普遍接受性概论

马克·斯万卡雷克 (Mark Svancarek)、
路易莎·维拉 (Luisa Villa)

关于本文档

目的

互联网技术，包括其命名组件，正处在不断发展和变化过程中。近年来，ICANN 发布了大量 ASCII 格式的新 TLD 以及 IDN 顶级域，例如 .nyc、.hsbc、.eco 和 .ストア 等等。但是，各方未能及时就命名方式的改变做出响应。许多应用和服务未进行更新，无法恰当处理这些新 TLD，从而导致用户体验不佳。例如：

- 有效电子邮件地址不被接受
- 浏览器地址栏中的域名被错误地当作搜索词处理

应用和服务未进行更新
无法恰当处理新 TLD，
从而导致用户体验不佳。

除非软件能够识别并恰当处理新域名（即达到“**普遍接受性**”状态），否则将无法为互联网用户提供一致、积极的体验。因此，本文档将全面介绍普遍接受性，以帮助开发支持普遍接受性的软件。

目标读者

- 软件开发者
- 首席技术官 (CTO)
- 普通技术社群

文档结构

第 1 部分　**普遍接受性基本概念**，如什么是域名和域名系统、ASCII 和统一域名编码 (Unicode)、国际化域名编码 (Punycode)、国际化电子邮件地址以及其它基本概念。

第 2 部分　**普遍接受性的五个标准**以及针对每项标准的**最佳实践**。此外，这部分内容还包括普遍接受性的**用户场景**和**不合规做法**、技术要求和当前面临的挑战。

第 3 部分　一些**进阶主题**，如从右至左书写的文本、Bidi 算法、标准化和大小写转换 (Case Folding)。

第 4 部分　提供了**术语表**和一些有用的**在线资源**。

想要了解更多信息？

UASG 和社群将随时为软件开发者和实施者提供所需建议。

- **联系我们**，发表您对这一主题的看法并提出建议：info@uasg.tech
- **参加普遍接受性**讨论：http://tinyurl.com/ua-discuss
- 如需**了解更多信息**，请访问：http://www.icann.org/universalacceptance

目录

简介

域名国际化发展简史

20 世纪 70 年代，可用于注册域名的字符仅限于一部分 **ASCII** 字符（字母 a-z、数字 0-9 和连字符“-”）。自 1985 年最早注册 .com 域名 symbolics.com 以来，域名的数量不断增多，特征也越来越多样化，以体现全球日益将互联网用作公共资源的需求。如今，绝大多数互联网用户都不讲英语。但是，英语是互联网使用的主导语言。为帮助实现互联网的国际化，2003 年，互联网工程任务组 (IETF) 开始发布标准，为通过转换机制来部署**国际化域名 (IDN)** 提供技术指南，以支持采用各个不同地区当地文字书写的非 ASCII 格式域名（例如，普遍接受-测试.世界、ua-test.世界等）的识别和处理。

普遍接受性是指所有有效域名和电子邮件地址能在所有支持互联网连接的应用、设备和系统中正确、一致地获得接受、确认、存储、处理和显示。

2009 年 10 月，互联网名称与数字地址分配机构 (ICANN) 董事会批准了引入新 IDN 国家和地区顶级域 (ccTLD) 的流程，并于 2010 年 5 月将第一个 IDN ccTLD 添加到根区中。2011 年 6 月，董事会批准并授权启动新通用顶级域 (gTLD) 项目，其中包括了新 ASCII 以及 IDN TLD。此项目中的第一批 TLD 已于 2013 年添加到根区中。IDN ccTLD 和新 TLD 的引入大大加快了在根区中添加 TLD 的步伐。

在 IETF 发布其 IDN 相关指南 10 年之后，加之 ICANN 新 TLD 项目的推动作用，目前已经开放注册了一千多个新 TLD。但是，尽管做出了所有这些努力，许多软件和应用仍然没有为普遍接受性做好准备。这给互联网用户（包括那些所使用的文字中包含非 ASCII 字符的用户）带来了问题。

实现普遍接受性的必要性

为了跟上新 TLD 的发展步伐，必须在开发新软件的同时，更新旧有的软件和应用。成功满足新 TLD 的识别和处理需求即表明具有**普遍接受性**。

普遍接受性是指所有有效域名和电子邮件地址能在所有支持互联网连接的应用、设备和系统中正确、一致地获得**接受**、**确认**、**存储**、**处理**和**显示**。换言之，即每一个有效网址都能被解析到所需的目标网站，每一个有效的电子邮件地址都能将邮件发送到预期的目的地。由于域名领域的变化非常迅速，许多系统无法识别或恰当处理新域名，这主要是由于这些新域名采用非 ASCII 格式，或软件不认可新发布的 TLD，或 TLD 长度各异所致。采用这些新扩展名的电子邮件地址也出现了这种情况。

普遍接受性指导小组 (UASG) 由 ICANN 组建，是一个由社群领导、覆盖整个行业的工作组，其任务是提高认识，确定并解决与域名普遍接受性有关的问题，帮助为全球互联网用户提供一致、积极的体验。

第 1 部分：普遍接受性基本概念

这一部分将介绍一些基本术语和概念，为理解本文后面讲述的更深入的问题做准备。

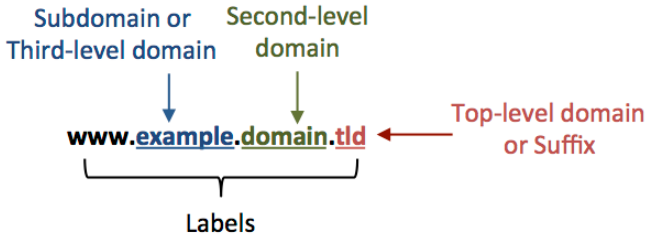
域名

域名是指供互联网上的计算机和网络用作技术标识符的易于记忆的带点文本字符串。例如：

```
www.domain.tld
```

域名的解读：

- 每个点表示域名系统 (DNS) 结构中的一个**层级**。
- 顶级域 (TLD) 常被称为域名末尾的**后缀**。
- 点之间的单个词或字符称为**标签**。对那些**从左至右 (LTR)** 书写的语言或文字来说，[1]最右侧的标签表示顶级域。
- 倒数第二个标签表示**二级域**。
- 二级域前面的任何标签均被视为二级域的**子域**（有时也称为**三级域**）。



域名系统 (DNS)

互联网上的每个软硬件资源都分配有一个供互联网协议 (IP) 使用的地址。由于 IP 地址不便于记忆，因此，DNS 在 IP 地址与人类可读的域名之间建立了映射。在互联网上，提供公共 DNS 的服务器一般都使用的是简单好记的地址。

顶级域 (TLD)

人类可读的域名由**注册管理机构**负责管理。注册的域名由多个分别代表不同域名层级的文本字符串构成，每个字符串用“.”字符分隔开来。在 LTR 文字中，最右侧的域名层级为顶级域 (TLD)。一些 TLD 会被分配给特定国家或地区。这些域名称为**国家和地区顶级域 (ccTLD)**。

[1] 本文后面部分将讨论从右至左 (RTL) 书写的语言或文字。

通用顶级域 (gTLD)

从 2013 年开始，ICANN（负责创建和维护 TLD 分配的组织）批准创建了大量新 TLD。这些新 TLD 可能代表品牌、利益社群、地理社群（城市、地区）以及更加通用的概念。所有这些新 TLD 统称为通用顶级域 (gTLD)。

