Universal Acceptance

Основные сведения об универсальном принятии (UA)

Группа управления по универсальному принятию (UASG)

23.09.2019

СОДЕРЖАНИЕ

Описание документа

Технологии интернета, в том числе относящиеся к присвоению имен, постоянно развиваются и меняются. За последние годы с разрешения Интернет-корпорации по присвоению имен и номеров (ICANN) созданы новые домены верхнего уровня (TLD). Некоторые из них состоят из традиционных символов ASCII, а некоторые (интернационализированные доменные имена) из символов, не входящих в набор ASCII. Например, .nyc, .संगठन, .eco и .католик. Однако многие приложения и службы не были обновлены для правильного обращения с этим расширенным спектром TLD. Кроме того, теперь интернет-стандарты электронной почты допускают использование в адресах электронной почты символов не из набора ASCII. Поэтому, пока программное обеспечение не будет обновлено, оно не будет правильно обрабатывать эти домены и адреса. Это влияет на пользовательский опыт несколькими способами:

- Допустимые адреса электронной почты не распознаются и не принимаются.
- Доменные имена в адресной строке браузера ошибочно считаются поисковыми запросами.

Только после того как программное обеспечение начнет распознавать и обрабатывать все доменные имена и адреса электронной почты, — это состояние называется универсальным принятием (UA), — удастся обеспечить согласованный и позитивный опыт для всех интернет-пользователей. В этом документе представлены общие сведения об универсальном принятии и усилиях, которые предпринимаются для содействия разработке программного обеспечения, отвечающего критериям универсального принятия.

Целевая аудитория

Настоящий документ призван ознакомить с универсальным принятием техническую аудиторию (разработчиков, администраторов и операторов), которая возможно осведомлена о некоторых аспектах интернет-технологий, но не всегда знает, как новые доменные имена, IDN-домены и адреса электронной почты меняют надлежащий подход к их принятию, проверке, хранению, обработке и отображению. Он представляет собой отправную точку для изучения универсального принятия людьми с самыми разными техническими знаниями.

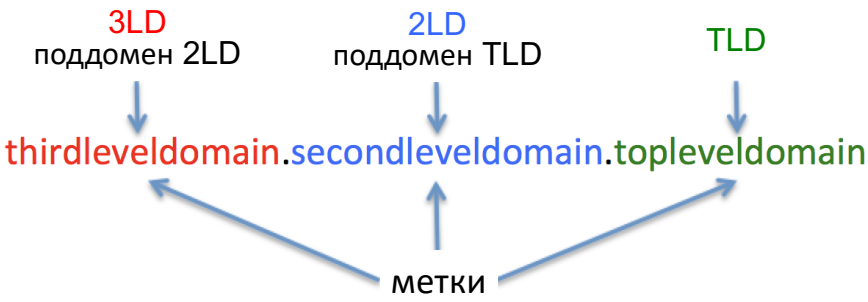
Основные понятия

Доменные имена

Доменное имя — это удобный для человека идентификатор компьютеров и сетей в интернете. Обычно он представлен в виде последовательности текстовых меток, разделенных «точками» (точка — знак препинания); например, www.example.tld. Каждая метка представляет один из уровней в иерархии системы доменных имен (DNS).

Высший уровень или «корень» иерархии представлен метками доменов верхнего уровня (TLD), такими как com, jp и বাংলা , которые находятся в конце доменного имени. Поскольку они находятся в конце, TLD иногда называют «суффиксами».

Если перейти от корня ниже по иерархии DNS, следующая метка идентифицирует поддомен TLD, обычно называемый доменом второго уровня. Следующая метка идентифицирует поддомен домена второго уровня, который обычно называется доменом третьего уровня, и так далее. Каждая метка отделена от соседней точкой. Например, доменное имя с тремя уровнями может выглядеть так:



Национальные домены верхнего уровня (ccTLD)

Некоторые TLD делегируются конкретным странам или территориям. Они называются национальными доменами верхнего уровня (ccTLD). В прошлом все ccTLD состояли из двух букв, соответствующих двухбуквенному коду, выделенному стране или территории Международной организацией по стандартизации в стандарте ISO 3166. После 2010 года также появились интернационализированные ccTLD, которые представляют название страны или территории на языке этой страны или территории.

Домены общего пользования верхнего уровня (gTLD)

Большинство TLD, не относящихся к категории ccTLD, называются доменами общего пользования верхнего уровня (gTLD), которые доступны всем либо только членам определенного сообщества. К ним относятся давно известные .com, .net и .org, а также более свежие дополнения.

Благодаря Программе New gTLD — инициативе, которую координирует ICANN, — система доменных имен (DNS) испытала экспоненциальный рост за счет создания новых доменов общего пользования верхнего уровня. Эти новые gTLD могут представлять бренды, сообщества лиц с общими интересами, географические районы (города, регионы) и многое другое.

Интернационализация доменных имен

В доменных именах изначально разрешалось использовать только подмножество символов ASCII (буквы a–z, цифры 0–9 и дефис «-»). Со времени регистрации первого домена в зоне .com, symbolics.com в 1985 году, количество и характеристики доменных имен расширились для удовлетворения потребностей, явившихся результатом постоянного расширения глобального использования интернета как общественного ресурса. Сегодня для большинства интернет-пользователей английский язык не является родным, однако именно он — основной язык интернета. Чтобы способствовать интернационализации интернета, в 2003 году Инженерная проектная группа интернета (IETF) начала выпускать стандарты с техническими принципами внедрения интернационализированных доменных имен (IDN-доменов) через механизм преобразования, позволяющий представить доменные имена без ASCII на любом алфавите, который поддерживается в Unicode (*напр.*, 普遍接受-测试.世界, ua-test.كاثوليك и т. д.).

Правление ICANN одобрило процесс создания новых IDN ccTLD в октябре 2009 года, и первые IDN ccTLD были добавлены в корневую зону в мае 2010 года. В июне 2011 года Правление утвердило и санкционировало запуск Программы New gTLD, которая охватывала новые TLD в кодировке ASCII, а также IDN TLD. Первая партия TLD в рамках этой программы была добавлена в корневую зону в 2013 году.

Необходимость универсального принятия (UA)

Спустя десятилетие после того, как IETF выпустила свои руководящие принципы для IDN-доменов, и благодаря Программе New gTLD ICANN сейчас работает более 1000 новых TLD. Однако некоторые программы и приложения остаются устаревшими и не способны обрабатывать эти новые TLD. Это создает проблемы для интернет-пользователей, в том числе для тех, кто использует символы и алфавиты, не входящие в набор ASCII.
Универсальное принятие обеспечивает правильное и единообразное принятие, проверку, хранение, обработку и отображение любых корректных доменных имен и адресов электронной почты всеми интернет-ориентированными приложениями, устройствами и системами. При этом, например, разрешение любого допустимого веб-адреса обеспечивает доступ к ожидаемому ресурсу на целевом сайте, а использование любого допустимого адреса электронной почты приводит к доставке почты ожидаемому получателю.

Группа управления по универсальному принятию (UASG) — это группа, созданная по инициативе интернет-сообщества и при поддержке со стороны ICANN для выполнения работы, которая будет реально способствовать универсальному принятию и поможет обеспечить единый подход и удобство для интернет-пользователей в мировом масштабе.

A-метки и U-метки

Доменные имена, где используются символы не из набора ASCII, называют интернационализированными доменными именами (IDN-доменами). Интернационализированная часть доменного имени может присутствовать в любой метке, не только в TLD.

Поскольку в самой DNS ранее использовался только ASCII, пришлось создать дополнительную кодировку, чтобы представить кодовые точки Unicode, не входящие в

набор ASCII, в виде строк ASCII. Алгоритм преобразования Unicode в ASCII называется Punycode**, а** получающиеся в результате строки называются A-метками. A-метку можно отличить от обычной метки ASCII, поскольку она всегда начинается со следующих четырех символов:

xn--

Эти символы называются префиксом ACE.[1]

Punycode — обратимое преобразование: можно преобразовать строку Unicode в метку A-label и снова преобразовать метку A-label в строку символов Unicode (которая называется U-меткой).

Алгоритм Punycode, как правило, используется только для представления интернационализированных доменов. Хотя гипотетически можно кодировать другие строки UTF-8, используя Punycode, это нестандартный подход, не обеспечивающий взаимодействие с другими системами.

Примеры (вымышленных) IDN-доменов

Вариант U-метки	**Вариант A-метки**
example.みんな	example.xn--q9jyb4c
大坂.info	xn--uesx7b.info
みんな.大坂	xn--q9jyb4c.xn--uesx7b

Интернационализация адреса электронной почты (EAI)

Адрес электронной почты состоит из двух частей:

- Локальная часть (перед символом «@»).
- Доменная часть (после символа «@»).

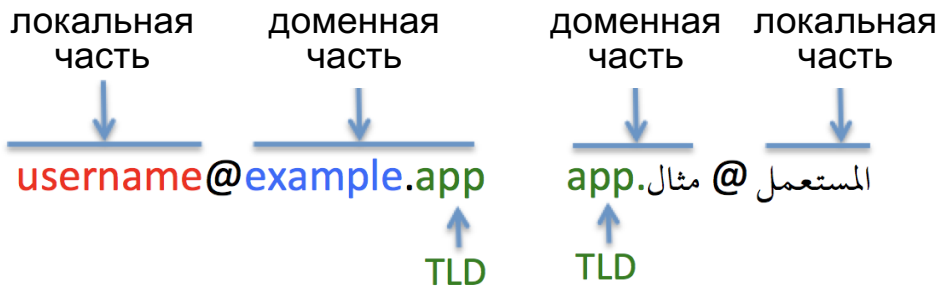
Поскольку в адресах электронной почты и доменных именах могут использоваться языки как с правосторонним (LTR), так и с левосторонним (RTL) письмом, слова «до» и «после» следует понимать в контексте направления письма.

Примеры (вымышленных) адресов EAI

user@example.みんな	Используется IDN TLD
user@大坂.info	Используется IDN-домен второго уровня
用戶@example.lawyer	Используется локальная часть Unicode и новый gTLD

[1]Префикс ASCII-совместимого кодирования (ACE), который отличает метки в кодировке Punycode от остальных меток ASCII.

