about linkification**
  - Not understanding user intent
To: Борис@пример.рф
Cc: Марк@实践的伊斯兰教；Марк@实践的伊斯兰教
Bcc: 微软测试@互联网.中国；微软测试@互联网.中国
Subject: Universal Acceptance
From: Lars Steffen - mail@larssteffen.de

HI!
Technical Challenges – Updating Old Software

• It’s usually not hard to update an individual piece of software to use latest versions of Unicode, IDNA, SMTP, etc.
  • Usually, it’s more like a “Bug Fix” than like a “Design Change Request”

• The tricky parts are:
  • Finding **ALL** the instances in the software which use or make assumptions about domain names, URLs, URIs, and email addresses
  • Identifying all the use cases which must be tested
  • Managing bi-directional strings
  • “Linkification”

• No one wants to fix software which is already working unless the business opportunity is clear
Managing backward compatibility:
Email Address
Internationalization (EAI)
Managing Backward Compatibility - EAI

- Email Address Internationalization (EAI) creates a new email stream, parallel to the legacy email stream
  - Services must advertise support for SMTPUTF8
  - SMTPUTF8 systems can interop with SMTP systems, but the reverse is not true

- Attempts to make SMTP systems interop with SMTPUTF8 systems is collectively known as “downgrading”
  - In general it doesn’t work
More about email “downgrading”

• UASG supports a single “downgrading” technique: “Downgrading with Aliasing”
  • An email provider can offer an EAI user an ASCII email alias, and decide “on the fly” which address to use for each To: or CC: destination
  • Coremail and XgenPlus both use this technique
• But other transformations are not allowed
  • Don’t ever attempt to transform an address if you do not manage the mailbox
  • Don’t send ACE encoding (punycode) in the local part
  • If you receive ACE-encoded local parts, don’t transform into a Unicode equivalent
Fun fact

Suppose I want mailbox = “孫悟空” on Outlook.com

– Note that ACE(孫悟空) = “xn--98sy4jmv0a”

Q: Can my non-SMTPUTF8 friend expect xn--98sy4jmv0a@outlook.com to work when sending me email?

A: NO

– xn--98sy4jmv0a@outlook.com is already an existing mailbox, and attempting to use it as a downgrading transformation will cause messages to go to the wrong destination!
– You cannot make assumptions about mailboxes you don’t manage!
Current Status of EAI – Email Address Internationalization

* UASG is creating an EAI evaluation program
  * Evaluate quality of support for non-ASCII mailbox names and good practice around presentations of IDNs

* Phase 1: The ability to send to and receive from EAI Addresses
  * Google, Office365, Outlook.com, Postfix, Exim, Halon, Outlook, and more claim compliance

* Phase 2: The ability to host non-ASCII mailbox names and domain names
  * Coremail, XgenPlus, Raseal, OpenFind, Throughwave all claim compliance
### Examples: Bi-directional Email Addresses

#### Left to Right (LTR) Scripts

<table>
<thead>
<tr>
<th>Username</th>
<th>Domain</th>
<th>TLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>example</td>
<td>.app</td>
</tr>
</tbody>
</table>

#### Right to Left (RTL) Scripts

<table>
<thead>
<tr>
<th>TLD</th>
<th>Domain</th>
<th>Username</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>.مثال</td>
<td>المستخدم</td>
</tr>
</tbody>
</table>

#### More Examples of (imaginary) Email Addresses including IDNs

- `user@example.みんな` (Uses internationalized TLD)
- `user@大坂.info` (Uses internationalized 2nd level domain)
- `用戶@example.lawyer` (Uses internationalized user name and new gTLD)
Hard problem:

Unicode + Bi-Directionality + Linkification
The Unicode Bidi Algorithm (UBA)

* UBA is a very useful, general, and standard approach to displaying text that contains right-to-left scripts, such as Arabic and Hebrew. But there are situations in which it is awkward to use and/or is visually confusing.

* IRLs (internationalized URLs)
  * Also applies to file paths and email addresses in addition to scheme IRIs

http://www.unicode.org/cldr/utility/bidi.jsp
Quick Bidi Intro

* Hebrew/Arabic text is normally displayed right-to-left (RTL)
* Even pure Hebrew & pure Arabic (no foreign words) can contain bidirectional text
* Digits are always displayed “left to right” (LTR) except for N’Ko
* Neutral characters can be displayed LTR or RTL
* Unicode Bidi Algorithm (UBA) specifies the classifications of Unicode characters and their visual layout
* IRIs with schemes like http have LTR
Linkification

* **UASG010** – Quick Guide to Linkification

* Modern software sometimes automatically creates a hyperlink by a user simply typing in a string that looks like a web address, email name or network path.

  EXAMPLE: Typing “www.icann.org” into an email message →
  [http://www.icann.org](http://www.icann.org)

* Application accepted a string and dynamically determined it should create a hyperlink to an Internet Location (URL/IRL)

* Users have expectations and developers need to code for those expectations.
  * In this example, “http:” and “www” were indicators of user intent
What’s the problem?