普通 TLD 示例	ccTLD 示例	新 gTLD 示例
.com	中国 = .cn	.app
.gov	德国 = .de	.lawyer
.info	美国 = .us	.shopping
.org		.panasonic
		.osaka

字符集和文字

语言通过书写系统来书写。大多数书写系统都使用一种文字，即一组用于以书面形式表示一种或多种语言的图形字符。少数书写系统同时使用多种文字。这些字符或文字可以被人类识别。但是，它们不能用于计算机。相反，计算机需要以某种方式对文字进行编码，以便进行处理（例如，解析网址）。这种机制称为**字符映射**或**编码字符集 (CCS)**，或称为**代码页**。[2]字符映射旨在将字符与特定数字关联起来。长期以来，人们出于各种目的创建了许多不同的代码页，但在本文档中，我们主要介绍两种代码：ASCII 和 Unicode。

ASCII 和 Unicode

在上文的 TLD 示例中，所有文本字符串均使用拉丁字符集表示。该字符集属于美国信息交换标准码（ASCII 或 US-ASCII）字符编码体系。ASCII 以英语为基础进行编码，是一种比较早的编码体系。由于某些历史原因，它成为了互联网采用的标准字符编码体系。ASCII 的每个字符仅为 7 位，因此只能支持 128 个字符，而且并非所有字符都可用在域名中。可用于域名的字符仅限于字母 A-Z、数字 0-9 和连字符“-”。

ASCII - ISO 8859-1 (Latin-1) 表[3]

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
20		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	

[2] 这些术语存在一些与普遍接受性没有直接关系的微妙之处。如果有兴趣了解更多与术语相关的信息，可以尝试访问：https://tools.ietf.org/html/rfc6365

[3] 资料来源：加利福尼亚州立大学。1997 年。*ASCII - ISO 8859-1 (Latin-1) 表（包含 HTML 实体名称）*。+ http://web.calstatela.edu/faculty/jchen13/Docs/CS120/Lectures/ASCIITable_with_HTML_Entity_Names.htm

由于大多数书写系统不使用拉丁字符集，因此人们采用了其他替代编码体系。Unicode 也称为**统一域名编码字符集 (UCS)**，它能够对一百多万个字符进行编码。这其中的每个 Unicode 字符称为一个**代码点**。最常用的 Unicode 实现方式称为**统一域名编码字符集 8 位转换格式 (UTF-8)**。

要查看所有 Unicode 字符的代码表，请访问：http://unicode.org/charts

国际化域名 (IDN) 和国际化域名编码 (Punycode)

使用 Unicode 可以对包含非 ASCII 字符的域名进行编码。如上文所述，这些使用非 ASCII 字符的域名又称为国际化域名 (IDN)。[4]域名的国际化可以发生在任何层级 — 不只是 TLD，其它标签也是如此。

由于以前 DNS 本身只使用 ASCII，[5]因此需要创建其它编码体系来实现非 ASCII Unicode 代码点与 ASCII 字符串之间的相互转换。实施这种 Unicode 到 ASCII 编码转换的算法称为 **Punycode**；其输出字符串称为 **A-标签**。A-标签可以与普通 ASCII 标签区分开来，因为它们总是以下面四个字符开头：

- **xn--**

这些字符称为 **ACE 前缀**。[6]

Punycode 转换是可逆的，它既可以从 Unicode 转换为 A-标签，也可以从 A-标签转换为 Unicode（称为 U-标签）。

根据 RFC 的定义[7]，Punycode 算法的唯一用途是用来表示国际化域名。但是，除了实现 Unicode 编码转换以外，一些开发者还选择将 Punycode 应用于其它场景。

IDN 示例（虚构）

example.みんな	（Punycode 编码 = example.xn--q9jyb4c）
大坂.info	（Punycode 编码 = xn--uesx7b.info）
みんな.大坂	（Punycode 编码 = xn--q9jyb4c.xn--uesx7b）

有关详细信息，请查看 IDN FAQ：http://unicode.org/faq/idn.html

电子邮件

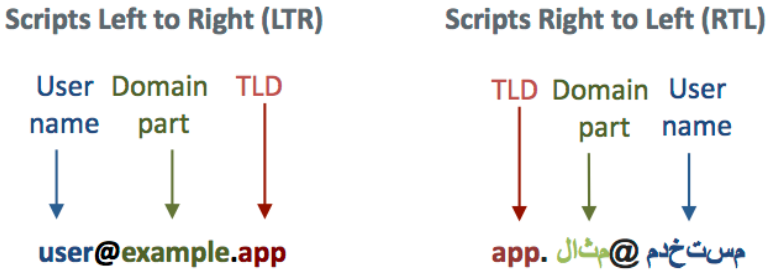
地址和国际化电子邮件地址 (EAI)

电子邮件地址包含两个部分：

1. 本地部分（用户名，位于“@”字符之前）
2. 域（位于“@”字符之后）

[4] 请注意，不是所有非 ASCII 字符都是 IDN。
[5] 有关最新动态，请访问：http://tools.ietf.org/html/rfc6055#section-3
[6] ASCII 兼容编码 (ACE) 前缀用于将 Punycode 编码标签与普通 ASCII 标签区分开来。
[7] RFC：征求意见稿。请参见本文第 4 部分的术语表了解更多信息。

域部分可以包含任何 TLD，包括新 TLD。这两个部分都可以为 Unicode U-标签。S



注：下面将介绍另一种格式，即 IDN 形式的电子邮件地址。

包含 IDN 的电子邮件地址示例（虚构）

user@example.みんな	（使用国际化 TLD）
user@大坂.info	（使用国际化二级域）
用戶@example.lawyer	（使用国际化用户名和新 gTLD）

国际化电子邮件地址 (EAI) 要求在电子邮件地址的所有部分使用 Unicode。上面的每个示例都可以表示为 EAI，并且这是首选格式。

动态链接生成（链接化）

现代软件（如常用的文字处理或电子表格应用）有时允许用户直接输入一个看似网址、电子邮件地址或网络路径的字符串，以此来创建超链接。例如，如果应用将“www.”视为特殊前缀，或将“.org”视为特殊后缀，则在电子邮件中输入“www.icann.org”后会自动创建一个指向 http://www.icann.org 的可点击链接。

链接化应一致处理所有符合规则的网址、电子邮件地址或网络路径。

第 2 部分：普遍接受性现状

普遍接受性的五个标准

如上文所述，普遍接受性是指，所有有效域名和电子邮件地址，能在所有支持互联网连接的应用、设备和系统中，正确、一致地获得**接受**、**确认**、**存储**、**处理**和**显示**。下面将详细介绍这五个标准。

