



Programming for UA Universal Acceptance





Somkiat Puisungnoen

Somkiat Puisungnoen

Update Info 1

View Activity Log 10+

...
Timeline About Friends 3,138 Photos More

When did you work at Opendream?
... 22 Pending Items

Post Photo/Video Live Video Life Event

What's on your mind?

Public Post

Intro
Software Craftsmanship

Software Practitioner at สยามชัมนาภิกิจ พ.ศ. 2556

Agile Practitioner and Technical at SPRINT3r

Somkiat Puisungnoen 15 mins · Bangkok · ⚙️
Java and Bigdata



somkiat.cc

Page Messages Notifications 3 Insights Publishing Tools Settings Help ▾

somkiat.cc
@somkiat.cc

Home Posts Videos Photos

Liked Following Share ... + Add a Button



Programming for UA



Scopes

Domain name
Email address



Accept



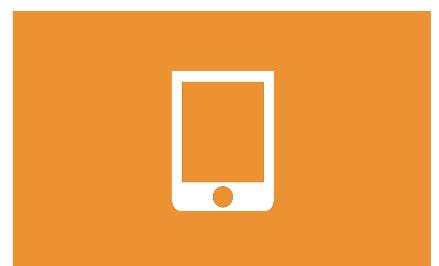
Validate



Process



Store



Display



Scopes

Internationalized Domain Names (IDNs)
Email Address Internationalization (EAI)



Domain name

Cases	Example
New top-level domain	sky, tech, shop, online, site
Longer top-level domain	lifeinsurance
Internationalized	ไทยร่วมใจ.com <u>example.ไทย</u>

<https://data.iana.org/TLD/tlds-alpha-by-domain.txt>



List of top-level Domain (TLD)

```
# Version 2021082000, Last Updated Fri Aug 20 07:07:01 2021 UTC
```

AAA

AARP

ABARTH

ABB

ABBOTT

ABBVIE

ABC

ABLE

ABOGADO

ABUDHABI

AC

ACADEMY

ACCENTURE

ACCOUNTANT

ACCOUNTANTS

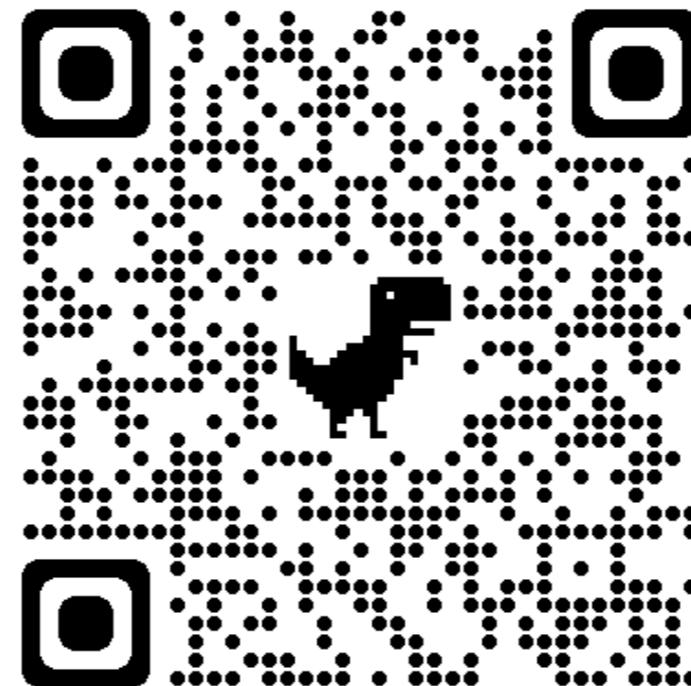
ACO

ACTOR

AD

ADAC

ADS



<https://data.iana.org/TLD/tlds-alpha-by-domain.txt>



Programming for UA

© 2017 - 2018 Siam Chamnkit Company Limited. All rights reserved.