“Logical” Order*
http://exchange.ث.ق
http://exchange.ث.ق

UBA LTR ¶
http://exchange.ث.ق
http://exchange.ث.ق

UBA RTL ¶
http://exchange.ث.ق
http://exchange.ث.ق

Readable Order
http://exchange.ث.ق
http://exchange.ث.ق
http://exchange.ث.ق
http://exchange.ث.ق

"Logical" Order*
http://exchange.ث.ق
http://exchange.ث.ق

UBA LTR ¶
http://exchange.ث.ق
http://exchange.ث.ق

UBA RTL ¶
http://exchange.ث.ق
http://exchange.ث.ق

Readable Order
http://exchange.ث.ق
http://exchange.ث.ق
http://exchange.ث.ق
http://exchange.ث.ق
# Unicode: Bidirectional Character Types

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Description</th>
<th>General Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strong</td>
<td>L</td>
<td>Left-to-Right</td>
<td>LRM, most alphabetic, syllabic, Han ideographs, non-European or non-Arabic digits, ...</td>
</tr>
<tr>
<td></td>
<td>LRE</td>
<td>Left-to-Right Embedding</td>
<td>LRE</td>
</tr>
<tr>
<td></td>
<td>LRO</td>
<td>Left-to-Right Override</td>
<td>LRO</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>Right-to-Left</td>
<td>RLM, ALM, Hebrew alphabet, and related punctuation</td>
</tr>
<tr>
<td></td>
<td>AL</td>
<td>Right-to-Left Arabic</td>
<td>Arabic, Thaana, and Syriac alphabets, most punctuation specific to those scripts, ...</td>
</tr>
<tr>
<td></td>
<td>RLE</td>
<td>Right-to-Left Embedding</td>
<td>RLE</td>
</tr>
<tr>
<td></td>
<td>RLO</td>
<td>Right-to-Left Override</td>
<td>RLO</td>
</tr>
<tr>
<td></td>
<td>PDF</td>
<td>Pop Directional Format</td>
<td>PDF</td>
</tr>
<tr>
<td>Weak</td>
<td>EN</td>
<td>European Number</td>
<td>European digits, Eastern Arabic-Indic digits, ...</td>
</tr>
<tr>
<td></td>
<td>ES</td>
<td>European Number Separator</td>
<td>Plus sign, minus sign</td>
</tr>
<tr>
<td></td>
<td>ET</td>
<td>European Number Terminator</td>
<td>Degree sign, currency symbols, ...</td>
</tr>
<tr>
<td></td>
<td>AN</td>
<td>Arabic Number</td>
<td>Arabic-Indic digits, Arabic decimal and thousands separators, ...</td>
</tr>
<tr>
<td></td>
<td>CS</td>
<td>Common Number Separator</td>
<td>Colon, comma, full stop (period), No-break space, ...</td>
</tr>
<tr>
<td></td>
<td>NSM</td>
<td>Nonspacing Mark</td>
<td>Characters marked Mn (Nonspacing_Mark) and Me (Enclosing_Mark) in the Unicode Character Database</td>
</tr>
<tr>
<td></td>
<td>BN</td>
<td>Boundary Neutral</td>
<td>Most formatting and control characters, other than those explicitly given types above</td>
</tr>
<tr>
<td>Neutral</td>
<td>B</td>
<td>Paragraph Separator</td>
<td>Paragraph separator, appropriate Newline Functions, higher-level protocol paragraph determination</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td>Segment Separator</td>
<td>Tab</td>
</tr>
<tr>
<td></td>
<td>WS</td>
<td>Whitespace</td>
<td>Space, figure space, line separator, form feed, General Punctuation spaces, ...</td>
</tr>
<tr>
<td></td>
<td>ON</td>
<td>Other Neutrals</td>
<td>All other characters, including OBJECT REPLACEMENT CHARACTER</td>
</tr>
</tbody>
</table>
UBA example

Caps = Arabic Text here in logical order

ADDRESS 1234 56th st.

What users mean (RTL para, display order):
1234 56th st. SSERDDA

What UBA concludes:
.56th st 1234 SSERDDA
Resolving IRIs using UBA

Logical order

http://msn.ARABIC.SA

Display order

http://msn. AS.CIBARA

AS.CIBARA.http://msn
Resolving IRIs Readably

http://msn.ARABIC.SA

http://msn.CIBARA.AS

AS.CIBARA.msn://http
Summary: Possible Readable Layouts

“Fields” flow in consistent direction:

* LTR
* RTL
* First strong character
* Paragraph

User context or predilection may influence preference. Paragraph choice best default.
Further information about UA

* Visit www.uasg.tech
* Email info@uasg.tech
* Subscribe www.uasg.tech/subscribe
* Report problems www.uasg.tech/global-support-centre
* Check out your web site https://github.com/uasg/uac-crawler
* Help define email address regexes https://www.ietf.org/archive/id/draft-seantek-mail-regexen-02.txt
* Get started with Universal Acceptance Quick Guides!