1. 接受[8]	***接受*是指电子邮件地址或域名作为字符串，被各类软件应用或在线服务的用户界面、文件或 API（应用程序接口）所接收的过程。** 应用和服务应允许： • 将域名和电子邮件地址输入到用户界面，以及/或者 • 通过 API 接收来自其它应用和服务的域名和电子邮件地址
2. 确认[9]	***确认*是指应用或在线服务将电子邮件地址或域名作为有效字符串进行接收或发送。** 确认旨在确保输入的信息为有效信息，或至少肯定不是无效信息。换言之，确认是为了确保给定信息的句法正确性。 对于域名和电子邮件地址，许多程序员一直使用某些验证方法来加以确认（例如检查 TLD 的字符数是否“正确”，或字符是否来自 ASCII 字符集）。但是，由于近年来互联网发生了以下变化，这些验证方法已不再适用： • 域名和电子邮件地址现在可以包含 Unicode（非 ASCII）字符 • TLD 列表不断增长 • TLD 最多可以长达 63 个字符
3. 存储	***存储*是指将电子邮件地址或域名以字符串形式存储在数据库中，或存储在软件应用或在线服务所使用的文件中。** 应用和服务可能需要长期和/或暂时存储域名和电子邮件地址。无论数据的使用期有多长，都必须： • 以 RFC 规定的格式存储，或 • 采用可以轻松转换为 RFC 规定格式和从 RFC 规定格式恢复（较少需要）的替代格式存储 虽然 RFC 要求使用 UTF-8，但遗留代码可能采用其它格式。请参阅下文的“最佳实践”部分。

[8] 在本文中，接受和确认被视为不同的功能。实际上，这些功能之间可能会出现重叠。

[9] 在本文中，接受、处理和确认被视为不同的功能。实际上，这些功能之间可能会出现重叠。

4. 处理[10]	***处理*是指电子邮件地址或域名被应用或服务用于执行某一项任务（例如搜索或对列表排序）、或者转换为替代格式（例如将 ASCII 存储为 Unicode）的过程。** 处理意味着在某项功能中使用域名和电子邮件字符串。处理过程中可能会同时执行确认操作。处理域名和电子邮件地址的方式没有限制（示例：“通过在 .nz ccTLD 中搜索以识别所有位于新西兰的用户”；“通过搜索 user@example.pharmacy 电子邮件地址来识别所有药剂师”；“识别可能会过滤不符合其政策的 DNS 请求的防火墙”）。
5. 显示	***显示*是指在用户界面中提供电子邮件地址或域名的过程。** 当域名和电子邮件地址使用的文字受到基础操作系统支持，且字符串以 Unicode 格式存储时，通常可直接显示。如果不满足这些条件，可能需要根据特定应用进行某些转换。

用户场景

上述示例和定义可能会让人认为普遍接受性只与计算机系统和在线服务有关。但是，实际上，它也与使用这些系统和服务的人员有关。

下面提供了一些需要普遍接受性的活动示例：

注册新 TLD	某企业采用“brand”TLD，通过提供 customername @example.brand 格式的电子邮件地址，为其客户提供差异化的客户体验。 这时，普遍接受性是指： • 网络应用接受这些新的“@example.brand”电子邮件地址，将它们视为与 .com、.net、.org 等 TLD 一样有效。
访问 gTLD	用户通过在浏览器中输入地址或单击文档中的链接，访问域名中包含新 TLD 的网站。 这时，普遍接受性是指： • 即使此 TLD 为新 TLD，但用户希望使用的任何浏览器都会以本地格式显示该网址，并如用户的期望成功访问该网站。浏览器不会向用户显示 Punycode 文本，除非这样做在一定程度上对用户有利。
使用包含新 gTLD 的电子邮件地址作为网络身份	用户获得一个域部分使用新 gTLD 的电子邮件地址，并以此电子邮件地址作为网络身份来访问其银行和航空公司的会员账户。 这时，普遍接受性是指： • 即使电子邮件地址中使用的域为新域，但银行或航空公司网站仍然接受该地址，如同它是 .biz 或 .eu 之类的既有 TLD 一样。

[10] 在本文中，处理和确认被视为不同的功能。实际上，这些功能之间可能会出现重叠。

访问 IDN	用户通过在浏览器中输入地址或单击文档中的链接来访问 IDN URL。 这时，普遍接受性是指： • 即使域名中包含与用户计算机的语言设置不同的字符，但用户希望使用的任何浏览器都会按预期显示该网址，并成功访问相应网站。
电子邮件使用国际化电子邮件地址	用户获得多个电子邮件地址，其中一些为国际化地址（如 info@普遍接受-测试.世界）。 这时，普遍接受性是指： • 用户可以使用任何电子邮件地址，通过任何电子邮件客户端发送和接收邮件。
使用国际化电子邮件地址作为网络身份	用户获得一个 EAI 电子邮件地址，并以此电子邮件地址作为网络身份来访问其银行和航空公司的会员账户。 这时，普遍接受性是指： • 银行或航空公司的网站接受该 EAI 身份，与接受其它电子邮件身份一样。
在应用中动态创建超链接	用户在文档或电子邮件中输入网址。 这时，普遍接受性是指： • 即使地址为 EAI 或包含新 TLD，应用程序仍应根据相同的规则自动生成超链接。
开发应用程序	开发者编写将用于访问网络资源的应用。 这时，普遍接受性是指： • 开发者使用的工具包含支持 Unicode、IDN 和 EAI 的库，可实现普遍接受性。

不符合普遍接受性的做法

以下做法被视为**错误做法**：

✖	向用户显示 Punycode 文本，但这样做对用户并无实际帮助。 例如，显示 U-标签与 A-标签之间的映射。
✖	在注册新电子邮件地址或新托管域时要求用户输入 Punycode 文本。
✖	使用过时标准或非权威性在线域名资源验证域名或电子邮件地址的语法。
✖	即使会定期更新新 TLD，但仍然使用过时的 TLD 列表。
✖	向用户显示内部使用的 Punycode 文本。 例如，在回复 EAI 用户时将 EAI 转换为 IDN 形式的电子邮件地址。
✖	由于应用程序无法识别某些域名，而将这些域名作为搜索关键词处理。
✖	将垃圾邮件拦截器设置为自动拦截整个 TLD。

实现 UA 的技术要求

高层次要求

支持普遍接受性 (UA) 的应用程序或服务应：

1. **支持任意长度或使用任何字符集的域名。**

   请参见 RFC 5892。

2. **允许在域名和电子邮件地址中使用多个有效字符集。**

   也就是说，允许使用 Unicode 代码点。

3. **能够正确显示 Unicode 字符串中的所有代码点。**

   请参见 RFC 3490。

4. **能够正确显示从右至左 (RTL) 书写的字符串，如阿拉伯文和希伯来文字符串。**

   有关 RTL 文字的信息，请参见 RFC 5893。

5. **能够在应用与服务之间以支持 Unicode 并且可与 UTF-8 相互转换的格式传输数据。**

   有关 UTF-8 的信息，请参见 RFC 3629。

6. **提供支持 Unicode 和 UTF-8 的公共 API。**

7. **提供支持 Unicode 和 UTF-8 的私有 API。**

   私有 API 只适用于由同一供应商进行服务间调用。

8. **以支持 Unicode 并且可与 UTF-8 相互转换的格式存储用户数据。**

   此类转换应仅对产品/服务所有者可见。

9. **支持 ICANN TLD 权威列表和服务于社群的公共后缀列表中的所有域名字符串，而不论其长度或使用的字符集。**

   请参见 https://newgtlds.icann.org/en/program-status/delegated-strings。

10. **可以向收件人发送或从收件人处接收各类电子邮件，而不论电子邮件地址使用的域名或字符集。**

    请参见 RFC 6530。

11. **像处理 Punycode 地址（IDN 电子邮件格式）一样处理 EAI 地址。**

开发者注意事项

由于许多现有的软件系统对域名和电子邮件地址做出了一些编码硬性规定，因此可能需要更改代码才能识别 IDN 和新 TLD。本部分将说明开发者应如何更改代码，以便所有新 TLD 都可实现普遍接受性。