>=5 characters

```
$wget -qO - http://data.iana.org/TLD/tlds-alpha-by-domain.txt |  
awk 'length($0) >= 5'
```

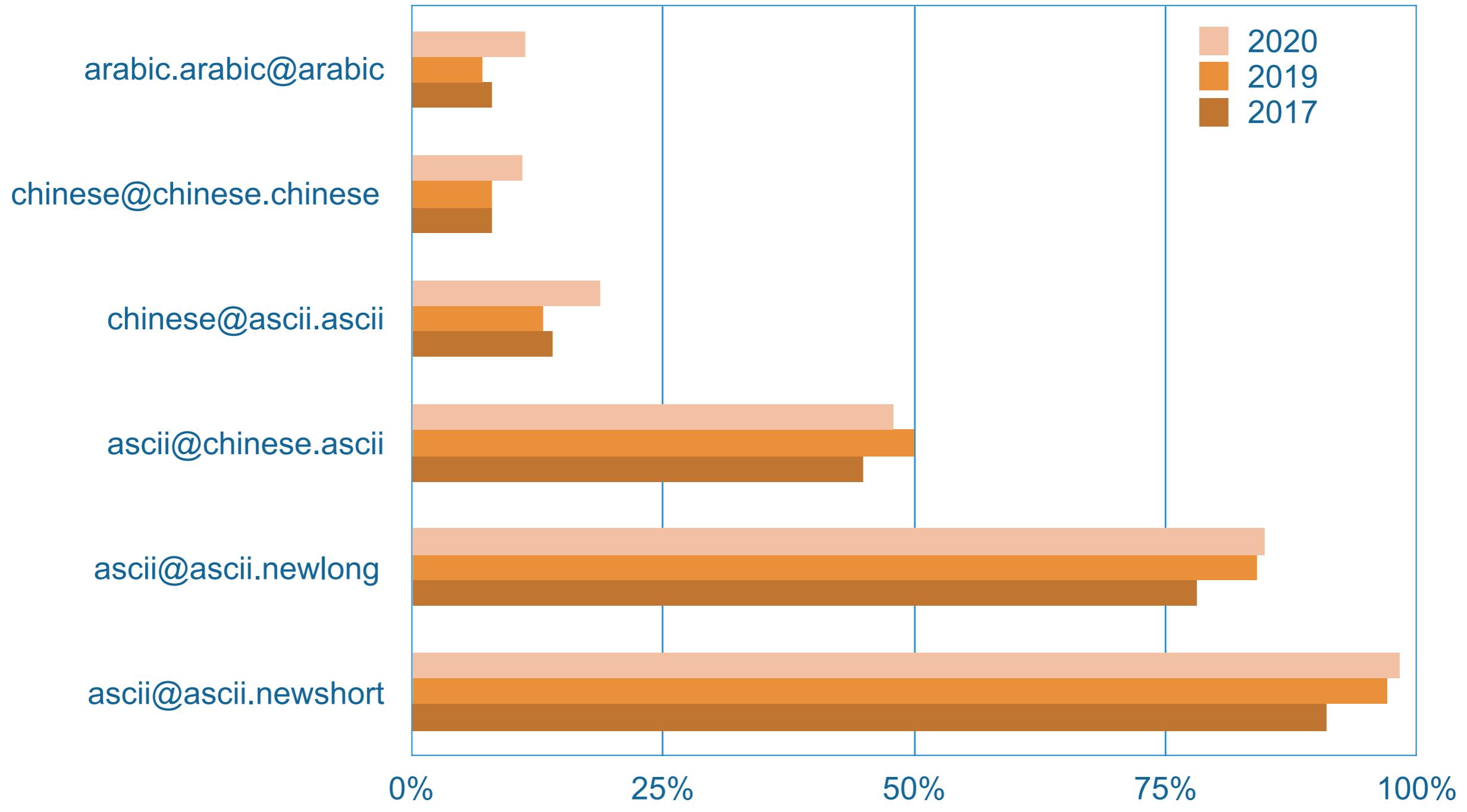


Internationalized Email Address (EAI)

Format	Example
ASCII@IDN	marc@société.org
UTF8@ASCII	ईमेल@example.com
UTF8@IDN	测试@普遍接受-测试.世界
UTF@IDN	موقع.مثال@میل-ای