Пример левостороннего текста в адресе EAI



В интернационализированном адресе электронной почты доменная часть может содержать любое доменное имя, в том числе с новыми TLD, а также U-метки Unicode. Локальная часть не является доменным именем и в принципе может содержать практически любой символ Unicode, хотя на практике почтовые системы ограничивают набор символов, используемых в именах почтовых ящиков.

Термин «интернационализация адреса электронной почты (EAI)» часто используется, чтобы описать применение интернационализированных адресов в электронной почте.

Динамическое создание ссылок (Linkification)

Современное программное обеспечение, такое как популярные текстовые редакторы или приложения для работы с электронными таблицами, иногда позволяет пользователю создать гиперссылку, просто введя строку, которая выглядит как веб-адрес, адрес электронной почты или сетевой путь. Например, ввод «www.icann.org» в сообщение электронной почты может привести к автоматическому созданию активной ссылки на сайт http://www.icann.org, если приложение распознает «www.» как специальный префикс или «.org» как специальный суффикс.

При этом механизм создания ссылок должен слаженно работать для всех правильно сформированных веб-адресов, адресов электронной почты или сетевых путей, а не только для некоторых. Точная система создания ссылок очень сложна и зависит от контекста (например, на некоторых языках «www» не обозначает веб-адрес), поэтому здесь она не рассматривается.

Динамический характер регистратуры корневой зоны

DNS — крупная распределенная база данных, которая разделена на несколько сегментов, называемых зонами. Сегмент, содержащий все TLD, называется корневой зоной, поскольку концептуально он находится в корне дерева имен DNS. Все зоны DNS, включая корневую, обновляются по мере необходимости. По мере добавления новых TLD или прекращения работы старых TLD их имена добавляются в корневую зону или удаляются из нее.

Это означает, что любой фиксированный список TLD, например, список, хранящийся в приложении или файле, в конечном итоге неизбежно устареет. Чтобы надежно проверить TLD в доменном имени, программное обеспечение может выполнить онлайн-проверку с помощью DNS-запроса или, если используется файл, регулярно обновлять файл. Оба способа позже будут рассмотрены подробнее.

Универсальное принятие на практике

Пять критериев универсального принятия

Универсальное принятие — состояние, когда все интернет-ориентированные приложения, устройства и системы правильно и единообразно принимают, проверяют, хранят, обрабатывают и отображают все корректные доменные имена и адреса электронной почты. Эти пять критериев универсального принятия поясняются ниже.

1. Принятие[2]	***Принятие* — это процесс получения адреса электронной почты или доменного имени в виде строки символов из пользовательского интерфейса, файла или API, используемого приложением или интернет-сервисом.** Приложения и сервисы должны принимать следующие доменные имена и адреса электронной почты: ▪ введенные через пользовательские интерфейсы или ▪ полученные из других приложений и сервисов через API.
2. Проверка[3]	***Проверка* может выполняться во многих местах при получении или создании адреса электронной почты или доменного имени в виде строки символов приложением или интернет-сервисом.** Цель проверки — убедиться, что введенная информация является допустимой или, по крайней мере, определенно не является недопустимой. Проверка гарантирует синтаксическую правильность информации. Могут проводиться и другие проверки. Что касается доменных имен и адресов электронной почты, многие программисты традиционно опирались на *специальные методы проверки*, такие как проверка того, что длина TLD находится в допустимых пределах, или что используются символы из набора ASCII. Однако эти методы основаны на предположениях, которые больше не применимы, потому что интернет меняется: ▪ Доменные имена и адреса электронной почты теперь могут содержать символы Unicode не из набора ASCII. ▪ Список TLD меняется. ▪ Любая метка в доменном имени, включая метку TLD, может содержать до 63 символов.[4] Сохраняется возможно проверять TLD с помощью других методов, как описано ниже.

[2] В этом документе принятие и проверка разграничены. На практике эти действия могут пересекаться.
[3] В этом документе принятие и обработка отделены от проверки. На практике эти действия могут пересекаться.
[4] Предел длины в 63 символа применяется к самой метке, если это метка ASCII, или к A-метке, если это IDN-домен.

3. Хранение	***Хранение* — этап, когда адрес электронной почты или доменное имя хранится в виде строки символов в базе данных или файле, используемом приложением или интернет-сервисом, а затем извлекается тем же самым или другим приложением.** Приложениям и сервисам может требоваться долговременное и/или кратковременное хранение доменных имен и адресов электронной почты. Независимо от периода существования данных, они должны храниться: ▪ в форматах, соответствующих интернет-стандартам Запрос комментариев (RFC), или (что менее желательно) ▪ в других форматах, позволяющих выполнить преобразование в форматы, соответствующие RFC. Хотя Unicode в именах DNS и адресах электронной почты хранится как UTF-8, в устаревшем коде могут встречаться другие форматы. См. раздел «Передовой опыт» ниже.
4. Обработка[5]	***Обработка* выполняется, когда адрес электронной почты или доменное имя используется приложением или сервисом для выполнения действия (например, поиска или сортировки списка) или преобразуется в альтернативный формат (например, при преобразовании U-меток в A-метки).** Во время обработки может осуществляться дополнительная проверка. Способы обработки адресов электронной почты и доменных имен ограничиваются только фантазией разработчиков приложений, но важно не делать предположений (например, что электронное письмо на имя pākehā@tetaurawhiri.govt.nz адресовано получателю в Новой Зеландии), которые зависят от политики, выходящей за рамки DNS.
5. Отображение	***Отображение* происходит, когда адрес электронной почты или доменное имя обрабатывается в пользовательском интерфейсе.** При отображении доменных имен и адресов электронной почты обычно, но не всегда, не возникает затруднений, если используемые в имени или адресе алфавиты поддерживаются операционной системой, а строки хранятся в Unicode.[6] Если

[5] В этом документе обработка и проверка разграничены. На практике эти действия могут пересекаться.

[6] Важно признать, что отображение не является простой задачей, даже если эти условия выполняются для некоторых сложных сценариев.

	эти условия не выполнены, могут потребоваться операции преобразования, которые зависят от конкретного приложения. Кроме того, даже если строки поддерживаются операционной системой, при отображении могут возникнуть трудности, например, когда смешаны языки RTL и LTR или неясно общее направление текста.

Пользовательские сценарии

Приведенные выше примеры и определения могут создать впечатление, что универсальное принятие касается только компьютерных систем и интернет-сервисов. Однако реальность такова, что речь идет также о людях, использующих эти системы и сервисы.

Ниже приведены примеры действий, требующих универсального принятия:

Регистрация нового TLD	Организация создает «фирменный» TLD, чтобы предложить своим клиентам дифференцированный подход к обслуживанию, предоставив им адреса электронной почты в формате имя_клиента@example.brand. При этом универсальное принятие означает следующее: ▪ Сайты и приложения принимают такие адреса электронной почты «@example.brand» так же, как и в случае старых TLD, таких как .com, .net, .org.
Доступ к gTLD	Пользователь получает доступ к сайту, доменное имя которого содержит новый TLD, введя адрес в браузере или щелкнув ссылку в документе. При этом универсальное принятие означает следующее: ▪ Хотя TLD является новым, браузер пользователя отображает веб-адрес в исходной форме и получает доступ к сайту, как этого ожидает пользователь. Браузер не отображает доменные имена в виде A-меток, если это не приносит какой-либо выгоды пользователю.

Использование адреса электронной почты, содержащего новый gTLD, для онлайн-идентичности	Пользователь получает адрес электронной почты, в доменной части которого есть новый gTLD, и использует этот адрес электронной почты как свой идентификатор для доступа к банковским счетам и учетным записям в программах лояльности авиакомпаний. При этом универсальное принятие означает следующее: ▪ Хотя в адресе электронной почты используется новый домен, сайт банка или авиакомпании принимает адрес так же, как адрес в старом TLD, например .biz или .eu.
Доступ к IDN-домену	Пользователь получает доступ к URL IDN-домена, введя этот URL в браузере или щелкнув ссылку в документе. При этом универсальное принятие означает следующее: ▪ Даже если доменное имя содержит символы, не совпадающие с языковыми настройками на компьютере пользователя, любой браузер пользователя правильно отображает веб-адрес и успешно получает доступ к сайту.
Отправка электронного письма на интернационализированный адрес электронной почты	Пользователь получил новый адрес электронной почты, в доменном имени которого есть символы, не входящие в набор ASCII (*например,* info@普遍接受-测试. 世界). При этом универсальное принятие означает следующее: ▪ Пользователь может отправлять сообщения на любой адрес электронной почты и получать сообщения с любого адреса электронной почты через любой почтовый клиент.
Использование интернационализированного адреса электронной почты для онлайн-идентичности	Пользователь получает адрес электронной почты с символами, не входящими в набор ASCII, и использует его как свой идентификатор для доступа к банковским счетам и учетным записям в программах лояльности авиакомпаний. При этом универсальное принятие означает следующее: ▪ Сайт банка или авиакомпании принимает новый идентификатор точно так же, как если бы это был любой другой адрес электронной почты.

Динамическое создание гиперссылки в приложении	Пользователь вводит веб-адрес в документ или сообщение электронной почты. При этом универсальное принятие означает следующее: ▪ Правила, используемые приложением для автоматического создания гиперссылки, не меняются, если в адресе есть символы не из набора ASCII или новый TLD.
Разработка приложения	Разработчик пишет приложение, которое обращается к веб-ресурсам. При этом универсальное принятие означает следующее: ▪ Используемые разработчиками инструменты включают библиотеки, которые обеспечивают универсальное принятие, поддерживая новые TLD и IDN-домены.