实现普遍接受性的指导原则：伯斯塔尔法则 (Postel's Law)

在 RFC 793 中，Jon Postel 提出了**鲁棒性原则**（现在称为**伯斯塔尔法则**），将其作为实施新 TCP 的指导方针。在计算领域，鲁棒性原则是软件的总体设计原则：

> "发送时要保守，接收时要开放。"

也就是说，对您发送的内容持谨慎态度，对您接收的内容持宽容态度。在处理当前在生态系统内实施普遍接受性时出现的异常行为时，这也是一个不错的方法。

开发和更新软件以实现 UA 的最佳实践

接受	
✔	始终提供相应的 Unicode 文本。 输入时，允许但不强制要求用户输入 ASCII 兼容编码（或"Punycode"）文本来代替对应的 Unicode 文本。 输出时，应该默认显示 Unicode 文本，或只在对用户有利时才显示 Punycode 文本。
!	**不要**生成 IDN 形式的电子邮件地址，但如果其他人的软件提交了这类地址，**应该**能够处理它们。
✔	任何要求用户输入域名或电子邮件地址的用户界面，必须同时支持 Unicode、最多包含 63 个字符的标签，以及最多包含 253 个字符的域名字符串。 • 请参见 RFC 1035。

确认	
✔	仅执行所需的最低程度的确认操作。 **仅在应用或服务执行域名相关操作时才进行必要的确认。**这是确保所有有效域名都被系统接受的最可靠方法。
✔	应认识到，有些输入可能并不与互联网当前使用的域名或电子邮件地址相同，但在句法上仍是正确的。

!	如果必须进行确认，请考虑以下因素：
	
	• 根据权威性表格验证域名的 TLD 部分。下面提供了一些可供使用的权威性表格示例：
	  ○ http://www.internic.net/domain/root.zone
	  ○ http://www.dns.icann.org/services/authoritative-dns/index.html
	  ○ http://data.iana.org/TLD/tlds-alpha-by-domain.txt
	
	另见：https://www.icann.org/en/system/files/files/sac-070-en.pdf
	
	• 根据 DNS 查询域名
	  ○ 考虑使用 GETDNS API (http://getdnsapi.net/)
	• 要求重复输入电子邮件地址以防止输入错误
	• 确认标签中的字符，但仅限于确定 U-标签不含"禁用"代码点或者在其 Unicode 版本

	中未赋值的代码点 ○ 请参见 RFC 5892 • 将标签确认的范围限制为 RFC 中定义的少数全标签规则 ○ 请参见 RFC 5894 • 如果一个看似域名的字符串包含阿拉伯文句号字符“-”(U+06D4) 或表意文字句号字符“。”(U+002E)，则将其转换为句点“.”(U+002E)。 • 确保产品或功能正确处理数字 ○ 例如：应将 ASCII 数字和亚洲表意数字均视为数字

存储

✔	应用和服务应支持相应的 Unicode 标准。
✔	如有可能，应以 UTF-8（Unicode 转换格式）存储信息。 某些系统可能要求也支持 UTF-16，但一般情况下都只要求支持 UTF-8。应避免使用 UTF-7 和 UTF-32。
!	存储时，在将 A-标签 (Punycode) 转换为 U-标签（反之亦然）之前，应考虑所有端到端情景。 可以只要求在文件或数据库中保留 U-标签，因为这将简化搜索和排序。但是，在与旧有的未启用 Unicode 的应用和服务交互操作时，可能需要进行必要的转换，在此时应考虑同时以两种格式存储并创建索引。
✔	在存储时对电子邮件地址和域名做出清楚标记，以便于使用。 如果电子邮件地址和域名被保存在某些文档的“作者”字段或日志文件的“联系信息”中，可能会导致域名和邮件地址信息丢失。
✔	如果*不统一*存储为 Unicode 格式，则必须能够匹配多种格式的字符串。 例如，搜索 example.みんな应该可以同时找到 example.xn--q9jyb4c。

处理

✔	确保所有服务器响应将内容类型指定为Unicode。
✔	请在 Web 服务器 http 头文件以及 Web 文件中将编码直接指定 为Unicode。 • 每个 Web 文件应包含 UTF-8 字符集 • 必须确保在每个响应中指定了编码
!	在处理过程中，在将 A-标签 (Punycode) 转换为 U-标签（反之亦然）之前，应考虑所有端到端情景。 可以只要求在文件或数据库中保留 U-标签，因为这将简化搜索和排序。但是，在与旧有的未启用 Unicode 的应用和服务交互操作时，可能需要进行必要的转换，在此时应考虑同时以两种格式存储。

✔	确保功能上根据区域设置/语言规范来处理排序次序、搜索和排序规则，并能够处理多语言搜索和排序问题。
✖	不要对域名使用 URL 编码： • example.みんな 为正确域名 • example.%E3%81%BF%E3%82%93%E3%81%AA 为错误域名
✔	由于 Unicode 标准在持续扩展，应对在开发应用或服务时未定义的代码点进行检查，确保它们不会使用户体验出现“中断”。 基础操作系统中缺少字体可能会导致无法显示某些字符（通常用“□”字符来表示这些字符），但是这种情况应该不会造成致命崩溃。
✔	使用支持 Unicode编码 的 API。
✔	针对 IDN，使用最新国际化域名应用 (IDNA) 协议和表格： • RFC 5891 • RFC 5892
✔	尽量以 UTF-8 格式进行处理。
✔	保持应用和服务器/服务的同时升级与一致。 如果服务器为 Unicode 格式而客户端为非 Unicode 格式（或者反之），则每次在服务器与客户端之间传输数据时，都需要逐一进行数据编码转换。
✔	执行代码审核，避免缓冲区溢出攻击。 进行字符转换时，文本字符串的长度可能会显著增多或减少，而影响缓冲区。

显示	
✔	显示所有受基础操作系统支持的 Unicode 代码点。 即便应用使用自己携带的字体集，也应为操作系统中的可用字体集提供全面支持。
✔	开发应用或服务时，考虑所有受支持的语言并确保操作系统和应用涵盖了这些语言。
✔	显示前，将非 Unicode 数据转换为 Unicode 数据。 例如，最终用户应看到“example.みんな”而不是“example.xn--q9jyb4c”。（这一转换是先为普遍接受性做好预操作准备再予以处理的一个例子）。
✔	默认显示 Unicode 数据。 仅在对用户有利时才显示 Punycode 文本。
!	请注意，采用混合文字书写的地址将变得更加常见。 • 一些 Unicode 字符肉眼看起来很相似，但对计算机而言却存在不同

	• 不要假定混合文字字符串旨在用于实现网络钓鱼等恶意目的 • 如果用户界面希望字符串引起用户注意，请确保这样做不会损害非拉丁文用户的利益 有关 Unicode 安全注意事项的更多信息，请访问：http://unicode.org/reports/tr36
✔	使用 Unicode IDNA 兼容性处理，以满足用户的预期。 如需了解更多信息，请访问：http://unicode.org/reports/tr46
✔	了解域名中的未赋值字符和无效字符。 • 请参见 RFC 5892

