Universal Acceptance Quick Guide

What Does “Universal Acceptance” Mean?

ACCEPT

VALIDATE

STORE

PROCESS

DISPLAY

Software and online services support Universal Acceptance when they offer the capabilities listed above for all domains and email names.

Universal Acceptance (UA) is the state where all valid domain names and email addresses are accepted, validated, stored, processed and displayed correctly and consistently by all Internet-enabled applications, devices and systems.

Due to the rapidly changing domain name landscape, many systems do not recognize or appropriately process new domain names, primarily because they may be more than three characters in length or in a non-ASCII format. The same is true for email addresses that incorporate these new extensions.

The Universal Acceptance Steering Group (UASG), established by Internet Corporation for Assigned Names and Numbers (ICANN), is a community-led, industry-wide initiative working on creating awareness and identifying and resolving problems associated with the universal acceptance of domain names. The purpose of these efforts is to help ensure a consistent and positive experience for Internet users globally.

For more information on the UASG and recent developments, visit: www.uasg.tech.

Note that accept, validate and process are treated as distinct in this document. In actual practice these capabilities may overlap.

ACCEPT

Accept is the process by which an email address or a domain name is received as a string of characters from a user interface, file or an API (application program interface) used by a software application or online service.

UASG Recommendations

- Any user interface element requiring a user to type a domain name or email address must support Unicode and strings up to 256 characters.
- Users should be allowed, but not required, to enter ASCII Compatible Encoded (aka “Punycoded”) text in place of its Unicode equivalent. However, Unicode should be shown by default, with Punycoded text shown to the user only when it provides a benefit.

VALIDATE

The process by which an email address or domain name, received or emitted, is checked for syntax correctness. Many programmers have been trained to validate by following heuristics that require checking that a top-level domain has the "correct" number of letters, or that the letters are from the ASCII character set. These heuristics are no longer applicable because of the introduction of domain names with more than three characters, and Unicode (non-ASCII) characters.

UASG Recommendations

- Validations should not occur unless required for the operation of the application or service. This is the easiest way to ensure that all valid domain names are accepted into systems.
- If validation is required, consider the following:
  - Verify the TLD portion of a domain name against an authoritative table:
    http://www.internic.net/domain/root.zone
    http://www.dns.icann.org/services/authoritative-dns/index.html
    http://data.iana.org/TLD/tlds-alpha-by-domain.txt
    See also SAC070: https://tinyurl.com/sac070.
  - Query the domain name against the Domain Name System (DNS).
  - Require repeated entry of an email address to preclude typing errors.
  - Validate the characters in labels only to the extent of determining that the U-label does not contain "DISALLOWED" code points or code points that are unassigned in its version of Unicode. Visit: https://tools.ietf.org/html/rfc5892.
  - Limit validation of labels to a small number of whole-label rules defined in the Request for Comments (RFCs). Visit: https://tools.ietf.org/html/rfc5894.
  - If a string resembling a domain name contains the ideographic full stop character '。', it should be converted it to a '.' before validation is performed.

STORE

Store refers to the long-term and/or transient storage of domain names and email addresses. Regardless of the lifetime of the data, it should be stored in RFC-defined formats (preferred) or other formats which can transform between RFC-defined formats.

UASG Recommendations

- Applications and services should support the appropriate Unicode standards.
- Information should be stored in the UTF-8 (Unicode Transformation Format) whenever possible. Some systems may require support for UTF-16 as well, but generally UTF-8 is preferred. UTF-7 and UTF-32 should be avoided.
- Consider all end-to-end scenarios before converting A-Labels to U-Labels and vice versa when storing. It may be desirable to maintain only U-Labels in a file or database, because it simplifies searching and sorting. However, conversion may have implications when interoperating with older, non-Unicode-enabled applications and services. Consider storing in both formats.
- Clearly mark email addresses and domain names during storage for easier access. Instances where email addresses and domain names have been filed under the “author” field of a document or “contact info” in a log file have led to the loss of origin as an address.

PROCESS

Process occurs whenever an email address or domain name is used by an application or service to perform an activity (e.g., searching or sorting a list), or transformed into an alternate format (e.g., storing ASCII as Unicode). Additional validation may also occur during processing.