Отклонение от универсальной практики

Нижеперечисленное считается плохой практикой:

✖	Отображение A-меток через пользовательский интерфейс без соответствующей выгоды для пользователя, такой как демонстрация соответствия U-метки и A-метки.
✖	Требование к пользователю вводить A-метки при регистрации нового адреса электронной почты или нового доменного имени.
✖	Проверка синтаксиса доменного имени или адреса электронной почты с использованием устаревших критериев или неавторитетных интернет-ресурсов о доменных именах.
✖	Использование устаревшего списка TLD, несмотря на регулярное добавление и удаление новых TLD.
✖	Раскрытие внутреннего использования A-меток пользователям. Например, преобразование доменов в адресах EAI в A-метки при ответе пользователю EAI.
✖	Обработка некоторых доменных имен как поисковых запросов, поскольку приложение не распознает их как доменные имена.
✖	Настройка спам-фильтров на автоматическую блокировку всех (новых) TLD в отсутствие доказательств злоупотребления.

Технические условия готовности к UA

Чтобы приложение или сайт были готовы к UA, они должны удовлетворять различным требованиям.

Требования высокого уровня

Обеспечивающее универсальное принятие (UA) приложение или сервис:

1. Поддерживает все доменные имена независимо от длины или набора символов.
См. RFC 5892.

2. Позволяет использовать для доменных имен и адресов электронной почты все допустимые наборы символов.

Принимает кодовые точки Unicode, а также ASCII.

3. Может правильно отображать все кодовые точки в строках Unicode.

См. RFC 3490. Обратите внимание, что в Unicode регулярно добавляются новые кодовые точки, так что это меняющаяся цель.

4. Может правильно отображать строки с направлением письма справа налево (RTL), например, на арабском и иврите.

Сведения об алфавитах RTL см. в документе RFC 5893.

5. Может передавать данные между приложениями и сервисами в формате UTF-8 и, при необходимости, в других кодировках, которые могут быть преобразованы в UTF-8 и обратно.

Сведения о UTF-8 см. в документе RFC 3629.

6. Предлагает публичные и закрытые API, которые поддерживают UTF-8.

Закрытые API применяются только при обмене данными между службами одного и того же поставщика.

7. Правильно обрабатывает адреса EAI.

В частности, не преобразует IDN-домены в адресах в A-метки.

8. Может отправлять и получать электронную почту, независимо от имени домена или набора символов.

См. RFC 6530.

9. Хранит пользовательские данные в форматах, которые поддерживают Unicode и могут быть преобразованы в UTF-8 и обратно.

Такие преобразования будут видны только оператору продукта или сервиса.

10. Поддерживает все доменные имена верхнего уровня, включенные в официальный список TLD ICANN, независимо от их длины или набора символов.

Официальный список см. здесь: https://data.iana.org/TLD/.

Важные для разработчика аспекты

Поскольку многие существующие программные системы содержат жестко запрограммированные предположения о доменах и адресах электронной почты, может возникнуть необходимость изменения кода для распознавания IDN-доменов, новых TLD и почтовых адресов EAI. В этом разделе рассматривается, какие изменения кода позволят разработчикам обеспечить универсальное принятие.

Разработка совместимого и гибкого программного обеспечения

Общим принципом разработки программного обеспечения является принцип надежности, сформулированный Джоном Постелом в RFC 793:

«Будьте консервативны в том, что делаете, и будьте либеральны в том, что принимаете от других».

То есть будьте консервативны в том, что отправляете: в любой области, где спецификация может быть неоднозначной или неясной, избегайте всего, что может удивить других. С другой стороны, при получении принимайте все, что является предположительно допустимым. Это *не* означает изменение кода для обхода очевидных ошибок в других реализациях, так как результатом подобной практики станет недокументированный и «не поддающийся отладке» беспорядок.

Передовой опыт разработки и обновления программного обеспечения для достижения готовности к UA

Принятие	
✓	Отображать имена в кодировке Unicode всегда, когда это возможно. Пользователям следует разрешать, но не требовать от них, вводить доменные имена как A-метки вместо U-меток. Однако для отображения по умолчанию следует использовать U-метки, а A-метки показывать пользователю только тогда, когда это приносит ему выгоду.
!	**Не** генерировать адреса EAI с A-метками, но **обрабатывать** их при получении из стороннего программного обеспечения.
✓	Любой компонент пользовательского интерфейса, требующий от пользователя ввести доменное имя или адрес электронной почты, должен принимать длинные имена. Доменные имена в кодировке ASCII могут

	содержать до 63 символов в каждой метке и иметь общий размер до 253 байтов. Метки UTF-8 могут быть намного длиннее меток ASCII, а их общая длина может составлять до 670 байтов. Помните, что код UTF-8 для большинства кодовых точек Unicode больше одного байта. ▪ См. RFC 1035.

Проверка	
✓	Проверять только по мере целесообразности. **Проверять только в том случае, если это необходимо для работы приложения или сервиса.** Это самый надежный способ обеспечить принятие системами всех допустимых доменных имен.
✓	Признавать, что синтаксически правильные входные данные могут представлять доменные имена или адреса электронной почты, которые в настоящее время используются в интернете. Они могут быть действительными или недействительными, в зависимости от приложения.
!	При проверке учитывайте следующее: ▪ Убедитесь, что та часть доменного имени, которая представляет TLD, есть в официальной таблице. IANA публикует список доменов верхнего уровня здесь: * https://data.iana.org/TLD/tlds-alpha-by-domain.txt * См. также: https://www.icann.org/en/system/files/files/sac-070-en.pdf ▪ Запросите данные по доменному имени в DNS. * GETDNS API (http://getdnsapi.net/) — платформонезависимый способ отправки DNS-запросов. * Большинство операционных систем также имеют встроенный API DNS-запросов. ▪ Для обнаружения опечаток требуйте повторного ввода адреса электронной почты. ▪ Проверьте символы в метках, убедившись, что каждая метка соответствует правилам интернационализации доменных имен в приложениях (IDNA 2008). * См. RFC 5892 ▪ При проверке самих меток ограничьтесь небольшим количеством правил для метки целиком, которые определены в RFC. * См. RFC 5894 ▪ Убедитесь, что продукт или функция правильно обрабатывает числа. * Например: Символы арабских цифр следует обрабатывать как числа в полях числового ввода, а также как цифры ASCII.

	* Обратите внимание, что арабские цифры допустимы в U-метках, но не считаются эквивалентными цифрам ASCII в этом контексте.

Хранение	
✓	Приложения и сервисы должны поддерживать последнюю версию стандартов Unicode.
✓	По мере возможности информацию следует хранить в формате UTF-8. Кроме того, некоторым системам может потребоваться поддержка старого формата UTF-16, но в большинстве случаев формат UTF-8 предпочтительнее. UTF-7 устарел, а UTF-32 слишком громоздкий для хранения файлов. При необходимости строки следует нормализовать (иногда нормализация может привести к потере информации).
!	Примите во внимание все комплексные сценарии перед преобразованием A-меток в U-метки и наоборот во время хранения. В новых приложениях целесообразнее хранить в файле или базе данных только U-метки, поскольку это упрощает поиск, сортировку и отображение. Однако преобразование может повлиять на взаимодействие с более старыми приложениями и сервисами, не поддерживающими Unicode.
✓	Пометьте адреса электронной почты и доменные имена как таковые в хранилище, чтобы упростить доступ. Хранение адресов электронной почты и доменных имен в полях «Автор» документа или «Контактная информация» в журнале событий приводило к утрате исходного адреса.
✓	Независимо от способа хранения адресов и доменных имен, вы должны иметь возможность сопоставлять строки в нескольких форматах. Например, при поиске example.みんな следует также найти example.xn--q9jyb4c.

Процесс	
✓	Убедитесь, что для всех ответов веб-сервера и почты MIME указан тип контента UTF-8.
✓	Укажите UTF-8 в заголовке HTTP веб-сервера. ▪ Важно обеспечить, чтобы кодировка указывалась в каждом ответе.

!	Учитывайте ситуацию перед преобразованием A-меток в U-метки и наоборот во время обработки. Целесообразно хранить в файле или базе данных только U-метки, поскольку это упрощает поиск и сортировку. Однако преобразование может повлиять на взаимодействие с более старыми приложениями и сервисами, не поддерживающими Unicode.
✓	Убедитесь, что продукт или функция обрабатывает операции сортировки, поиска и сравнения в соответствии с языковыми спецификациями, а также обеспечивает возможность многоязычного поиска и сортировки.
✗	Не используйте процентную кодировку для меток в доменных именах: ▪ example.みんな — правильно, ▪ example.%E3%81%BF%E3%82%93%E3%81%AA — неправильно.
✓	Поскольку стандарт Unicode постоянно расширяется, следует проверить те кодовые точки, которые не были определены в момент создания приложения или сервиса, и убедиться в отсутствии ошибочных или противоречивых выходных данных. Результатом отсутствия шрифтов в базовой операционной системе могут стать неотображаемые символы (часто для их представления используется небольшой прямоугольник), но такая ситуация не должна приводить к сообщению о сбое или ошибке.
✓	Используйте поддерживаемые API, совместимые с Unicode.
✓	Используйте последнюю редакцию документов по протоколам и таблицам интернационализированных доменных имен в приложениях (IDNA 2008) для IDN-доменов: ▪ RFC 5891 ▪ RFC 5892
✓	По возможности выполняйте обработку текста в формате UTF-8.
✓	Координируйте обновления приложений и служб, от которых они зависят. Если сервер использует Unicode, а клиентская часть нет, или наоборот, придется выполнять преобразование при каждой операции передачи данных, что способствует возникновению ошибок и может замедлить работу.
✓	При преобразовании символов текстовые строки могут существенно удлиняться или сокращаться. Каждая кодовая точка UTF-8 может быть от 1 до 4 байтов длиной, и в некоторых случаях один символ в другой кодировке может соответствовать нескольким кодовым точкам UTF-8 или наоборот.