统一域名编码 (Unicode)	
✔	使用支持Unicode编码 的 API。
✖	不建议自行构建 API 以用于： • 字符串格式转换 • 确定哪些文字包含字符串 • 确定字符串是否包含多种文字 • Unicode 规范化/分解
✖	请勿使用 UTF-7 或 UTF-32。 • 通常情况下，应用不会将 UTF-7 作为本地表示方式，因为它非常难以处理。尽管它相比于采用 Quoted-Printable 或 Base64 编码的 UTF-8 组合在大小上具有优势，但互联网邮件协会并不建议使用 UTF-7。 • UTF-32 的主要缺点在于它的每个代码点使用四个字节，导致无法有效利用空间。特别地，非 BMP 字符在大多数文本中都极为少见[需要引证]，以致于它们通常被认为不存在占用空间大小的问题，这使得 UTF-32 的占用空间高达 UTF-16 的两倍，UTF-8 的四倍。
✔	在 Cookie 中使用 Unicode，以便应用正确读取。
✔	使用 IDNA 2008 协议和表格文档： • RFC 5891 • RFC 5892
✖	请勿使用 IDNA 2003；在绝大部分系统中，它已被 IDNA 2008 所取代。
✖	不要机械地认为外部 API 会占用已经进行 NFKC[11] 转换的数据。
!	与各版本的 IDNA 和 Unicode 表保持一致 例如，除非应用确实执行了表文档 (RFC 5892) 中的分类规则，否则它的 IDNA 表就必须衍生于在

[11] **NFKC**（*兼容等价合成*）：按兼容性分解字符，然后按规范等价重组。请参见：http://unicode.org/reports/tr15

	系统中受到更广泛支持的 Unicode版本。与注册一样，这些表不需要体现为最新的 Unicode版本，但它们必须保持一致。
!	确认标签中的字符，但仅限于确定 U-标签不含“禁用”[12]代码点或者在这个 Unicode 版本中未赋值的代码点。
✔	将标签确认的范围限制为“少数全标签规则”： • 无前导组合字符 • 如果显示从右至左书写的字符，则满足双向条件 • 测试任何与连接符（以及更常见的 CONTEXTJ[13] 字符）关联的上下文规则
!	除非明确要求（如在某些 Windows API 中），否则请勿使用 UTF-16。 请注意，在使用 UTF-16 时，16 位可能仅包含从 0x0 到 0xFFFF 范围内的字符，要存储此范围以外的值（0x10000 到 0x10FFFF），需进行额外的处理。这可以使用代码单元（称为代理）对来实现。如果代理对的处理过程未经过全面测试，这可能会导致棘手的错误和潜在的安全漏洞。

链接化

✔	如果一个看似域名的字符串包含表意文字句号字符“。”(U+3002)，请接受它并将其转换为“.”。

一般信息

✔	使用权威资源来确认域名。 请勿做出预判假设，如“所有 TLD 的长度都是 2字符、3字符、4字符 或6字符”。
✔	确保产品或功能会正确处理数字。 例如，应该将 ASCII 数字和亚洲表意数字均视为数字。
!	寻找在不常见地方出现的电子邮件地址： • 艺术家/作者/摄影师/版权元数据 • 字体元数据 • DNS 联系记录 • 二进制版本信息 • 支持信息 • OEM 联系信息 • 注册、反馈和其它表单

[12] 禁用：IDN 中不应包含的代码点。请参见：https://tools.ietf.org/html/rfc5892

[13] CONTEXTJ：连接控制的上下文规则。请参见：https://tools.ietf.org/html/rfc5892

!	在一些不常见的字段数据中寻找潜在 IRI[14] 路径： • 单一标签机器名称，而不论加载的系统代码页如何 • 完整的机器名称，而不论加载的系统代码页如何
✔	除 UTF-8 外，使用 GB18030（中国）来提供中文支持[15]。
!	在生成新域名和电子邮件地址时，限制允许使用的代码点 所有使用电子邮件地址的产品，在接受国际化电子邮件地址时，应允许大于 U+007f 的字符。也就是说，大于 U+007f 的字符不应被视为无效字符。 但是，用户创建新 IDN 或电子邮件地址时，并不需要使用所有字符，而应使用以下IDN可用字符列表：http://unicode.org/reports/tr36/idn-chars.txt 预先防止创建某些 IDN 或电子邮件地址可以缓解某些潜在的安全性和可访问性问题。（注：伯斯塔尔法则仍然要求软件接受此类字符串[如果存在]。）
!	请注意，不能总是仅通过自动化测试案例来评估普遍接受性。 例如，测试应用或协议如何处理网络资源或许并不可行，有时，最好是通过功能规范审核和设计审核来验证合规性。
!	不能机械地认为，某些功能组件不直接调用域名解析 API，也不直接使用电子邮件地址，因此它们与UA无关。 了解功能组件如何获取域名和电子邮件地址；组件并不仅限于通过网络交互来获取这些信息，下面提供了一些示例，说明组件如何获取网络名称： • 组策略 • LDAP 查询 • 配置文件 • Windows 注册表 • 与其它组件/功能相互传输
✔	执行代码审核，避免缓冲区溢出攻击。 • 在 Unicode 中，字符串大小写可能会导致长度变化：Fluß → FLUSS → fluss • 进行字符转换时，文本尺寸可能会显著增多或减少

域名的权威来源

DNS 根区

有一些方法可提供 TLD 权威列表。首选是 TLD 根区，根区文件已经DNSSEC 签名，可视为已经过适当验证。通过以下任意链接均可获取根区：

[14] IRI：国际化资源标识符。请参见：https://www.ietf.org/rfc/rfc3987.txt
[15] GB 18030-2000 是中国政府制定的标准，它指定了经扩展的代码页，以便在中国市场使用。请参见：http://icu-project.org/docs/papers/unicode-gb18030-faq.html

- http://www.internic.net/domain/root.zone
- http://www.dns.icann.org/services/authoritative-dns/index.html
- http://data.iana.org/TLD/tlds-alpha-by-domain.txt

公共后缀列表

公共后缀列表 (PSL) 由谋智基金会 (Mozilla Foundation) 的志愿者负责管理，它提供了域名后缀的准确列表。此列表包含一组用点连接并采用 UTF-8 编码的 DNS 域名或通配符。如果要将 PSL 作为域名的权威来源，您的软件必须定期接收 PSL 更新。请勿将 PSL 的静态副本复制到没有更新机制的软件中。可以使用下面的链接，通过您的应用定期下载经过更新的列表。该列表每天从 Github 接收一次更新：

- https://publicsuffix.org/list/public_suffix_list.dat

其它挑战

一般信息

IDN 的可变编码	在某些应用中，IDN 的编码方式是可变的： • 如果域名被视为互联网名称，则会根据 IDNA 标准，采用 Punycode 编码，但是 • 如果域名被视为局域网（“内联网”）上的名称，则会采用 UTF-8 编码
检测和转换字符集的机制	一些比较早的电子邮件应用在编码时采用的是本地代码页，而且未建立任何可在必要时检测和转换字符集的机制。电子邮件标头（TO、CC、BCC、主题）尤其如此。
无法处理非 DNS 协议数据	一些执行 IDNA 的应用（如 IE7+）无法处理非 DNS 协议数据。这会对使用非 DNS 协议数据访问资源造成影响。
管理单一用户身份中的多个电子邮件地址的机制	如果用户将多个电子邮件地址作为别名使用，则可能很难将这些地址作为单一用户身份进行管理。 电子邮件程序可能会将指定此类别名的流量传送到同一邮箱，但应用可能仍然认为这些电子邮件属于不同身份。