Usages by website globally



<https://uasg.tech/wp-content/uploads/documents/UASG027-en-digital.pdf>



Programming for UA



Accept



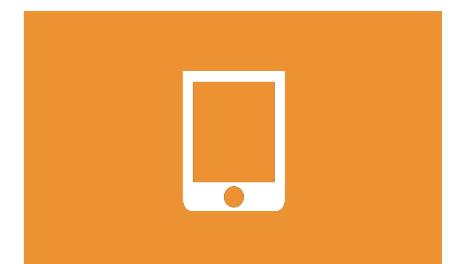
Validate



Process



Store



Display



Tasks for programmer

Validate input
Processing
Store data
Display data



Libraries and framework

Domain name

Language	Framework/Library
Java	Commons Validator
Java	Guava
Java	ICU
Java	JRE
Python3	Django_auth
Python3	Encodings_Idna
Python3	Idna
Rust	Idna

For email address

Java	ICU
Java	Javamail/JakartaMail
Java	JRE
Javascript	Idna-uts46
Javascript	Nodemailer
Javascript	Validator
Python3	Django_auth
Python3	Email_Validator
Python3	Encodings_Idna
Python3	Idna
Python3	Smtplib
Rust	Idna
Rust	Lettre

Language	Framework/Library
C	libcurl
C	libidn2
C#	Mailkit
C#	Microsoft
Go	Idna
Go	Mail
Go	Smtp
Java	Commons-Validator
Java	Guava

<https://uasg.tech/wp-content/uploads/documents/UASG018A-en-digital.pdf>



Example

Working with domain name
Working with Email address



Validate input



Start with data for test cases

<https://uasg.tech/wp-content/uploads/documents/UASG033-en-digital.pdf>



Validate domain name

21	IDN.IDN	විශ්ව-සමූහ-පිරික්ෂුම.ලංකා	Sinhala
22	IDN.IDN	பொது-ஏற்பு-சோதனை.சிங்கப்பூர்	Tamil
23	IDN.IDN	యూనివర్సల్-ఆమ్‌దం-పరీక్.భారత్	Telugu
24	IDN.IDN	ไทย.ເວັດສອບ.	Thai
25	IDN.IDN	普遍适用测试.我爱你	Simplified Chinese
26	IDN.IDN	普遍適用測試.台灣	Traditional Chinese
27	IDN.ASCII	ሁ.ለንአቀፍ-ተ.ቀበደንት-.መ.ከራ.	Ethiopic
28	IDN.ASCII	ការសាកល្បែនទួលយកជាមន្ត្រជាតិ.com	Khmer
29	IDN.ASCII	အလုံးစုံလက်ခံမှုစမ်းသပ်ချက်.com	Myanmar

<https://uasg.tech/wp-content/uploads/documents/UASG004-en-digital.pdf>



Validate email address

50	Unicode@IDN.IDN	ഇമയിൽ-പരിശോധന@സാർവ്വതിക-സ്വികാര്യതാ-പരിശോധന.ഭാരതം	Malayalam
51	Unicode@IDN.IDN	ଇମେଲ୍-ରେଷ୍ଟ୍‌@ଯୁନିଭରସାଳ-ଏକସେପ୍ତ୍-ରେଷ୍ଟ୍.ଭାରତ	Oriya
52	Unicode@IDN.IDN	ඉ-තැපැල්-පිරික්සුම@විශ්ව-සමූහ-පිරික්සුම.ලංකා	Sinhala
53	Unicode@IDN.IDN	மின்னஞ்சல்-சோதனை@பொது-ஏற்பு-சோதனை.சிங்கப்பூர்	Tamil
54	Unicode@IDN.IDN	ఇమెయిల్-పరీక్ష@యూనివర్స్-ఆమ్‌దం-పరీక్ష.భారత్	Telugu
55	Unicode@IDN.IDN	อีเมลทดสอบ@ยูเนกซ์.ไทย	Thai
56	Unicode@IDN.IDN	电子邮件测试@普遍适用测试.我爱你	Simplified Chinese
57	Unicode@IDN.IDN	電子郵件測試@普遍適用測試.台灣	Traditional Chinese

<https://uasg.tech/wp-content/uploads/documents/UASG004-en-digital.pdf>



Let's coding for Domain name



Coding with Java

java.net.InetAddress

Apache commons validator (TLD outdated)

ICU (International Components for Unicode)



Popular libraries

“Regular Expression”

Library	Occurrence (projects)	Status
hibernate-validator	62963	Not UA-Ready. RegEx via annotations; Hibernate implementation of <i>validation-api</i> .
validation-api	25190	Not UA-Ready. RegEx via annotations.
springfox-bean-validators	12501	Not UA-Ready. RegEx via annotations; SpringFox implementation of <i>validation-api</i> .
commons-validator	4906	Not UA-Ready. Relies on a static list of TLDs from 2017.
icu4j	886	UA-Ready. IDNA2008.
libidn	29	Not UA-Ready. IDNA2003, deprecated and ported to the Java language as “java.net.IDN”.