Отображение	
✓	Отображайте все кодовые точки Unicode, которые поддерживаются базовой операционной системой. Все современные операционные системы поддерживают Unicode, но их механизмы визуализации не всегда подходят для всех сценариев и языков. Обеспечивайте визуализацию символов в приложениях только в том случае, если правильная визуализация невозможна в целевой операционной системе.
✓	При разработке приложения или сервиса обращайте внимание на поддерживаемые языки и обеспечивайте охват этих языков операционными системами и приложениями.
✓	Преобразуйте A-метки в U-метки перед отображением. Например, конечный пользователь должен увидеть строку «example.みんな», а не строку «example.xn--q9jyb4c». (Это преобразование — пример обработки, обеспечивающей UA-готовность).
✓	По умолчанию отображайте доменные имена как U-метки. Показывайте пользователю A-метки только тогда, когда это приносит выгоду.
!	Помните, что могут быть доменные имена, где используется несколько алфавитов. ▪ Некоторые символы Unicode могут выглядеть одинаковыми для человеческого глаза, но разными для компьютеров. Например, латинская O, кириллическая O и греческая омикрон O. ▪ Строки с несколькими наборами символов часто встречаются в тесто связанных алфавитах (*напр.*, японские наборы символов кандзи, катакана, хирагана и ромадзи.) Смешивание алфавитов также может использоваться для злонамеренных целей, таких как фишинг. Для проверки того, что алфавиты в последовательности Unicode соответствуют передовой практике, применяется технический стандарт Unicode № 39 «Механизмы безопасности Unicode»[7]. ▪ Если пользовательский интерфейс обращает внимание пользователя на такие строки, избегайте предвзятого отношения к пользователям нелатинских алфавитов. Дополнительные сведения об аспектах обеспечения безопасности при использовании Unicode: http://unicode.org/reports/tr36.

[7] См. https://www.unicode.org/reports/tr39/#Restriction_Level_Detection

✓	Помните о наличии символов, не назначенных и запрещенных для использования в доменных именах. ▪ См. RFC 5892

Unicode	
✓	Используйте поддерживаемые API, совместимые с Unicode.
✗	Используйте стандартные, хорошо отлаженные API для выполнения следующих задач: ▪ Преобразование формата строк. ▪ Определение алфавита строки. ▪ Определение того, содержит ли строка символы нескольких алфавитов. ▪ Нормализация/декомпозиция Unicode.
✗	Не используйте UTF-7 и ограничивайте использование UTF-32. ▪ UTF-7 устарел. ▪ UTF-32 использует четыре байта для каждой кодовой точки. Поскольку каждая кодовая точка занимает одинаковое количество места и может быть напрямую проиндексирована в массивах, ее удобно использовать в программном коде, но она может оказаться слишком громоздкой для хранения в файлах и базах данных.
✗	Не используйте UTF-16, кроме случаев, когда это настоятельно необходимо (как в некоторых API Windows и приложениях Javascript). В UTF-16 16 битов могут представлять только символы от 0x0 до 0xFFFF. Значения выше этого диапазона (от 0x10000 до 0x10FFFF) используют пары псевдосимволов, известных как суррогаты. Если не выполнить тщательное тестирование обработки суррогатных пар, могут возникнуть сложные ошибки и потенциальные дыры в безопасности.
✓	Используйте UTF-8 в файлах cookie, чтобы приложения могли правильно их считывать.
✓	Используйте документы протоколов и таблиц IDNA 2008: ▪ RFC 5891 ▪ RFC 5892
✗	Не используйте IDNA 2003, который был заменен на IDNA 2008.
!	Поддерживайте таблицы IDNA и Unicode соответствующих версий. Например, если приложение не применяет классификационные правила в документе таблиц для интерпретации кодовых точек как они введены (RFC 5892), должны быть получены таблицы IDNA, которые соответствуют поддерживаемой в системе версии Unicode. Эти таблицы не обязательно должны представлять последнюю версию Unicode, но они должны быть согласованными.

✓	Проверяйте метки, используя правила IDNA 2008 для полной метки. ▪ Иногда целесообразна дополнительная проверка; например, если приложению известно, какие алфавиты разрешены в используемых им доменных именах.

Общие вопросы	
✓	Используйте авторитетные ресурсы для проверки доменных имен. Не делайте устаревших *бессистемных* предположений, таких как «длина всех TLD не должна превышать 6 символов».
✓	Убедитесь, что продукт или функция правильно обрабатывает числа. Например, числовые символы ASCII и азиатские иероглифы, представляющие числа, в определенных ситуациях должны обрабатываться как числа.
!	Ищите почтовые адреса, которые могут быть адресами EAI, в неожиданных местах: ▪ Метаданные исполнитель/автор/фотограф/авторские права. ▪ Метаданные шрифтов. ▪ Контактные данные DNS. ▪ Двоичная информация о версии. ▪ Вспомогательная информация. ▪ Контактные данные OEM. ▪ Регистрация, обратная связь и другие формы.
!	Ограничьте список кодовых точек, которые разрешено использовать при создании новых доменных имен и адресов электронной почты: Все продукты, обрабатывающие адреса электронной почты, должны принимать интернационализированные адреса электронной почты, в локальной части которых допускается использование большинства символов UTF-8. Однако приложение или сервис не должны разрешать использование всех символов при создании пользователем нового IDN-домена или адреса EAI. Изначальный запрет на создание определенных IDN-доменов или адресов электронной почты может снизить вероятность возникновения проблем с безопасностью и доступностью. (ПРИМЕЧАНИЕ: Однако рекомендуется, чтобы программное обеспечение принимало такие строки в случае их получения.)
!	Помните, что универсальное принятие не всегда можно измерить с помощью одних только сценариев автоматизированного тестирования. Например, не всегда можно протестировать, как приложение или протокол обрабатывает сетевой ресурс, и иногда лучше всего проверить соблюдение требований путем анализа функциональной спецификации и проекта.

!	Ошибочно считать, что, поскольку компонент не вызывает напрямую API-интерфейсы разрешения имен или не использует напрямую адреса электронной почты, это не влияет на него. Следует понимать, как компонент получает доменные имена — это не всегда происходит через взаимодействие с пользователем. Ниже приведены некоторые примеры того, как компонент может получить доменное имя: ▪ Групповая политика. ▪ Запрос LDAP. ▪ Файлы конфигурации. ▪ Реестр Windows. ▪ Передача или получение из другого компонента или функции.
✓	Анализируйте код, чтобы избежать атак на переполнение буфера. ▪ В Unicode строки могут удлиняться или сокращаться при выравнивании регистра или нормализации. ▪ При преобразовании символов текстовые строки могут существенно удлиняться или сокращаться.

Прочие трудности	
Механизм обнаружения и преобразования наборов символов	Некоторые старые почтовые приложения использовали локальные кодировки символов и не могли обнаружить и преобразовать текст в UTF-8 и обратно по мере необходимости. Это было особенно справедливо для заголовков электронных писем (Кому, Копия, Скрытая копия, Тема).
Управление несколькими адресами электронной почты, идентифицирующими одного пользователя	Когда у пользователя несколько адресов электронной почты, такими идентификаторами личности иногда сложно управлять. Почтовые программы могут направлять трафик, адресованный этим псевдонимам, в один и тот же ящик, но приложения могут по-прежнему обрабатывать такие адреса как разные идентификаторы.

Авторитетные источники данных о доменных именах: Списки корневой зоны DNS и IANA

Есть два источника официального списка TLD. Первый — это сама корневая зона DNS. Она подписана с использованием расширений безопасности системы доменных имен (DNSSEC), поэтому подлинность данных может быть аутентифицирована DNS-сервером, поддерживающим DNSSEC, хотя эту информацию довольно сложно анализировать как текстовый файл. Другим источником является текстовый файл TLD, который публикует IANA (по одному TLD на строку в алфавитном порядке). Эти файлы хранятся на веб-серверах https, поэтому при загрузке рекомендуется проверять сертификат безопасности транспортного уровня (TLS), чтобы убедиться в подлинности загружаемого файла.

Список TLD можно получить по любой из следующих ссылок:

- https://www.internic.net/domain/root.zone (файл корневой зоны)
- https://data.iana.org/TLD/tlds-alpha-by-domain.txt (текстовый файл TLD)

Электронная почта с IDN-доменами и почему это не то же самое, что EAI

При интернационализации адреса электронной почты (EAI) предпочтительны доменные имена UTF-8; использовать A-метки в кодировке ASCII не рекомендуется. В некоторых почтовых системах вместо полной поддержки EAI реализованы частичные меры для обработки адресов электронной почты с IDN-доменами. Поскольку IDN-домены могут быть представлены в виде A-меток ASCII, часть существующих программ разрешает указывать IDN-домены в адресе электронной почты в ASCII или Unicode. Например, некоторые программы будут одинаково обрабатывать эти два IDN-адреса для всех целей (отправка, получение и поиск):

user@example.みんな **=** user@example.xn--q9jyb4c

Однако некоторые программы не будут считать эти адреса одинаковыми, даже если оба они действительны, поскольку перед сравнением не выполняют преобразование A-метки («xn--q9jyb4c») в эквивалентную U-метку («みんな»). Это может привести к непредсказуемому взаимодействию с пользователем. Взаимодействие с пользователем может особенно усложниться, если программное обеспечение преобразует U-метки в A-метки для «совместимости». При отправке ответов или пересылке сообщений может вырасти количество адресов, которые визуально отличаются или не позволяют выполнить поиск и сортировку надлежащим образом.

Как и в приведенном ниже примере, ряд программ может попытаться преобразовать локальную часть адреса электронной почты с использованием Punycode, алгоритма преобразования A-меток в U-метки (и *наоборот*). Этот вид преобразования недопустим и создаст недействительные, не позволяющие доставить почту адреса.

Никогда не пытайтесь преобразовать локальную часть адреса электронной почты в другую форму

✓ 用戶@example.みんな

✗ xn--youq53b@example.xn--q9jyb4c

Надежное программное обеспечение и сервисы, готовые к UA, должны правильно распознавать и обрабатывать все эти форматы, — как локальные части UTF-8, так и U-метки UTF-8 в адресах, — а также принимать A-метки в адресах для обратной совместимости.

Создание ссылок и соответствующие проблемы

Современное программное обеспечение иногда дает пользователю возможность автоматически создать гиперссылку, просто введя строку, которая выглядит как веб-адрес, адрес электронной почты или сетевой путь. Например, ввод «www.icann.org» в сообщение электронной почты может привести к автоматическому созданию активной ссылки на сайт http://www.icann.org, если приложение распознает «www.» как начальную метку или «.org» как TLD.