给软件开发者的提示

✔	允许用户生成域名或电子邮件地址时，请考虑避免使用视觉上易于混淆的字符，以防范同形异义字攻击。IDN 只允许使用以下可用字符列表：http://unicode.org/reports/tr36/idn-chars.txt

IDN 形式的电子邮件地址以及它为什么与 EAI 不同

EAI 被定义为仅使用 Unicode编码，而不允许使用 A-标签 (Punycode)。

但是，开发者有时使用第三方电子邮件软件和服务来处理 IDN 形式的电子邮件地址，而不是将其完全转换为 Unicode形式。 由于 IDN 可以采用 Punycode编码，因此一些现有的软件会同时允许以 ASCII（Punycode） 或 Unicode 来表示电子邮件地址的 IDN 部分。例如，一些软件会在各种用途（发送、接

并非所有软件都会同等对待这两个 IDN 形式的电子邮件地址

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

收和搜索）中同等对待这两个“IDN 形式的电子邮件”地址：

但是，即使两者等效，一些软件也并不会同等地对待它们，因为尚没有任何规范要求软件在进行比较之前应将 A-标签（如“xn--q9jyb4c”）转换成其对应的 U-标签（如“みんな”）。这可能会导致无法预料的用户体验结果。如果一些软件为了实现“兼容性”而将 U-标签转换成 A-标签，用户可能会感到特别

困惑；由于邮件通常会被回复或转发，那些在用户看来明显不同或无法按预期进行搜索和排序的地址可能会有所增多。

在下面的示例中，一些软件可能会尝试使用 Punycode 来转换电子邮件地址的本地部分，在地址的本地部分创建看起来像是 A-标签的内容。现有 RFC 并不允许这样做，而且，这很可能会导致某些系统无法接收电子邮件，以及造成上文所述的搜索和排序困难。

切勿使用 Punycode 转换电子邮件地址的本地部分

✔ 用戶@example.みんな

✖ xn--youq53b@example.xn--q9jyb4c

一些支持 UA 的鲁棒性软件和服务也许能够处理并同等对待所有这些格式，甚至是那些并不符合 RFC 规定的格式。但是，支持 UA 的软件不应该只能生成真正的 EAI 电子邮件地址。

链接化及其面临的挑战

现代软件有时允许用户直接输入一个看似网址、电子邮件地址或网络路径的字符串，以此来自动创建超链接。例如，如果应用将“www.”视为特殊前缀，或将“.org”视为特殊后缀，则在电子邮件中输入“www.icann.org”后会自动创建一个指向 http://www.icann.org 的可点击链接。

链接化应一致处理所有符合规则的网址、电子邮件名称或网络路径。

链接化是应用接受某个字符串，并动态确定是应该创建指向互联网位置 (URL) 还是电子邮件地址 (mailto:) 的超链接时执行的操作。

链接化使用软件开发者创建的算法和规则来确定是否应将字符串视为链接。与此相关的是，人们如何确定某个字符串为域名。很明显，浏览器、电子邮件客户端和文字处理程序肯定属于这类应用，但除此之外，还有许多其它应用也存在做出这种决策的需要。

最佳实践建议

1. 尝试在出现显式协议前缀（如“http://”、“ftp://”、“mailto:”）时执行链接化操作，但仅在字符串的剩余部分也符合规则时才完成链接化

示例字符串	预期行为/结果
example.com	无链接化操作，因为协议标识不存在或无法推导得出。
http://example.com	创建超链接，因为协议标识很明确
http:example.com	无链接化操作，因为句法错误（缺少 //）
http://example.a	无链接化操作，因为 ICANN 政策要求 TLD 至少为两个字符。注意：内部网络可能支持此句法。
http://example..ab	无链接化操作，因为句法错误（连续的点）

示例字符串	预期行为/结果
http://普遍接受-测试.世界	创建超链接，因为协议标识明确。

2. 尝试在出现隐式协议前缀时执行链接化操作（如由“www”推导出“http://www”）

示例字符串	预期行为/结果
www.example.com	创建超链接，因为隐含了协议[16]标识
label@example.com	创建 mailto:label@example.com，因为隐含了协议标识。

3. 如果字符串的其它部分均符合规则，则将表意文字句号“。”(U+3002) 映射为句点“.”(U+002E)（例如，将 http://田中。com 映射为 http://田中.com）。

4. 如果将 TLD 用作“特殊后缀”来确定可链接性，必须包括所有 TLD。应经常动态更新有效 TLD 列表。

[16] 注：实际的网站可能会要求终端用户输入 https:// 而不是 http://。如果是这样，便可能无法解析超链接，或返回错误页面。

第 3 部分：进阶主题

复杂文字

如果您并非开发者，不需要创建自己的字符串解析库，您可能会对复杂文字的细节不感兴趣。但是，这里提供的摘要信息可确保所有读者在用户体验中遇到这些文字时，都足以识别出与它们相关的代码错误。

从右至左书写的语言和 Unicode 一致性

水平呈现文本时，大多数文字都从左至右显示字符。但是，也有一些文字，如阿拉伯文或希伯来文，它们按从右至左的顺序显示水平文本。此外，当从右至左书写的文字使用从左至右书写的数字，或使用英语或其它文字中的嵌入词时，也可能会双向显示文本（从左至右 — 从右至左）。

如果水平方向的文本不统一，就可能会造成问题和歧义。为解决此问题，人们设计了一个算法来确定双向 Unicode 文本的方向性。

应用程序应根据 **Unicode 双向算法**描述的一套规则来确定文字在显示时的正确顺序。我们通常将这种算法称为“**Bidi 算法**”。

Bidi 算法

Bidi 算法规定了软件应如何处理同时包含从左至右 (LTR) 和从右至左 (RTL) 字符序列的文本。为短语指定的**基本方向**[17]将确定文本的显示顺序。

为确定是从左至右还是从右至左显示序列，Unicode 中的每个字符都具有关联的方向属性。大多数字母为**强类型**（**强字符**）的 LTR（从左至右）字符。从右至左书写的文字中的字母为强类型的 RTL（从右至左）字符。强类型 RTL 字符序列将从右至左显示。这不依赖于周围的基本方向。例如：

(LTR) example - مثال (RTL)。

一行中可以混合不同方向的文本。这种情况下，Bidi 算法会根据每个方向性相同的连续字符序列生成单独的**方向串**。

在 Unicode 中，空格和标点不是强类型字符，可以按 LTR 或 RTL 显示，因为它们可以用在任何类型的文字中。因此，它们被归类为**中性**或**弱字符**。弱字符是指那些方向性不明确的字符。这类字符的示例包括：

- 欧洲数字
- 东阿拉伯-印度数字
- 算术符号和货币符号
- 许多文字中常用的标点符号，如冒号、逗号、句号和不换行空格

中性字符的方向性取决于上下文。部分示例如下：

- 制表符
- 段落分隔符

[17] 在 HTML 中，基本方向要么继承自文档的默认方向（为从左至右），要么由最近的父元素使用 `dir` 属性进行显式设置。

- 大多数其它空白字符

如果中性字符介于两个方向性相同的强类型字符之间，则它也会采用该方向性。例如，介于两个 RTL 字符之间的中性字符本身也会被视为 RTL 字符，并且会延续方向串的方向性：