Domain names and email addresses can be processed in an unlimited number of ways*, which reinforces the need for conventions that ensure data is being understood and classified consistently.

UASG Recommendations

- Since the Unicode standard is continually expanding, code points not defined when the application or service was created should be checked to ensure they will not "break" the user experience. Missing fonts in the underlying operating system may result in non-displayable characters (frequently the " " character is used to represent these), but this situation should not result in a fatal crash.
- Use supported Unicode-enabled APIs.
- Use the latest Internationalized Domain Names in Applications (IDNA) Protocol [http://tools.ietf.org/html/rfc5891] and Tables [http://tools.ietf.org/html/rfc5892] documents for Internationalized Domain Names (IDNs).
- Process in UTF-8 format wherever possible.
- Ensure that the product or feature handles numbers as expected. For example, ASCII numerals and Asian ideographic number representations should be treated as numbers. [RFC5892, link above]
- Upgrade applications and servers/services together. If the server is Unicode and client is non-Unicode or vice versa, the data will need to be converted to each code page every time the data travels between server and client.
- Perform code reviews to avoid buffer overflow attacks. When doing character transformation, text strings may grow or shrink substantially.

Examples: Identify people in New Zealand by searching within the .nz ccTLD; identify pharmacists by searching for user@.pharmacist email addresses.

DISPLAY

Display occurs whenever an email address or a domain name is rendered within a user interface. Displaying domain names and email addresses is usually straightforward when the scripts used are supported in the underlying OS and strings are stored in Unicode; however, application-specific transformations may be required otherwise.

UASG Recommendations

- Display all Unicode code points which are supported by the underlying operating system. If an application maintains its own font sets, comprehensive Unicode support should be offered to the collection of fonts available from the operating system.
- When developing an app or a service, or when operating a registry, consider the languages supported and make sure OS and applications cover those languages.
- Convert non-Unicode data to Unicode before display. For example, the end user should see “everyone.みんな” as opposed to “everyone.xn--q9jyb4c”. (This conversion is an example of UA-ready processing).
- Display Unicode by default. Use Punycoded text to the user only when it provides a benefit. Augment Unicode display with Punycoded hover text as a mitigation.
- Consider that mixed-script addresses will become more common. Some Unicode characters may look the same to the human eye, but different to computers. Don’t assume that mixed-script strings are intended for malicious purposes, such as phishing, and if the user interface calls the strings to the user’s attention, be sure that it does so in a way which is not prejudicial to users of non-Latin scripts. Learn more about Unicode Security Considerations at: http://unicode.org/reports/tr36/.
- Use Unicode IDNA Compatibility Processing in order to match user expectations. To learn more, go to: http://unicode.org/reports/tr46/.
- Be aware of unassigned and disallowed characters. Learn more at RFC 5892: https://tools.ietf.org/rfc/rfc5892.txt.

Become Universal Acceptance-Ready

Source Code Reviews & Unit Testing

The process of inspecting the source code and verifying that only the correct programming techniques, software libraries and interfaces (AKA "APIs") have been used. After this has been completed, the administrator can verify the app or service works by testing it against specific capabilities (accept, validate, etc.) listed above. This method is typically only used by app developers and online service providers.

As part of UASG's awareness efforts, the group is reaching out directly to application developers and the largest online service providers to encourage them to perform universal acceptance source code reviews and testing and share a list of criteria which can be used to develop the test cases.

Manual Testing

Requires running multiple tests against new and non-ASCII domains, such as submitting an email address when registering for an online service and verifying that it has been accepted. Since there are a myriad number of potential online services to sign up for, as well as potential new email address combinations, this method requires trying different combinations of application, services, email address and/or domain names to provide a broad spectrum of use cases. This method can be performed by anyone, but is the most labor-intensive.

UASG is also helping promote this method by developing a list of top web sites, applications, email addresses and domain names suitable for testing.

Automated Testing

The use of automated scripts or directives to test a variety of URLs. This method requires more up-front technical work, but is more scalable to large measuring and monitoring efforts. A real-world example is the recent gTLD investigation performed by APNIC on behalf of ICANN: https://tinyurl.com/new-gtld-ua.

UASG is investigating methods of automated testing for universal acceptance and will share findings as they are available.