Создание ссылки — это действие, при котором приложение принимает строку и динамически определяет, следует ли создать гиперссылку на местоположение в

интернете (http:// или https://) или адрес электронной почты (mailto:). При этом механизм создания ссылок должен слаженно работать для всех правильно сформированных веб-адресов, адресов электронной почты и сетевых путей.

При генерации ссылок применяются алгоритмы и правила, созданные разработчиками программного обеспечения для определения того, следует ли считать строку ссылкой или нет. Это связано с возможностями человека идентифицировать строку как доменное имя. Хотя браузеры, почтовые клиенты и текстовые редакторы — очевидные места, есть множество других приложений, которые принимают такие решения.

Рекомендации по эффективной практике

1. Старайтесь создавать ссылки на основе явных префиксов протокола (*напр.,* «https://», «ftp://», «mailto:»), но выполняйте такую операцию, только если остальная часть строки сформирована правильно.

Пример строки	**Ожидаемое поведение/результат**
example.com	Ссылка не создается, так как протокол не указан.
http://example.com	Гиперссылка создается, поскольку протокол четко указан.
http:example.com	Ссылка не создается из-за неправильного синтаксиса (отсутствует //).
http://example.a	Ссылка не создается, потому что «a» не является TLD.
http://example..ab	Ссылка не создается из-за неправильного синтаксиса (две точки подряд).
http://普遍接受-**测试**.世界	Гиперссылка создается, поскольку протокол четко указан.

2. Старайтесь создавать ссылки на основе неявных префиксов протоколов *(напр.,* «www» подразумевает «http://www»).

Пример строки	**Ожидаемое поведение/результат**
www.example.com	Гиперссылка создается, поскольку протокол косвенно указан[8]

[8] Примечание: фактический сайт может быть доступен только по https и для него необходим префикс https:// вместо http://. Если это так, гиперссылка может не работать.

label@example.com	Создается команда mailto: label@example.com, поскольку протокол косвенно указан.

3. HTML, прилегающий к URL-адресам с двунаправленным текстом, может содержать коды, влияющие на направление отображения текста. В ссылке следует сохранить то же самое направление отображения.

4. Если TLD используются в качестве «специального токена» для определения возможности создания ссылки, должны быть охвачены все TLD. Список TLD следует часто обновлять.

Unicode — справочная информация и атрибуты кодовых точек

Стандарт Unicode развивается с тех пор, как в 1991 году была опубликована его первая версия Unicode 1.0. В каждую версию добавляются новые символы и кодовые точки для обработки большего количества языков и алфавитов. Текущая версия — 12.1.

У каждой кодовой точки в Unicode есть набор атрибутов, таких как Uppercase_Letter, Decimal_Number или Nonspacing_Mark. У многих символов есть атрибут алфавита, такой как латиница, хань (китайский) или арабский, в то время как у других, например у знаков пунктуации, такого атрибута нет.

Как указано ниже, IDNA использует атрибуты кодовых точек, чтобы определить, какие символы разрешены в IDN-доменах. Атрибуты кодовых точек описаны в документе UAX#44 «База данных символов Unicode».

UTF8, UTF16 и другие способы кодирования

Кодовая точка Unicode может иметь числовое значение в диапазоне от нуля до 0x10FFFF. Поскольку один байт может содержать только значения от 0 до 0xFF, для хранения кодовых точек Unicode требуется многобайтовое кодирование.

В первоначальной версии Unicode было менее 64K (0xFFFF) кодовых точек, поэтому каждая кодовая точка могла поместиться в 16-битное целое число. Это привело к двухбайтовому кодированию, известному как USC или UCS-2. Когда количество кодовых точек в Unicode превысило 64K, USC был расширен до UTF-16[9], где для значений больше 64K используются пары в остальных случаях недействительных 16-битных кодовых точек, называемые *суррогатными* парами. Хотя эта система работает, она привела к проблемам отладки, так как суррогаты усложняют любой код для определения количество кодовых точек в строке или сортировки строк по порядку кодовых точек. Дополнительная проблема заключается в том, что некоторые компьютеры, в частности созданные IBM, первым хранят старший байт 16-разрядного значения («обратный порядок байтов»), а некоторые, например компьютеры Intel, первым хранят младший байт («прямой порядок байтов»). Вследствие этого у UTF-16 есть два варианта хранения: UTF-16BE и UTF-16LE. Существуют методы обнаружения и устранения проблем с порядком байтов, но они могут привести к ошибкам. В

[9]Подробные технические сведения о UTF-8, UTF-16 и UTF-32 см. в разделе 3.10 стандарта Unicode: https://www.unicode.org/versions/Unicode12.0.0/ch03.pdf.

настоящее время UTF-16 в основном используется в существующих приложениях с API-интерфейсами Microsoft Windows, а также в языках Java и Javascript.

При альтернативном кодировании UTF-8 каждая кодовая точка Unicode — это строка переменной длины от одного до четырех байтов. У формата UTF-8 несколько преимуществ перед UTF-16, в том числе то, что подмножество ASCII Unicode кодируется как один байт, поэтому любая строка ASCII автоматически является строкой UTF-8. Код UTF-8 обычно более компактен, чем его эквивалент в формате UTF-16, и его легче сортировать, поскольку строки UTF-8, отсортированные в порядке байтов, автоматически располагаются в порядке кодовых точек. IDNA и EAI требуют кодировки UTF-8.

UTF-32 — это простой формат, в котором каждая кодовая точка хранится как 32-битное целое число. Это удобно для внутренней обработки в программах, поскольку кодовые точки в массиве UTF-32 можно индексировать напрямую, но редко используется при хранении из-за большого объема.

IDNA — краткая история и современное состояние

Стандарт «Интернационализированные доменные имена в приложениях (IDNA)» впервые был определен IETF в 2003 году как IDNA2003[10]. Он содержал алгоритм Nameprep для преобразования кодовых точек Unicode в элементах доменного имени к стандартному виду и алгоритм Punycode для кодирования меток кодовых точек Unicode в ASCII. Nameprep включает в себя преобразования, такие как замена верхнего регистра на нижний.

После того как был накоплен некоторый опыт работы с IDNA, IETF разработала и опубликовала в 2010 году пересмотренную спецификацию, известную как IDNA2008[11]. В спецификации IDNA2008 были введены термины U-метка и A-метка, а этап Nameprep был исключен с рекомендацией выполнять преобразование регистра в соответствии с языковым стандартом и средой приложения. В 2011 году IDNA2008 был обновлен для поддержки Unicode 6.0 в RFC 6452 и продолжает пересматриваться IETF.

На практике слишком во многих реализациях все еще используется IDNA2003. Немногие библиотеки используют таблицы (например, включенные в IDNA2003), созданные для IDNA2008. Для IDNA2008 не существует языковых правил преобразования, кроме стандартных правил выравнивания и нормализации регистра, включенных в стандарт Unicode.

Единственным исключением является несколько правил преобразования из UTS#46, «Обработка Unicode для обеспечения совместимости с IDNA». Они определяют, следует ли принимать или преобразовывать ряд общих символов, которые подлежат преобразованию согласно IDNA2003, но разрешены в качестве символов в IDNA2008. Важно, чтобы приложения обрабатывали эти символы в соответствии с IDNA2008, а не IDNA2003, и чтобы при использовании UTS#46 обеспечивалась совместимость с IDNA2008.

---

[10] Определение дано в RFC 3490, 3491 и 3492.
[11] Определение дано в RFC 5890, 5891, 5892, 5893, 5894 и 5895.

Сценарии использования для тестирования

Программное обеспечение, предназначенное для обработки IDN-доменов и почтовых адресов EAI, должно быть протестировано в широком диапазоне доменных имен и адресов. См. документUASG 004, *«Сценарии использования для оценки готовности к универсальному принятию»*, в котором представлена группа тестовых примеров.

Обновление программного обеспечения для EAI

Соответствие требованиям EAI требует обновления почтовых серверов, программного обеспечения для отправки и доставки, почтовых пользовательских агентов и веб-почты, а также всех приложений, которые обрабатывают адреса электронной почты и отправляют почту.

Подробный обзор EAI, соответствующих проблем и способов реализации см. в документе UASG 012, *«Интернационализация адреса электронной почты (EAI):Техническое описание».*

Дополнительные темы

Наборы сложных символов

Сведения о наборах сложных символов могут представлять ограниченный интерес для тех, кто не занимается разработкой собственных алгоритмов анализа строк или библиотек отображения. Тем не менее, здесь приведена общая информация, чтобы все читатели в дальнейшем могли распознать ошибки в коде, связанные с такими наборами символов, если они возникнут при взаимодействии с пользователями.

Для отформатированного HTML-текста на веб-страницах и в электронной почте в стандартах HTML предусмотрены продвинутые функции обработки и отображения сложного и двунаправленного текста, которые разработчики должны понимать и использовать при визуализации текста. См. раздел стандарта WHATWG HTML, посвященный визуализации[12], и соответствующий раздел стандарта W3C HTML[13].

Языки с письмом справа налево и Unicode-совместимость

В некоторых алфавитах, таких как латиница и деванагари, символы в горизонтальных строках отображаются слева направо. В других, например арабском или иврите, символы отображаются справа налево. Кроме того, текст может быть двунаправленным, когда в алфавите с письмом справа налево используются цифры, написанные слева направо, или тогда, когда в тексте есть слова из английского или других языков с направлением письма слева направо.

[12] Доступно по адресу https://html.spec.whatwg.org/multipage/rendering.html
[13] Доступно по адресу https://www.w3.org/TR/2018/WD-html53-20181018/rendering.html

Если горизонтальное направление текста не является равномерным, возможны проблемы и неоднозначности. Для решения этого вопроса есть алгоритм определения направления в двунаправленном тексте Unicode.

Двунаправленный алгоритм Unicode определяет набор правил, которые должны применяться приложением, чтобы обеспечить правильный порядок при отображении. Обычно его называют «алгоритмом двунаправленного отображения текста».