<https://uasg.tech/wp-content/uploads/documents/UASG033-en-digital.pdf>



Apache common validator



Apache Commons™
<http://commons.apache.org/>

Apache Commons Validator™ Last Published: 03 August 2020 | Version: 1.7 ApacheCon Apache Commons

VALIDATOR

- Overview
- Download
- Release Notes
- Dependencies

DOCUMENTATION

- Framework
- Routines
- Wiki

DEVELOPMENT

- Javadocs
- Mailing Lists
- Issue Tracking
- Source Repository
- Building

PROJECT DOCUMENTATION

- Project Information
- About
- Summary
- Team

Commons Validator

A common issue when receiving data either electronically or from user input is verifying the integrity of the data. This work is repetitive and becomes even more complicated when different sets of validation rules need to be applied to the same set of data based on locale. Error messages may also vary by locale. This package addresses some of these issues to speed development and maintenance of validation rules.

Releases

See the [Downloads](#) page for current/previous releases. For details of what's new in each version see the [Release Notes](#). [Community Notes](#) on release are maintained on the [Apache Commons Wiki](#).

Overview

Features

Validator provides two distinct sets of functionality:

1. A configurable (typically XML) validation engine
2. Reusable "primitive" validation methods

Your validation methods are plugged into the engine and executed against your data. Often, these methods use resources specific to one application or framework so Commons Validator doesn't directly provide pluggable validator actions. However, it does have a set of common validation methods (email addresses, dates, URLs, etc.) that help in creating pluggable actions.

Usage

In order to use the Validator, the following basic steps are required:

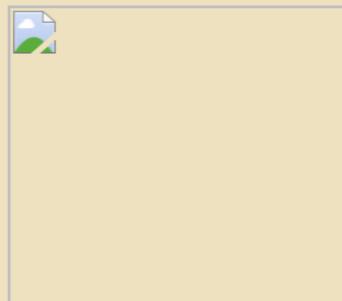
<https://commons.apache.org/proper/commons-validator/>



ICU4J

 **ICU - International Components for Unicode**

Navigation

- 

[Unicode](#)
[About ICU](#)

- [· ICU Home](#)
- [· Download ICU](#)

[Demos & Tools](#)

- [· ICU4C Demos](#)
- [· ICU Collation Demo](#)
- [· ICU4J Demos](#)
- [· Data Customizer](#)

[Documents](#)

- [· User Guide](#)
- [· ICU4C Readme](#)
- [· ICU4J Readme](#)
- [· Docs & Papers](#)

ICU-TC Home Page

News

●●● **Do you need ICU to work on EBCDIC platforms?** ●●●

- ICU4C used to work on IBM mainframes (OS/390, z/OS) and other native-EBCDIC platforms (OS/400, AS/400, iSeries, etc.).
- This has not been tested since [ICU 59](#) moved to C++11 (2017). Apparently there are now C++11 compilers available for these platforms.
- **We need help:** Someone needs to build ICU4C on such a machine, fix C++ compiler issues (if any), and add the necessary support code to our CI. We will assist you, especially with EBCDIC-specific character and string handling.
- **Otherwise we will remove the support code for non-ASCII-family platforms.** ([ICU-21672](#))
- Please contact us via the [icu-support mailing list](#). See the thread "[ICU users: Do you need ICU to work on EBCDIC platforms?](#)"

Special Notice About Branch Renaming: On 2021-March-24 we renamed the `master` branch to `main` branch.

If you have a local git clone of the repo, you need to rename your default branch:

```
$ git branch -m master main  
$ git fetch origin
```

<http://site.icu-project.org/home>



Coding with Python

Library	Occurrence in projects	Status
idna	70789	UA-Ready. IDNA2008.
validators	1660	Not UA-Ready. Email validation based on Django validator; URL validation based on RegEx.
email_validator	1178	UA-Ready. IDNA2008.
pyicu	243	UA-Ready. IDNA2008.
idna_ssl	10	UA-Ready. IDNA2008.

