Universal Acceptance

EAI Technical Education and Awareness Directed at Developer Community Websites – Proposed FAQs

1 December 2022

Report UASG044A

TABLE OF CONTENTS

About ThinkTrans

ThinkTrans, a techno-linguistic company, works in the domain of natural languages and the allied technical and consultancy services. ThinkTrans has a pool of around 120 language experts from all the 22 scheduled languages of India. They work on projects from various national bodies, international organizations, private entities as well as industry associations. ThinkTrans specializes in consultancy and development services related to Internationalized Domain Names (IDNs) and a multilingual Internet.

Introduction

This is a report details the execution of the “Universal Acceptance Email Address Internationalization (EAI) Technical Education and Awareness to the Developer Community via Q&A Websites” project constituted under the UASG initiative. It captures analysis of developer forums, execution details, learnings captured, and subsequent actions required by the UA community to maximize UA outreach to developer communities.

Proposed FAQs

The FAQs are categorized into four broad categories:

1. Character/Length Specific
2. General Validation Specific
3. Programming Language Specific
4. General IDN/EAI Protocol Specific

The following are the actual questions per their respective categories.

1. Character/Length Specific
   a. What is the maximum length of a valid email address?
   b. What is the maximum length of a valid domain name?
   c. How to detect language/script of an A-label?
   d. Can I treat all the domain names (IDNs and ASCII-based) to be IDNs and process them as such without any ill-effects in my code flow?
   e. Are ASCII symbols prohibited altogether from IDNs?
   f. I see a special character in an IDN that I want to type out. How should I type it?
   g. What is the difference between Punycode and an A-label?

2. General Validation Specific
   a. What is the maximum length of a valid email address?
   b. What is the maximum length of a valid domain name?
   c. What is the simplest regular expression to validate emails to not accept them blindly?
   d. Are ASCII symbols prohibited altogether from IDNs?

3. Programming Language Specific
   a. What is the level of support for IDNs in Android in terms of UA-related requirements?
   b. What is the level of support for IDNs in iOS in terms of UA-related requirements?

4. General IDN/EAI Protocol Specific
   a. Are IDNs case-sensitive?
   b. Are internationalized email addresses case-sensitive?
   c. How to detect language/script of an A-label?
   d. Is there a way to avoid showing “xn--" for IDNs?
   e. How to compare A-label for visible similarity?
   f. Is the whole URL required to be converted to Punycode or only the domain name part?
   g. Can I treat all domain names (IDNs and ASCII-based) to be IDNs and process them as such without any ill-effects in my code flow?

h. While converting IDN to ASCII (or vice-versa) in any programming language, can the whole URL (with protocol parameters such as HTTPS, colons, etc.) be submitted as it is?
i. Are IDN conversion functions reversible?
j. Just like there is a list of valid TLDs, where can I find a list of valid email TLDs?

FAQs with Answers

Proposed answers to the questions identified have been provided, wherever applicable, below.

1. What is the simplest regular expression to validate emails to not accept them blindly?

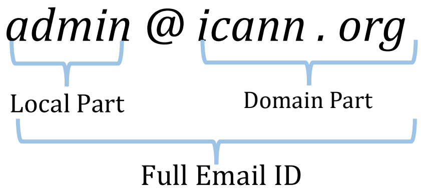
Proposed Answer:

When it comes to validating an email, there is no simple solution through the regular expression-based validation. Given the way email protocol is framed, technically speaking, the “regex” that will validate it fully will be huge and not feasible from the processing efficiency point of view. It is recommended to perform the validation by sending an email to the email ID to be validated with some personalized link or some code which the user has to click/input to verify himself/herself to be the rightful owner of the submitted email ID.

2. What is the maximum length of a valid email address?

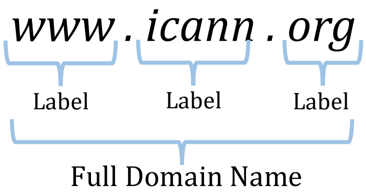
There is a good amount of confusion about the valid length of an email address. It stems from the fact that RFC 3696 states the limit to be 320, whereas in the subsequent errata it is clarified by the author of the Standards Track RFC 2821 that the limit should be considered to be 254. Those who are interested in a detailed discourse around it can go through the links.