Алгоритм двунаправленного отображения текста

Алгоритм двунаправленного отображения текста описывает, как программное обеспечение должно обрабатывать текст, который содержит последовательности символов с написанием слева направо (LTR) и справа налево (RTL). Основное направление,[14] назначенное фразе, будет определять порядок отображения текста. Последовательности символов могут отображаться слева направо или справа налево. В этом документе основное направление текста слева направо, поэтому все последовательности символов отображаются так, что первая находится слева от второй.

Чтобы узнать направление последовательности, слева направо или справа налево, каждому символу в Unicode присвоено соответствующее свойство направления. Большинство букв имеет строгую типизацию (символы со строгой типизацией) — LTR (слева направо) или RTL (справа налево) в зависимости от алфавита, в состав которого они входят. Последовательность строго типизированных символов RTL отображается справа налево. Это направление не зависит от окружающего базового направления. Например:

(LTR) пример — مثال (RTL).

В строке можно смешивать разнонаправленный текст. При этом алгоритм двунаправленного отображения выбирает отдельное направление для вывода каждой последовательности смежных символов с одинаковым направлением.

У пробелов и большинства знаков препинания в Unicode нет строгого типа LTR или RTL, потому что они могут использоваться в любом алфавите. Поэтому они классифицируются как нейтральные символы или символы с нестрогой типизацией. Символы с нестрогой типизацией — это те, которые обычно используются в одном направлении, но в некоторых ситуациях могут использоваться в другом. Примеры таких символов:

- Европейские цифры.
- Арабские цифры.
- Арифметические и валютные символы.
- Общие для многих алфавитов знаки пунктуации, такие как двоеточие, запятая, точка и неразрывный пробел.

[14] В HTML основное направление либо наследуется от направления текста в документе, используемого по умолчанию (слева направо), либо задается явно ближайшим родительским элементом, имеющим атрибут направления «dir».

Направление нейтральных символов неопределимо в отсутствие контекста. Примеры:

- Знаки табуляции.
- Разделители абзацев.
- Большинство других символов пробела.

Когда нейтральный символ находится между двумя строго типизированными символами с одинаковым направлением, ему также присваивается это направление. Например, нейтральный символ между двумя символами RTL будет обрабатываться как символ RTL, продолжая строку в этом направлении:

- مثال.نطاق

Даже если между двумя строго типизированными символами есть несколько нейтральных символов, все они будут обрабатываться одинаково.

Когда пробел или знак пунктуации встречается между двумя строго типизированными символами, имеющими разную направленность, нейтральные символы обрабатываются так, как если бы они имели преобладающее основное направление. Например:

- example. مثال

Вспомните, что в этом документе основное направление последовательности символов слева направо, поэтому *example* — домен второго уровня, а مثال — TLD.

Если направление не переопределено, у чисел сначала всегда кодируются и вводятся цифры старшего разряда, и числа отображаются в направлении LTR. Нестрогая типизация направления распространяется только на все число целиком.

Полная информация об алгоритме двунаправленного отображения содержится в Техническом отчете по Unicode № 9.

Правило двунаправленного отображения доменных имен

Двунаправленное доменное имя — это имя, которое содержит хотя бы одну метку RTL. Правило двунаправленного отображения доменных имен, сформулированное в RFC 5893[15], ограничивает кодовые точки в именах так, чтобы не было двух имен, которые представляют собой разные последовательности кодовых точек, но отображаются одинаково из-за правил двунаправленного отображения.

Соединители

В алфавитах некоторых языков отдельные фонемы записываются в виде двух символов, называемых диграфом. Другими словами, диграф — это группа из двух последовательных букв, которые представляют один звук (или фонему).

[15] *Алфавиты с направлением письма справа налево в интернационализированных доменных именах в приложениях (IDNA)*, RFC 5893, https://www.rfc-editor.org/info/rfc5893

Примеры диграфов в английском языке		
ch (как в church) ph (как в phony)	th (then) th (think)	sh (shoe) gh (rough)

Некоторые диграфы объединены полностью в виде лигатур. На письме и в типографике лигатура возникает при объединении двух или более графем или букв в один глиф. Примером является символ амперсанда (&), возникший из двух смежных латинских букв *e* и *t* («et» означает «и»). При наборе текста на английском языке fi и ffi часто отображаются в виде лигатур.

Если лигатуры и диграфы имеют одинаковое толкование на всех языках, использующих данный алфавит, нормализация Unicode обычно устраняет различия и обеспечивает их согласование. Если у них разное толкование, для согласования должны использоваться альтернативные методы (вероятно, выбранные на уровне регистратуры) или пользователей необходимо обучить, чтобы они понимали невозможность согласования. Пример различного толкования приведен в разделе 4.3 RFC 5894[16]. Консорциум Unicode предлагает две основные стратегии определения поведения конкретного символа при соединении после применения алгоритма двунаправленного отображения для обработки символов соединителей нулевой ширины, так называемых ZWJ и ZWNJ. (Подробнее об этих соединителях см. http://www.unicode.org/L2/L2005/05307-zwj-zwnj.pdf.)

- При формировании текста реализованный алгоритм может обратиться к исходному резервному хранилищу для поиска смежных символов ZWNJ или ZWJ.
- В качестве альтернативы реализованный алгоритм может заменить ZWJ и ZWNJ внешним атрибутом, связанным с этими смежными символами, чтобы информация не мешала алгоритму двунаправленного отображения и сохранялась при перегруппировке символов. После применения алгоритма двунаправленного отображения эту внешнюю информацию можно использовать для правильного формирования строки.

Регистратуры доменных имен и любые другие структуры, которые позволяют создавать доменные имена (*напр.,* приложения, создающие метки третьего уровня и ниже) должны соблюдать правила двунаправленного отображения доменных имен, чтобы обеспечить системный подход к отображению и предотвратить возможность перепутать имена, которая может использоваться для омографических атак.

Подробнее о соединителях см. в разделе 4.3 RFC 5894.

Омоглифы и схожие символы

Омоглифы — это символы, которые из-за сходства по размеру и форме кажутся идентичными или похожими друг на друга до степени смешения. Они часто встречаются при смешивании латинского, кириллического и греческого алфавитов.

[16] *Интернационализированные доменные имена в приложениях (IDNA): История вопроса, пояснение и обоснование*, RFC 5894, https://www.rfc-editor.org/rfc/rfc5894.html#section-4.2

Например, латинская буква «o» (код U+006f), кириллическая строчная буква «о» (код U+043e) и греческая строчная буква омикрон «ο» (код U+03bf). Иногда омоглифы встречаются в одном шрифте, такие как строчная буква хорватского алфавита «lj» (код U+01c9) и две буквы «lj» (код U+006c U+006a). Другие примеры см. в следующей таблице: http://homoglyphs.net/.

Чтобы предотвратить создание доменных имен с омоглифами, регистратуры должны применять правила генерирования меток (LGR), которые ограничивают список кодовых точек в метке набором символов одного алфавита или совместимых алфавитов. У каждой регистратуры должны быть LGR для всех алфавитов, на которых она регистрирует доменные имена[17].

Подробнее о механизмах безопасности Unicode для обнаружения схожести строк см. здесь:

- http://www.unicode.org/reports/tr39/#Confusable_Detection

Подробнее о схожих до степени смешения символах и соответствующей передовой практике см. здесь:

- Общая информация и руководство M3AAWG по борьбе с ненадлежащим использованием Unicode https://www.m3aawg.org/sites/default/files/m3aawg-unicode-tutorial-2016-02.pdf
- Рекомендации M3AAWG по предотвращению злоупотреблений Unicode https://www.m3aawg.org/sites/default/files/m3aawg-unicode-best-practices-2016-02.pdf

Нормализация, выравнивание регистра и подготовка строк

Нормализация Unicode помогает определить, являются ли две строки Unicode одинаковыми, и предоставляет стандартные формы, которые можно использовать для обработки и хранения строк. Некоторые символы могут быть представлены в Unicode несколькими кодовыми последовательностями. Это называется эквивалентностью Unicode. В Unicode предусмотрено два вида эквивалентности:

- Каноническая
- Совместимость

Последовательности, представляющие один и тот же визуальный символ, называются канонически эквивалентными. Эти последовательности имеют одинаковый внешний вид и значение при печати или отображении. Например:

U+006E (латинская строчная буква «n»), за которой следует U+0303 (комбинируемая тильда «◌̃»)	= ñ
U+00F1 (строчная буква «ñ» испанского алфавита)	= ñ

[17] Коллекция LGR регистратур есть в репозитарии практик в области IDN-доменов IANA по адресу https://www.iana.org/domains/idn-tables.

Unicode определяет NFC (форму нормализации C) как каноническую декомпозицию, за которой следует каноническая композиция. Это сокращает текст до минимального количества кодовых точек, не изменяя его внешний вид. Следует отметить, что в этом примере все три символа выше разрешено использовать в соответствии с IDNA2008.

Совместимыми эквивалентами являются последовательности, которые могут выглядеть по-разному, но в некоторых ситуациях имеют одинаковое значение. Это более слабый вид эквивалентности символов или их последовательностей. Например:

U+FB00 (типографская лигатура «ff»)	= ff
U+0066 U+0066 (две латинские буквы «f»)	= ff

В приведенном выше примере кодовая точка U+FB00 определена как совместимая, но не канонически эквивалентная последовательности U+0066 U+0066. Канонически эквивалентные последовательности также совместимы, но обратное не всегда верно.

Следует отметить, что в соответствии с IDNA2008 кодовую точку U+FB00 не разрешено использовать.
Unicode определяет NFKC (форму нормализации KC) как каноническую декомпозицию, за которой следует каноническая композиция. Это сокращает текст до стандартного набора кодовых точек и может изменить его внешний вид. Например, NKFC превращает лигатуру «ff» в две буквы «ff», а символ времени до полудня (U+33C2) в четыре символа «a.m.» (U+0061 U+002E U+006D U+002E).

Чтобы избежать проблем с функциональной совместимостью, возникающих из-за использования канонически эквивалентных, но разных символьных последовательностей, W3C рекомендует использовать для всего текста NFC.

Полный список символов, которые могут измениться при любой из форм нормализации см. здесь: http://www.unicode.org/charts/normalization.