Dataset: 450 million dependency files; occurrence in entire dataset: 70,813; approx. 37%.

<https://uasg.tech/wp-content/uploads/documents/UASG033-en-digital.pdf>



Very easy with Go

Use IDNA package



<> Documentation

Overview

Package idna implements IDNA2008 using the compatibility processing defined by UTS (Unicode Technical Standard) #46, which defines a standard to deal with the transition from IDNA2003.

IDNA2008 (Internationalized Domain Names for Applications), is defined in RFC 5890, [RFC 5891](#), [RFC 5892](#), [RFC 5893](#) and [RFC 5894](#). UTS #46 is defined in <https://www.unicode.org/reports/tr46>. See <https://unicode.org/cldr/utility/idna.jsp> for a visualization of the differences between these two standards.

<https://pkg.go.dev/golang.org/x/net/idna>



Android application

Use class IDNA

IDNA 

Added in API level 24

[Kotlin](#) | [Java](#)

```
public abstract class IDNA
extends Object

java.lang.Object
↳ android.icu.text.IDNA
```

Abstract base class for IDNA processing. See <http://www.unicode.org/reports/tr46/> and <http://www.ietf.org/rfc/rfc3490.txt>

The IDNA class is not intended for public subclassing.

The non-static methods implement UTS #46 and IDNA2008. IDNA2008 is implemented according to UTS #46, see `getUTS46Instance()`.

<https://developer.android.com/reference/android/icu/text/IDNA>



Normalized before process



ไทยร่วมใจ.com

xn--82c3a4adf1rc3b.com



Normalized before process

With python + idna

```
import idna

before = 'ไทยร่วมใจ.com'
normalized = idna.encode(before)
displayed = idna.decode(normalized)
```