- نطاق.مثال

即使两个强类型字符之间有多个中性字符，软件也会以相同方式处理这些字符。

如果空格或标点位于两个方向性不同的强类型字符之间，则在处理时，软件会认为中性字符具有与主导基本方向相同的方向性。例如：

- example.مثال

除非存在方向覆盖，否则**数字**会始终按 big-endian 编码（和输入）[18]，并显示为 LTR。弱方向性仅适用于整个数字。

有关 Bidi 算法的详细信息，请访问：http://unicode.org/reports/tr9/tr9-11.html

域名的 Bidi 规则

Bidi 域名是指域名中至少包含一个 RTL 标签。在确定 Bidi 域名中的标签需满足哪些条件时应遵循一定的规则。此规则可以在 RFC 5893 第 2 部分找到：https://tools.ietf.org/html/rfc5893

连接符

一些语言使用拼音文字，其中的单音素使用两个字符（称为**二合字母**）来书写。换言之，二合字母是指用两个连续字母来表示一个音（或**音素**）的字母组。

英语中的二合字母示例

ch（如用在 *church* 中）	*th* (*then*)	*sh* (*shoe*)
ph (*phone*)	*th* (*think*)	

一些二合字母完全组合成了**连字**。在书写和印刷过程中，如果两个或多个字素或字母组合成单个字形，就会出现连字。例如，与字符 (&) 即由拉丁字母 *e* 和 *t* 组合而成（“et”等于“and”，意思是“和”）。

如果连字和二合字母在所有使用给定文字的语言中具有相同的含义，Unicode 规范化通常会消除它们之间的差异，使它们相互匹配。如果两者具有不同的含义，则必须使用其它方法（可能在注册管理机构级别做出选择）进行匹配，否则必须告知用户不会进行匹配。RFC 5894 第 4.3 部分提供了针对不同含义的示例：https://tools.ietf.org/html/rfc5894

统一域名编码协会 (Unicode Consortium) 制定了两个主要策略，以确定在应用 Bidi 算法之后特定字符的结合行为：

- “进行变形时，可以重新参考原始后备存储器，看是否存在相邻的 ZWNJ 或 ZWJ[19] 字符。

[18] “Big-endian 和 little-endian 是说明在计算机内存中存储字节序列时所采用顺序的术语。Big-endian 是指首先存储（最低存储地址）‘大头’（序列中最重要的值）。Little-endian 是指首先存储‘小头’（序列中最不重要的值）。”
资料来源：http://searchnetworking.techtarget.com/definition/big-endian-and-little-endian

- 或者，也可以用与那些相邻字符关联的带外字符属性替代 ZWJ 和 ZWNJ，以确保相关信息不会干扰 Bidi 算法，并在重新安排这些字符的过程中保留这些信息。应用 Bidi 算法之后，就可以将带外信息用于完成相应的变形。”[20]

如果注册管理机构对于根据 IDNA2003 和当前规范如何处理可能具有不同含义的字符串未做规定，这种不同就可能会被用于实施域名匹配或域名混淆攻击。因此有必要做出此类规定。

有关连接符的详细信息，请参见 RFC 5894 第 4.3 部分：https://tools.ietf.org/html/rfc5894

同形字和易于混淆的相似字符

同形字是指由于大小和字形相似，乍看起来似乎完全相同的字符。

同形字示例

西里尔字符 **а**	=	Unicode 数字 0430
拉丁字符 **a**	=	Unicode 数字 0061

为避免注册易于混淆的域名，注册管理机构可以采用“同形字捆绑”规程。[21]

同形字捆绑是指在注册 IDN 时，注册系统会自动捆绑该域名的所有同形字（如有）。也就是说，会一次性捆绑多个域名，使得无法再注册该捆绑包中的任何其它域名。

同形字捆绑是注册管理机构避免潜在网络钓鱼攻击（即利用易于混淆的字符来欺骗用户）的最佳做法。

有关 Unicode 可混淆检测安全机制的详细信息，请访问：

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

如需查看同形字列表，请访问：

- http://homoglyphs.net

有关易于混淆的字符及最佳实践，请访问：

- M3AAWG Unicode 滥用概述和教程
  https://www.m3aawg.org/sites/default/files/m3aawg-unicode-tutorial-2016-02.pdf
- M3AAWG Unicode 滥用预防最佳实践
  https://www.m3aawg.org/sites/default/files/m3aawg-unicode-best-practices-2016-02.pdf

---

[19] 有关 ZWNJ/ZWJ 的详细信息，请访问：http://www.unicode.org/L2/L2005/05307-zwj-zwnj.pdf

[20] 资料来源：Mark Davis、Aharon Lanin、Andrew Glass。2015 年。*Unicode*。http://unicode.org/reports/tr9

[21] https://www.icann.org/resources/pages/idn-guidelines-2011-09-02-en

规范化和大小写转换

规范化

Unicode 规范化有助于确定两个 Unicode 字符串是否彼此等价。在 Unicode 中，一些字符可以用几个代码序列表示。这称为 **Unicode 等价性**。Unicode 提供两种类型的等价性：

- 规范 (NFD)
- 兼容性 (NFK)

表示同一字符的序列称为**规范等价**。在打印或显示时，这些序列具有相同的外观和含义。例如：

规范等价字符示例

U+006E（拉丁小写字母“n”）后接 U+0303（组合用颚化符“◌̃”）	=	**ñ**
U+00F1（西班牙小写字母“ñ”）	=	**ñ**

兼容性等价是指序列的外观可能不同，但在某些上下文中含义相同。这是字符或字符序列之间的一种较弱的等价关系。

兼容性等价字符示例

U+FB00（合字“ff”）	=	**ff**
U+0066 U+0066（两个拉丁字母“f”）	=	**ff**

在上例中，代码点 U+FB00 定义为与序列 U+0066 U+0066 兼容性等价，而不是规范等价。规范等价的序列同时也兼容性等价，但反之则不一定。

为避免在使用规范等价但相互存在不同的字符序列时出现互操作性问题，W3C 建议对所有内容应用“规范化形式 C”(Normalization Form C)[22]。

如需查看在任何规范化形式下可能出现变化的所有字符列表，请访问：
http://www.unicode.org/charts/normalization

其它一些要点包括：

- 只有未进行 NFKC[23] 转换的字符才为有效字符。
- 如果两个应用共享 Unicode 数据，但采用的规范化方式不同，这时就可能出现错误和数据丢失。
- 规范化形式不可频繁变动，必须长时间保持稳定。换句话说，字符串必须在所有未来版本的 Unicode 中始终保持规范化状态（向后兼容）。

[22] NFC：先进行规范分解，然后进行规范合并。

[23] NFKC：先进行兼容性分解，然后进行兼容性合并。

给软件开发者的提示

✖	不要通过转换为大写字母来实现规范化，或在规范化时忽略不占位字符，因为这些方法可能会增加排序、数据复制、数据导入和导出、数据检索的难度，并可能导致数据丢失或损坏。

有关规范化形式的详细信息，请访问：http://www.unicode.org/reports/tr15

大小写转换

大小写转换是指使两段大小写不同，但其它方面相同的文本完全匹配的过程。将 [a-z] 映射为 [A-Z] 适用于大多数简单的纯 ASCII 文本文档。但是，它不适用于使用其它字符的语言。