Следует отметить еще несколько моментов:

- Символы в метках IDN-доменов должны быть в форме NFC.
- Когда два приложения совместно используют данные Unicode, но по-разному их нормализуют, могут возникнуть ошибки и потеря данных.
- Консорциум Unicode утверждает, что формы нормализации должны сохранить стабильность с течением времени. Другими словами, строка должна оставаться нормализованной во всех будущих версиях Unicode для обратной совместимости.
- Как уже отмечалось, используйте консервативный подход, рассматривая вопрос о том, какие кодовые точки разрешить в доменном имени.

Советы разработчикам программного обеспечения	
✖	Не пытайтесь нормализовать путем преобразования в верхний регистр или игнорирования несамостоятельных символов, поскольку это затрудняет сортировку, импорт и экспорт данных при копировании, а также извлечение данных клиентскими приложениями и может привести к потере или повреждению данных.
✖	Никогда не разрешайте использовать в доменных именах кодовые точки, которые запрещены в IDNA2008.

Подробнее о нормализации Unicode см. здесь:

- http://www.w3.org/TR/charmod-norm
- http://unicode.org/reports/tr15

Выравнивание и преобразование регистра

Выравнивание и преобразование регистра — это процесс приведения всех символов в строке к одному регистру, обычно нижнему. Преобразование верхнего регистра [AZ] в нижний регистр [az] эффективно срабатывает только для текстовых документов ASCII, но гораздо сложнее в языках, использующих дополнительные символы. Преобразование регистра символов может быть контекстно-зависимым, *напр.*, в случае различных форм греческой сигмы. Оно также может быть обусловлено языковым стандартом, когда преобразуемый символ зависит от локали, в которой интерпретируется текст, *напр.*, турецкая буква I с точкой и без точки в верхнем и нижнем регистрах. Выравнивание регистра не зависит от локали и полученные строки будут интерпретироваться программным обеспечением, в то время как преобразование регистра зависит от локали и предназначено для чтения текста человеком. Наконец, преобразование в верхний регистр и преобразование в нижний регистр — **не** являются обратными функциями.

IDNA2008 позволяет приложениям использовать для IDN-доменов любое подходящее преобразование регистра, поскольку такое преобразование выполняется до проверки кодовых точек. На практике зависящего от локали преобразования идентификаторов не существует, и все используют правила преобразования из UTS#46 Unicode[18].

Советы разработчикам программного обеспечения	
✓	Учитывайте поставленную цель, прежде чем пытаться преобразовать регистр: это общая карта меток, строка на известном языке или что-то еще?
✓	Выполняйте нормализацию Unicode до выравнивания регистра.

[18] UTS#46, *Совместимая обработка Unicode IDNA*, https://www.unicode.org/reports/tr46/#Mapping

Глоссарий и другие ресурсы

Глоссарий

A-метка	Представление метки интернационализированного доменного имени в ASCII-совместимой кодировке (ACE), используемой внутри протокола DNS. A-метки всегда начинаются с префикса ACE «xn--». A-метку можно преобразовать в U-метку и обратно без потери информации.
Префикс ACE	Префикс ASCII-совместимого кодирования «xn--».
ASCII	Американский стандартный код для обмена информацией. ASCII включает латинские символы без диакритических знаков и арабские цифры. ASCII — это подмножество Unicode: каждый символ ASCII также является символом Unicode.
API	Интерфейс программирования приложений (API) — это набор процедур, протоколов и инструментов для создания программного обеспечения и приложений. API может быть предназначен для веб-системы, операционной системы или системы баз данных, и он предоставляет средства разработки приложений для этой системы с использованием конкретного языка программирования.
Кодовое пространство	Диапазон, который определяет нижнюю и верхнюю границы для кодировки.
Кодовая точка	Кодовая точка — это числовое значение в кодовом пространстве. Кодовые точки используются для того, чтобы отличить числовое значение от его кода в виде последовательности битов и отличить абстрактный символ от его конкретного графического представления (глифа).
Корневая зона DNS	Корневая зона — это центральный каталог DNS, который является ключевым компонентом при поиске по DNS; например, при преобразовании имен хостов в IP-адреса.
EAI	Интернационализация адреса электронной почты позволяет использовать символы UTF-8 в адресе электронной почты — в имени домена, локальной части или обеих компонентах.

IANA	Администрация адресного пространства интернета. Она выполняет следующие функции: ▪ Управление корнем DNS, доменами .int и .arpa, а также ресурсом с описанием практики в области IDN-доменов. ▪ Координация глобального пула IP-адресов и номеров AS, прежде всего предоставление их региональным интернет-регистратурам (RIR). ▪ Управлением системами нумерации интернет-протоколов совместно с органами стандартизации.
ICANN	Миссия ICANN — способствовать обеспечению стабильности, безопасности и единства глобального интернета. Для того, чтобы связаться с кем-нибудь в интернете, в компьютер или другое устройство необходимо ввести адрес — имя или номер. Этот адрес должен быть уникальным, чтобы компьютеры могли находить друг друга. ICANN занимается координацией этих уникальных идентификаторов во всем мире. ICANN была сформирована в 1998 году в качестве некоммерческой общественной корпорации и сообщества участников со всего мира.
IDN-домен	Интернационализированное доменное имя. IDN-домены представляют собой доменные имена, который содержат символы UTF-8, выходящие за рамки двадцати шести букв базового латинского алфавита «a-z», цифр 0–9 и дефиса «-».
IDNA	Интернационализированные доменные имена в приложениях.
IDN ccTLD	Национальный домен верхнего уровня, который содержит символы, выходящие за рамки двадцати шести букв базового латинского алфавита «a-z». Примеры: ▪ .рф (Россия) ▪ .صر (Египет) ▪ .السعودية (Саудовская Аравия)
IETF	Инженерная проектная группа интернета (IETF) — большое открытое международное сообщество проектировщиков, операторов, производителей и исследователей сетей, работающих над развитием архитектуры интернета и обеспечением бесперебойной работы интернета. Это сообщество открыто для всех заинтересованных лиц. IETF разрабатывает стандарты интернета, в том числе стандарты, относящиеся к стеку интернет-протоколов (TCP/IP) и веб-протоколам, таким как HTTP и TLS.
Язык	Способ человеческого общения, устный или письменный, подразумевающий использование слов структурированным и общепринятым способом.

Punycode	Алгоритм, который представляет UTF-8 в ограниченном подмножестве символов ASCII, поддерживаемом системой доменных имен (DNS). Punycode используется в A-метках в рамках платформы интернационализированных доменных имен в приложениях (IDNA).
Регистратор	Организация, в которой пользователи регистрируют доменные имена. Регистратор хранит контактные данные и передает техническую информацию в центральный каталог, который называется «регистратура».
Регистратура	Официальная основная база данных всех доменных имен, зарегистрированных в каждом из доменов верхнего уровня (TLD).
RFC	Запрос комментариев (RFC) — это официальный документ Инженерной проектной группы интернета (IETF), проект которого составляется комитетом и затем рассматривается заинтересованными сторонами. В некоторых (но не во всех) документах RFC утверждены стандарты интернета.
Алфавит	Совокупность букв или символов, используемых при письме и представляющих звуки языка.
Доменное имя второго уровня	В иерархии системы доменных имен (DNS) домен второго уровня (SLD или 2LD) — это домен, который находится на один уровень ниже домена верхнего уровня (TLD). Например, в адресе example.com доменом второго уровня TLD .com является example.
U-метка	U-метка — это удовлетворяющая требованиям IDNA строка из символов Unicode, содержащая хотя бы один символ, не входящий в набор ASCII. Ее можно преобразовать в A-метку и обратно без потери информации.
Готовое к UA программное обеспечение или UA-готовность	Программное обеспечение, в равной степени способное принимать, хранить, обрабатывать, проверять и отображать все домены верхнего уровня, IDN-домены и адреса электронной почты.
Unicode	Универсальный стандарт кодирования символов. Он определяет способ представления отдельных символов в текстовых файлах, на веб-страницах и в других видах документов. Unicode был разработан для поддержки символов всех языков мира. Он может поддерживать около 1 000 000 символов. См. http://unicode.org.

UTF	Формат преобразования Unicode. Это способ представления кодовых точек Unicode в виде потока байтов. Предпочтительным форматом UTF для обработки IDN-доменов и EAI является UTF-8. UTF-8 преобразует Unicode в 8-битные байты.
M3AAWG	Рабочая группа по борьбе с злоупотреблениями и вредоносным ПО в системах передачи сообщений и мобильной связи (M^3AAWG) объединяет представителей отрасли для борьбы с ботнетами, вредоносными программами, спамом, вирусами, DoS-атаками и другими видами злоумышленного использования сетевых ресурсов. См. https://www.m3aawg.org/.
W3C	Международный консорциум всемирной сети интернет (W3C) — это международное сообщество, в котором организации-участники, штатный персонал и общественность совместно разрабатывают веб-стандарты, такие как HTML. См. https://www.w3.org/.
WHATWG	Рабочая группа по веб-технологиям применения гипертекста (WHATWG) — это сообщество, заинтересованное в развитии веб-технологий с помощью стандартов и тестов. WHATWG была основана сотрудниками Apple, Mozilla Foundation и Opera Software в 2004 году после семинара W3C. См. https://whatwg.org/.
ZWJ	Соединитель нулевой ширины — это непечатаемый символ, используемый при компьютеризированной верстке текста для некоторых алфавитов, включая арабский и все индийские. При размещении между двумя символами, которые в противном случае не были бы связаны, ZWJ обеспечивает их печатать в связанном виде форме.
ZWNJ	Разделитель нулевой ширины — это непечатный символ, используемый при компьютеризации систем письма, использующих лигатуры. В некоторых языках и алфавитах многие буквы естественным образом связываются при письме со следующей буквой в слове, образуя лигатуру. Однако для правильного отображения определенных префиксов, суффиксов и составных слов используется ZWNJ, который переопределяет это естественное соединение букв и предотвращает их присоединение к следующей букве (но без добавления пробела между ними).