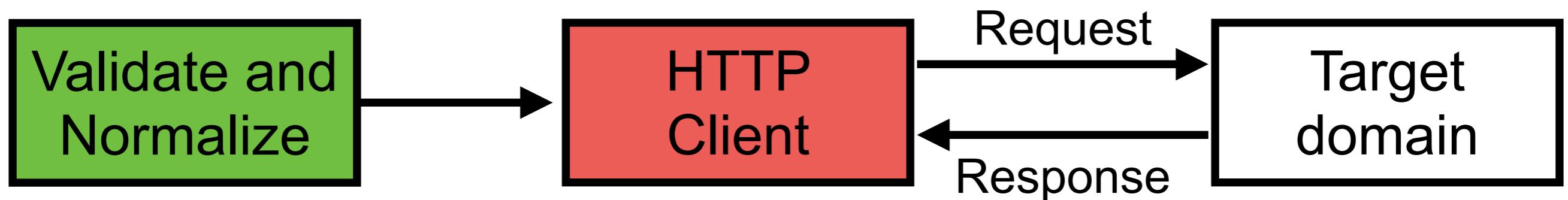
<https://pypi.org/project/idna/>



Making an HTTP request



Making an HTTP request



Coding with Java

HttpURLConnection

Apache HTTP Client

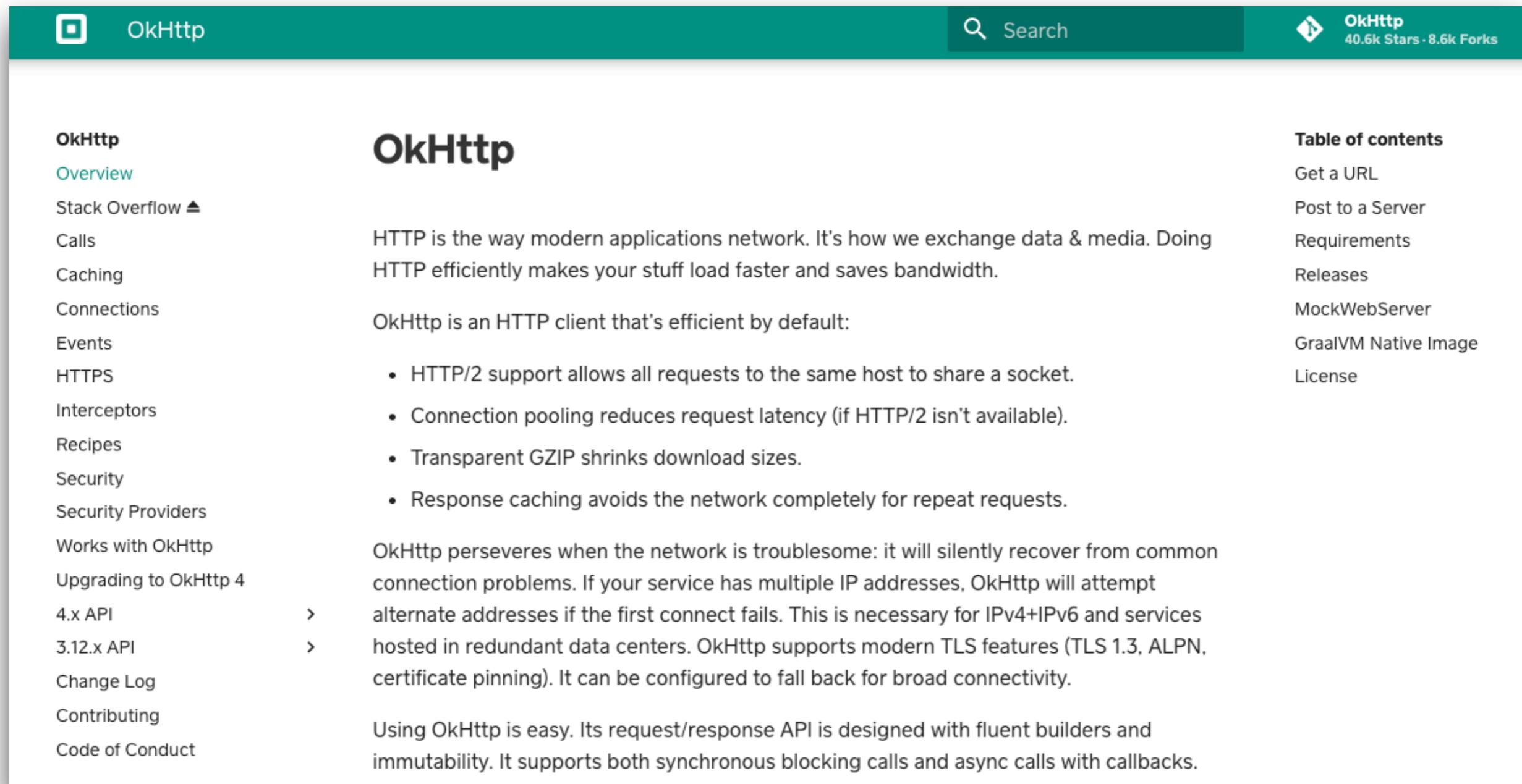
OkHTTP

Java 11 HTTP Client (not prepare UTF8)

Google Java HTTP Client



Working with OkHTTP



The screenshot shows the OkHttp GitHub repository page. The header includes the OkHttp logo, a search bar, and repository statistics (40.6k Stars · 8.6k Forks). The left sidebar lists navigation links: Overview, Stack Overflow ▲, Calls, Caching, Connections, Events, HTTPS, Interceptors, Recipes, Security, Security Providers, Works with OkHttp, Upgrading to OkHttp 4, 4.x API, 3.12.x API, Change Log, Contributing, and Code of Conduct. The main content area features a large title "OkHttp" and several sections of text describing the library's features and capabilities.

OkHttp

[Overview](#)

[Stack Overflow ▲](#)

Calls

Caching

Connections

Events

HTTPS

Interceptors

Recipes

Security

Security Providers

Works with OkHttp

Upgrading to OkHttp 4

4.x API

3.12.x API

Change Log

Contributing

Code of Conduct

OkHttp

HTTP is the way modern applications network. It's how we exchange data & media. Doing HTTP efficiently makes your stuff load faster and saves bandwidth.

OkHttp is an HTTP client that's efficient by default:

- HTTP/2 support allows all requests to the same host to share a socket.
- Connection pooling reduces request latency (if HTTP/2 isn't available).
- Transparent GZIP shrinks download sizes.
- Response caching avoids the network completely for repeat requests.