To sum up, the maximum length of an email local part is 64 (although it is frequently disobeyed by some implementations), while the maximum permissible length of the full email ID is 254. The length of the email local part is intrinsic to this limit.



3. What is the maximum length of a valid domain name?

As spelled out in RFC 1035, the maximum length of a full domain name is 255 octets including the label separator, i.e., “.”. Each label can have a maximum length of 63.



4. **Are IDNs case-sensitive?**

Proposed Answer:
IDNs as a part of their processing from U-label to A-label, undergo a stringprep operation, which includes "case-folding", i.e., converting all the characters to the lowercase. So, as per RFC 5895, they are case-sensitive. However, since user-facing applications are expected to do the case-mapping from the end user perspective, they could be considered intrinsically case-insensitive, or IDN processing includes "case-folding" before the Punycode step, i.e., converting all the characters to lowercase. So, intrinsically, IDNs are case-insensitive.

5. **Are internationalized email addresses case-sensitive?**

Proposed Answer:
An email ID comprises of two parts governed by two different sets of protocols. The following is the composition of an email ID:
<email local part>@<domain name part>

The <email local part> is governed by the mail administrator of the given email service and the email protocol does not explicitly bind the implementations to keep that in the case-insensitive form. Hence, though unwise, one should expect <email local part> to be case-sensitive. For more elaborate discussion and common guidelines released by the Universal Acceptance Steering Group (UASG), refer to UASG 028.

The <domain name part> as a part of conversion of IDN to Punycode, undergo a stringprep operation which includes "case-folding", i.e., converting all the characters to the lowercase. So, intrinsically, IDNs are case-insensitive.

6. **How to detect language/script of an A-label?**

Proposed Answer:
A-label is a language/script agnostic string and should be considered as such. However, if one wants to understand the language/script of the domain name that A-label represents, then conversion of the said A-label to its Unicode equivalent is a must. Once converted to Unicode, depending on the business requirement, appropriate deductions can be made as to the script/language of the given IDN label. For those interested in that process, refer to this link on the Unicode FAQs page.

7. **Is there a way to avoid showing "xn--" for IDNs?**

Proposed Answer:
The "xn--" representation of an IDN pertains indicates that the domain name is in its A-label format. To get rid of the same, depending on the programming language you are coding into, use the appropriate library to convert from "A-label to Unicode". For the recent studies on the subject, follow the document UASG-018A for some of the common programming languages like: C, C#, Go, Java, JavaScript, Python, and Rust. In addition, for additional platforms like iOS Swift, PHP, and Android Kotlin, follow the document UASG 037.

8. **How to compare A-label for visible similarity?**

Proposed Answer:
A-labels being a result of a mathematical algorithm, do not have direct bearing on the visual shape of the Unicode string they represent. For visual similarity comparison, one needs to convert the A-label to their equivalent Unicode (U-label) strings and rely either on the preferred string similarity assessment algorithms or resort to manual inspection.

Depending on the programming language you are coding into, use the appropriate library to convert from "Punycode to Unicode." For recent studies on the subject, follow the document

UASG-018A for some of the common programming languages like: C, C#, Go, Java, JavaScript, Python, and Rust. In addition, for additional platforms like iOS Swift, PHP, and Android Kotlin follow the document UASG 037.

9. Is the whole URL required to be converted to Punycode or only the domain name part?

Proposed Answer:
Technically speaking, only the domain part.

https://www.**थिंकट्रान्स.भारत**/index.php

Only the “थिंकट्रान्स.भारत” part in the above URL needs to be subjected to the Punycode conversion routine. The rest of the parts (protocol and file-system elements) should not be subjected to Punycode conversion. It can create broken and inaccessible links.

10. Can I treat all the domain names (IDNs and ASCII-based) to be IDNs and process them as such without any ill-effects in my code flow?

Proposed Answer:
Given the way A-label to Unicode conversion algorithm is framed, subjecting non-IDN labels to it do not necessarily undergo any undesirable change. It would be safe to say that one can process them without any ill-effects.

11. While converting IDN to ASCII (or vice-versa) in any programming language, can the whole URL (with protocol parameters such as HTTPS, colons, etc.) be submitted as is?