Полный глоссарий ICANN опубликован здесь: https://www.icann.org/icann-acronyms-and-terms/.

RFC и ключевые стандарты

RFC, относящиеся к IDN-доменам	
RFC 3492	**Punycode: Bootstring-кодирование строк Unicode для интернационализированных доменных имен в приложениях (IDNA)** RFC 3492 описывает Punycode как: «*простой и эффективный синтаксис кодирования символов при передаче, предназначенный для использования с интернационализированными доменными именами в приложениях (IDNA)*» Punycode обеспечивает обратимое преобразование строки в формате Unicode в уникальную строку ASCII. Этот RFC определяет общий алгоритм, который называется Bootstring. Данный алгоритм позволяет с помощью строки базовых кодовых точек уникальным образом представлять любые строки кодовых точек, принадлежащих к более широкому набору. https://tools.ietf.org/html/rfc3492
RFC 5890	**Интернационализированные доменные имена в приложениях (IDNA): Определения и структура документа** Этот RFC содержит описание протокола и среды использования интернационализированных доменных имен в приложениях (IDNA) в очередной редакции. https://tools.ietf.org/html/rfc5890
RFC 5891	**Протокол «Интернационализированные доменные имена в приложениях (IDNA)»** Этот RFC определяет механизм протокола, который называется «Интернационализированные доменные имена в приложениях (IDNA)» и предназначен для регистрации и поиска IDN-доменов без необходимости изменения самой DNS. https://tools.ietf.org/html/rfc5891
RFC 5892	**Кодовые точки Unicode и интернационализированные доменные имена в приложениях (IDNA)** RFC 5892 устанавливает правила принятия решений о том, может ли та или иная кодовая точка с учетом или без учета контекста использоваться в составе интернационализированного доменного имени (IDN-домена). https://tools.ietf.org/html/rfc5892

RFC 5893	**Использование алфавитов с направлением письма справа налево для интернационализированных доменных имен в приложениях (IDNA)** Этот RFC вводит новое правило двунаправленного отображения меток интернационализированных доменных имен в приложениях (IDNA) при использовании алфавитов с направлением письма справа налево. https://tools.ietf.org/html/rfc5893
RFC 5894	**Интернационализированные доменные имена в приложениях (IDNA): История вопроса, пояснение и обоснование** В этом справочном документе представлен обзор пересмотренной системы, которая способна обрабатывать новые версии Unicode, и пояснительные материалы к ее компонентам. https://tools.ietf.org/html/rfc5894
RFC 5895	**Преобразование символов интернационализированных доменных имен в приложениях (IDNA) 2008** В этом RFC описываются действия, которые могут выполняться в том или ином варианте реализации протокола после получения введенных пользователем данных перед передачей допустимых кодовых точек в новый протокол IDNA (2008). В нем указано, какую операцию обработки пользовательского ввода необходимо выполнить, чтобы подготовить введенные данные для использования в «сетевом» протоколе. Этот документ также содержит общую процедуру реализации преобразования. https://tools.ietf.org/html/rfc5895
RFC, относящиеся к EAI	
RFC 6530	**Общие сведения и концепция интернационализации электронной почты** Этим стандартом вводится ряд спецификаций, определяющих механизмы и расширения протоколов, которые необходимы для полной поддержки интернационализированных адресов электронной почты. В этом документе описывается, как различные элементы интернационализации электронной почты сочетаются друг с другом, а также описываются взаимосвязи между основными спецификациями, связанными с передачей сообщений, форматами заголовков и обработкой. https://tools.ietf.org/html/rfc6530

RFC 6531	**Расширение протокола SMTP для интернационализации электронной почты** В этом документе описано расширение протокола Simple Mail Transfer Protocol, позволяющее серверам оповещать о возможности приема и обработки интернационализированных адресов электронной почты и интернационализированных заголовков электронной почты. https://tools.ietf.org/html/rfc6531
RFC 6532	**Заголовки сообщений при интернационализации электронной почты** В этом документе указаны усовершенствования формата интернет-сообщений и MIME, позволяющие использовать Unicode в адресах электронной почты и значениях большинства полей заголовков. В этом документе определены усовершенствования формата интернет-сообщений (RFC 5322) и MIME, позволяющие напрямую использовать UTF-8, а не только ASCII в значениях полей заголовка, включая почтовые адреса. Определен новый тип медиа message/global для сообщений, использующих этот расширенный формат. Эта спецификация также снимает ограничение MIME на кодировку при передаче неидентификационных данных для любого подтипа сообщений верхнего уровня, так что части message/global можно безопасно передавать по существующей почтовой инфраструктуре. https://tools.ietf.org/html/rfc6532
RFC 6533	**Интернационализация уведомлений о доставке и расположении сообщений** Этой спецификацией добавляется новый тип интернационализированных адресов электронной почты, позволяющий корректно сохранять исходный адрес получателя, содержащий символы, отличные от ASCII, после понижения версии поддерживаемого стандарта. Кроме того, в ней представлен обновленный список типов данных возвращаемого содержания для уведомлений о доставке и расположении сообщений для поддержки использования нового типа адресов. https://tools.ietf.org/html/rfc6533
RFC 8398	**Интернационализированные адреса электронной почты в сертификатах X.509** Этот документ определяет новую форму имени для включения в поле otherName расширения альтернативного имени субъекта X.509 и альтернативного имени издателя, которое позволяет связать субъекта сертификата с интернационализированным адресом электронной почты. https://tools.ietf.org/html/rfc8398.

RFC 8399	**Обновления RFC 5290 для интернационализации** Описанные в этом документе обновления RFC 5280 обеспечивают соответствие спецификации 2008 года для интернационализированных доменных имен (IDN-доменов) и добавляют поддержку интернационализированных адресов электронной почты в сертификатах X.509. https://tools.ietf.org/html/rfc8399

Ключевые стандарты

ISO 10646 (Unicode)	Чтобы обеспечить общую техническую основу для обработки электронной информации на разных языках, Международная организация по стандартизации (ISO) разработала международный стандарт кодирования под названием ISO 10646. ISO 10646 вводит единый стандарт кодирования символов на всех основных языках мира, включая традиционные и упрощенные китайские иероглифы. Этот широкий набор символов называется универсальным набором символов (UCS). Тот же набор символов определяется стандартом Unicode, в котором дополнительно определены дополнительные свойства символов и другие сведения о применении, представляющие большой интерес для разработчиков. Unicode — это система кодирования символов, разработанная Консорциумом Unicode для поддержки обмена, обработки и отображения текстов на всех основных языках мира. ISO 10646 и Unicode определяют несколько видов кодирования их общего набора: UTF-8, UCS-2, UTF-16, UCS-4 и UTF-32. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=63182
GB18030 (Китай)	GB 18030-2000 — это государственный стандарт Китая, который определяет расширенную кодовую страницу для использования на китайском рынке в дополнение к UTF-8. Кодом внутренней обработки набора символов может и должен быть Unicode; однако стандарт предусматривает, что поставщики программного обеспечения должны гарантировать успешную передачу данных между GB18030 и кодом внутренней обработки. Для всех без исключения продуктов, которые в настоящее время продаются или будут продаваться в Китае, необходимо спланировать переход на поддержку кодовой страницы GB18030. GB18030 — «обязательный стандарт», и правительство Китая регулирует процесс сертификации для более глубокого внедрения GB18030. http://icu-project.org/docs/papers/unicode-gb18030-faq.html

Интернет-ресурсы

API	Интерфейсы программирования приложений Windows (API) https://www.msdn.microsoft.com/enus/library/windows/desktop/ff818516%28v=vs.85%29.aspx Интерфейсы SharePoint API https://msdn.microsoft.com/en-us/library/office/jj860569.aspx Публичный список суффиксов https://publicsuffix.org/list/public_suffix_list.dat Официальный список TLD ICANN http://data.iana.org/TLD/tlds-alpha-by-domain.txt Интерфейсы Android API http://developer.android.com/guide/index.html Интерфейсы MAC IOS API https://developer.apple.com/library/mac/navigation Фреймворк .Net https://msdn.microsoft.com/en-us/library/system.text.encoding(v=vs.110).aspx
Безопасность Unicode	Аспекты безопасности Unicode http://www.unicode.org/reports/tr36 Механизмы безопасности Unicode http://www.unicode.org/reports/tr39
Группировка символов Unicode	Кодовые плоскости Unicode https://www.unicode.org/versions/Unicode12.0.0/ch02.pdf; стр. 44-54 Обзор GB18030 http://icu-project.org/docs/papers/gb18030.html Официальная таблица сопоставления BG18030-2000 и Unicode http://source.icu-project.org/repos/icu/data/trunk/charset/data/xml/gb-18030-2000.xml Нормализация Unicode https://unicode.org/reports/tr15/

Уязвимости Unicode	Технический отчет по Unicode № 36, раздел 3.1 «Уязвимости UTF-8» http://unicode.org/reports/tr36/#UTF-8_Exploit Рекомендации M3AAWG по предотвращению ненадлежащего использования Unicode https://www.m3aawg.org/sites/default/files/m3aawg-unicode-best-practices-2016-02.pdf Общая информация и руководство M3AAWG по борьбе с ненадлежащим использованием Unicode https://www.m3aawg.org/sites/default/files/m3aawg-unicode-tutorial-2016-02.pdf Смотрите также: http://www.unicode.org
Разное	URI http://tools.ietf.org/html/rfc3986 Система доменных имен: Нетехническое объяснение — почему важна универсальная разрешимость http://www.internic.net/faqs/authoritative-dns.html Глоссарий ICANN https://www.icann.org/icann-acronyms-and-terms/

Необходима дополнительная информация?

Группа управления по универсальному принятию (UASG) и сообщество готовы давать рекомендации разработчикам программного обеспечения и специалистам по реализации.

👍 Свяжитесь с нами, чтобы поделиться своими идеями и предложениями на эту тему: info@uasg.tech.
👍 Подпишитесь на лист рассылки для обсуждения универсального принятия http://tinyurl.com/ua-discuss.
👍 Чтобы узнать больше об этой деятельности, посетите http://www.icann.org/universalacceptance.