OkHttp perseveres when the network is troublesome: it will silently recover from common connection problems. If your service has multiple IP addresses, OkHttp will attempt alternate addresses if the first connect fails. This is necessary for IPv4+IPv6 and services hosted in redundant data centers. OkHttp supports modern TLS features (TLS 1.3, ALPN, certificate pinning). It can be configured to fall back for broad connectivity.

Using OkHttp is easy. Its request/response API is designed with fluent builders and immutability. It supports both synchronous blocking calls and async calls with callbacks.

Table of contents

- Get a URL
- Post to a Server
- Requirements
- Releases
- MockWebServer
- GraalVM Native Image
- License

<https://square.github.io/okhttp/>



Working with OkHTTP

```
public boolean callByDomain02(String domainName) {  
    OkHttpClient httpClient = new OkHttpClient();  
    Request request = new Request.Builder().url("http://" + domainName).build();  
    try {  
        Response response = httpClient.newCall(request).execute();  
        return true;  
    } catch (IOException e) {  
        return false;  
    }  
}
```

<https://square.github.io/okhttp/>



Working with Email address (EAI)



Validate email format

Regular expression
Use libraries or framework



Regular Expression

From OWASP validation Regex Repository

[^a-zA-Z0-9_+&*-]+(?:\.[a-zA-Z0-9_+&*-]+)*@[?:[a-zA-Z0-9-]+\.\.]+[a-zA-Z]{2,7}\$]

https://owasp.org/www-community/OWASP_Validation_Regex_Repository



Regular Expression

Not support EAI

Not support ASCII TLD longer than 7

Not support U-labels in IDN TLD



Code Example

```
public class EmailValidation {  
  
    public boolean isValidWithRegex(String email) {  
        return email.matches(".*@[+.*\\\\.]*");  
    }  
  
    public boolean isValidWithRegex02(String email) {  
        String regex = "^[a-zA-Z0-9_+&*-]+(?:\\.[a-zA-Z0-9_+&*-]+)*@[a-zA-Z0-9_+&*-]+\\.[a-zA-Z0-9_-]{2,}$";  
        return email.matches(regex);  
    }  
}
```



Use Libraries

Jakarta Mail

Apache Commons Validator

EmailValidator4J



Apache Commons Validator

```
public boolean isValidWithApacheCommons(String email) {  
    EmailValidator emailValidator = EmailValidator.getInstance();  
    return emailValidator.isValid(email);  
}
```



Send Email ...



Send email to SMTP server



Sending Email

Jakarta Mail

Simple Java Mail (Jakarta Mail wrapper)



Example

```
public boolean send(String email) {  
  
    // 1. Validate email  
    if(!isValidEmail(email)) {  
        throw new RuntimeException("Email invalid");  
    }  
  
    // 2. Validate Domain name from email  
    String domain = email.substring(email.lastIndexOf("@") + 1);  
    ValidateDomain validateDomain = new ValidateDomain();  
    if(!validateDomain.isValidWithICU4j(domain)) {  
        throw new RuntimeException("Domain invalid");  
    }  
  
    // 3. Send email  
    return true;  
}
```