Proposed Answer:
Technically speaking, only the domain part.

https://www.**थिंकट्रान्स.भारत**/index.php

Only the “थिंकट्रान्स.भारत” part in the above URL needs to be subjected to the Punycode conversion routine. The rest of the parts (protocol and file-system elements) should not be subjected to Punycode conversion.

12. Are IDN conversion functions reversible?

Proposed Answer:
RFC 5891 recommends a set of operations on the IDN label (U-label to A-label) which are not per-se fully reversible, case in point being normalization and the case-folding routines. There could be labels which do not change even after undergoing these routines as they might already be normalized and in lower-case forms. However, as a whole, one can say that these functions may not always yield fully reversible strings.

13. Are ASCII symbols prohibited altogether from IDNs?

Proposed Answer:
Internationalized Domain Names are defined as those domain names which by definition have “at least one character from the non-ASCII space.” This is to say that it has at least one character which is not in the Unicode character range of 1-128. So as long as “at least one character” clause is satisfied, the rest of the characters, technically, can be from the ASCII space. ASCII symbols can be part of IDNs.

14. Just like there is a list of valid TLDs, where can I find a list of valid email TLDs?

Proposed Answer:

Typically speaking, such a list would be a list of domain names that have an “mx” entry as a part of their DNS record. However, neither presence of an mx record ensures a full implementation of an email server, nor absence of one indicates otherwise given the way DNS redirection mechanisms could work in tandem to redirect user queries. Having said this, mail-servers often are subjected to attacks through various spamming agents for the bombardment of advertisements/offers/phishing-spoofing attacks. Officially maintaining such a list would only amount to further abuse of the process and is not advisable.

15. What is the level of support for IDNs in Android in terms of UA-related requirements?

There was a systematic study constituted by the UASG studying around 22 libraries across four of the major operating systems vis-à-vis support titled “UA-Readiness Evaluation of Programming Languages and Development Frameworks.” Eight critical libraries from Android-Kotlin were evaluated namely okHttp, HttpUrlConnection, Retrofit, Fuel, Volley, HttpUrlConnection, Apache HttpClient, Jakarta Mail, and Email Intent. The support levels at the time of the study were found to be poor. The detailed study can be found in presentation form as well as document form at the respective links.

16. What is the level of support for IDNs in iOS in terms of UA-related requirements?

There was a systematic study constituted by the UASG studying around 22 libraries across four of the major operating systems vis-à-vis support titled “UA-Readiness Evaluation of Programming Languages and Development Frameworks.” Three critical libraries from iOS-Swift were evaluated namely MessageUI, Alamofire, and URLSession. The support levels at the time of the study were found to be poor. The detailed study can be found in presentation form as well as document form at the respective links.

17. I see a special character in an IDN that I want to type out. How should I type it?

Typically, all the operating systems have pre-defined set of language/script keyboards which enable users to input content in the respective languages. If the character you are looking for forms part of everyday usage of the language/script it belongs to, you can input the same using the respective language/script keyboard.

You can find the procedures of adding the language/script keyboard in some of the popular operating systems at links given. For Microsoft Windows, use this. For macOS, use this. For Android, use this. For iOS use this and for Ubuntu Linux use this.

In case you want to input the specific character without using the language/script inputting tools, there is some software that can be used that provide neat interface for searching and inputting the character. One such application is BabelPad for Windows platform. For those who are looking for a platform independent solution, one can make use of BabelMap, which is an online application and gives access to the entire Unicode repertoire.

18. Where can I find the compliance levels of various programming language libraries vis-à-vis UA compliance?

You can find the same at:

- UASG 018 Reviewing Programming Languages and Frameworks for Compliance with Universal Acceptance Good Practice EN
- UASG 031 FAQs: UA Readiness of Programming Languages and Email Tools EN
- UASG 037 UA-Readiness of Some Programming Language Libraries and Frameworks EN

19. What is the difference between Punycode and an A-label?

Punycode is the name of the algorithm that converts U-labels to A-labels. A domain name such as “थिंकट्रान्स.भारत” consists of the two U-labels थिंकट्रान्स and भारत, and Punycode converts the U-label थिंकट्रान्स to the A-label xn--i1b1b5avs1c3c4bf6ld, which is finally used in the DNS. A-labels are commonly referred to as Punycode, but this is technically incorrect.