Working with Spring Boot



TODO

Size of domain name to max (255)
Use string type

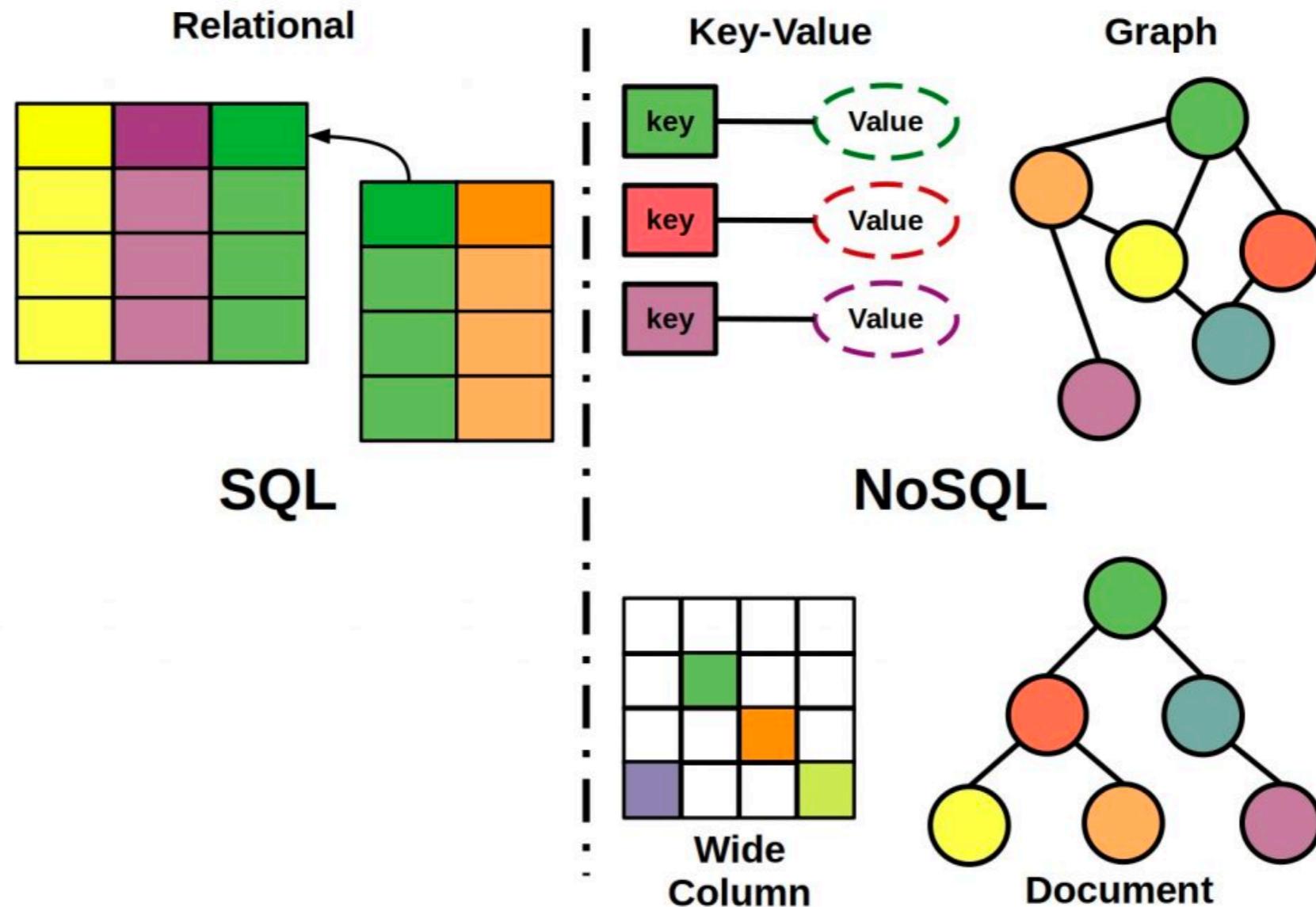


Store data in Database

Domain and email



SQL and NoSQL

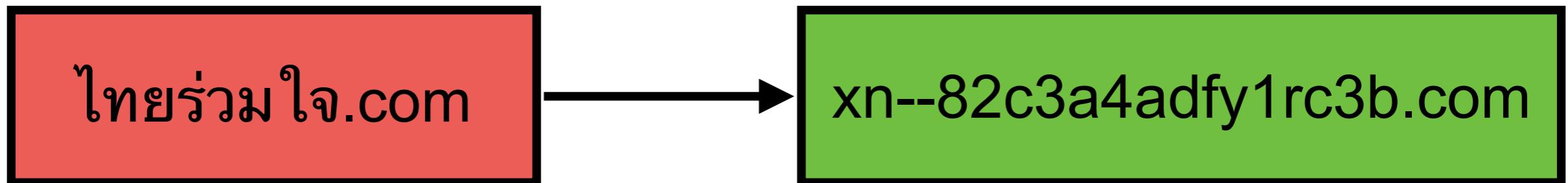


SQL

Size of domain name to max (255)

Use string type

Convert domain to A-label (normalised)



Summary



Suggestions

Validation for IDA, EAI
UA input is complicated

prefer basic validation



Suggestions

Never rely on a static list of TLDs.

Use String type to hold data

Convert domain to A-label when passing to library

Use a UA-conformant library/framework



Suggestions

Make sure to do normalization of input
before store, compare and process



Suggestions

Do unit and system testing of UA

<https://uasg.tech/wp-content/uploads/documents/UASG004-en-digital.pdf>



Thank you

Q/A