Unicode 为每个 Unicode 代码点定义了默认的大小写转换映射。这些映射分为**常用**和**完全大小写映射**：

- **常用大小写映射**是指那些直接映射为单一匹配（主要为小写）代码点的映射
- **完全大小写映射**是指那些正常情况下需要多个 Unicode 字符映射

W3C[24] 称，在转换映射时，必须注意相关值是否限定为 Unicode 的 ASCII 子集，或者词汇表是否允许使用可能具有更加复杂的大小写转换要求的字符（如拉丁字母重音或更广泛的 Unicode 字符，包括非拉丁文字）。[25]

给软件开发者的提示

✔	除大小写转换以外，还应考虑 Unicode 规范化。

有关 Unicode 规范化的详细信息，请参见：

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

有关大小写转换的建议，请访问：

- https://www.w3.org/International/wiki/Case_folding

[24] W3C：万维网联盟 (W3C) 是一个国际社区，其成员组织、全职员工和公众在此互相协作，共同制定网络标准。请参见：https://www.w3.org

[25] 资料来源：A Phillips.2015 年。*万维网的字符模型：字符串匹配和搜索*。https://www.w3.org/TR/charmod-norm

第 4 部分：术语表和其它资源

术语表

A-标签	以 ASCII 兼容编码 (ACE) 表示的国际化域名，即表示其如何根据 DNS 协议在内部传输。A-标签总是以前缀“xn--”开头，以与 U-标签相区别。
ACE 前缀	ASCII 兼容编码前缀。
ASCII 字符	美国信息交换标准码。这些字符包括基本拉丁字母以及欧洲-阿拉伯数字。它们被包括在为 IDN 奠定基础的、范围更广泛的“Unicode 字符”内。
API	应用程序接口 (API) 是一组用于构建软件和应用的例程、协议和工具。API 可能用于基于 Web 的系统、操作系统或数据库系统，它能够为使用给定编程语言开发该系统应用提供便利。
代码空间	一个范围，它定义了编码的上下界限。
代码点	代码点或代码位置指任何填充代码空间的数字值。它们用于区分数字与位序列编码，以及区分抽象字符与特定图示（图象字符）。
DNS 根区	根区是 DNS 的中央目录，它是将可读的主机名称转换为数字 IP 地址的关键组件。
EAI	国际化电子邮件地址 (EAI) 是一种要求在电子邮件地址的所有部分都使用 Unicode 的电子邮件地址。
IANA	它的全称是 Internet Assigned Numbers Authority（互联网号码分配机构）。其职能包括： • 维护互联网技术协议参数的注册 • 监督管理与互联网 DNS 根区相关的某些职责 • 分配互联网号码资源
ICANN	互联网名称与数字地址分配机构 (ICANN) 是一家国际性非营利机构，主要负责互联网协议 (IP) 地址的空间分配、协议标识符分配、通用顶级域名 (gTLD) 和国家和地区代码顶级域名 (ccTLD) 的系统管理，以及根服务器系统的管理职能。
IDN	国际化域名。IDN 是指包含在本地语种中使用的字符的域名，它们并非以基本拉丁字母表中的 26 个字母“a-z”、数字 0-9 和连字符“-”书写。
IDNA	国际化域名应用。
IDN ccTLD	包含在本地语种中使用的字符的国家和地区顶级域，它们并非以基本拉丁字母表中的 26 个字母“a-z”书写。例如： • .pф（俄罗斯） • .صر（埃及） • .السعودية（沙特阿拉伯）
IETF	互联网工程任务组 (IETF) 是一个由网络设计人员、运营商、供应商和研究人员组成的大型开放性国际社区，它主要关注互联网架构的发展和互联网的顺利运行，面向所有感兴趣人士开放。IETF 负责制定互联网标准，特别是与互联网协议集 (TCP/IP) 相关的标准。

语言	人类以口头或书面形式进行交流的方法，即以结构化、常规方式应用各种词汇。
国际化域名编码（Punycode）	它是一种算法，用于通过域名系统支持的有限 ASCII 字符子集来表示 Unicode。Punycode 旨在对国际化域名应用 (IDNA) 框架内的标签进行编码。
注册服务机构	用户在注册服务机构处注册域名。注册服务机构会记录注册人的联系人信息，并将技术信息提交给中央目录机构（称为“注册管理机构”）。
注册管理机构	囊括了在每个顶级域内注册的所有域名的权威主数据库。
RFC	意见征求稿 (RFC) 是互联网工程任务组 (IETF) 撰写的正式文档，它由委员会起草，随后接受利益相关方审核。
文字	书写时使用的字母或字符集，用于表示某种语言的音素。
二级域名	在域名系统 (DNS) 层级中，二级域（SLD 或 2LD）是指位于顶级域 (TLD) 下一级的域。例如，在 `example.com` 中，`example` 是 `.com` TLD 的二级域。
U-标签	U-标签是由 Unicode 字符组成且符合 IDNA 要求的字符串，其中至少包含一个非 ASCII 字符。U-标签与 A-标签之间的转换应依照 Punycode 规范 [RFC3492] 实现。
UA 就绪软件或 UA 就绪性	指支持普遍接受性的软件，即能够同等地接受、存储、处理、确认和显示所有顶级域，以及所有 IDN、超链接和电子邮件地址的软件。
统一域名编码 (Unicode)	通用字符编码标准。它定义了在文本文件、网页和其它类型的文档中显示单个字符的方式。Unicode 的目的在于支持全世界所有语言中的字符。它大约可支持 1,000,000 个字符，每个字符最多占用 4 个字节。请参见：http://unicode.org
UTF	Unicode 转换格式。它用于将 Unicode 代码点转换为字节流。UTF-8 是处理 IDN 和 EAI 的首选 UTF。UTF-8 可将 Unicode 转换为 8 位字节。
M3AAWG	信息传递、恶意软件和移动反滥用工作组 (M^{3}AAWG) 负责召集整个行业联合起来防范僵尸网络、恶意软件、垃圾邮件、病毒、DoS 攻击和其它在线漏洞。请参见：https://www.m3aawg.org/
W3C	万维网联盟 (W3C) 是一个国际社区，其成员组织、全职员工和公众在此互相协作，共同制定网络标准。请参见：https://www.w3.org/
ZWJ	零宽连接符是在对某些复杂文字（如阿拉伯文或任何印度文字）进行计算机化排版时使用的非打印字符。置于两个以其它方式无法连接的字符间时，ZWJ 将联合打印这两个字符。
ZWNJ	零宽无连接符是对利用连字的书写系统进行计算机化时使用的非打印字符。置于两个将以其它方式连接成连字的字符之间时，ZWNJ 将分别以最终格式和初始格式打印这两个字符。它也会产生空格字符的效果，但如果希望使单词间的连接更紧密，或将单词与其词素相连接时，最好是使用 ZWNJ。

如需查看完整的 ICANN 术语表，请访问：https://www.icann.org/resources/pages/glossary-2014-02-03-en

RFC

PUNYCODE RFC	
RFC 3492	**国际化域名编码 (Punycode)：国际化域名应用 (IDNA) 的 Unicode 的 Bootstring 编码** RFC 3492 将 Punycode 定义为： *“一种与国际化域名应用 (IDNA) 结合使用的简单有效的传输编码句法。”* Punycode 能够以独特、可逆的方式将 Unicode 字符串转换为 ASCII 字符串。此 RFC 定义了一个称为 **Bootstring** 的通用算法。使用此算法，基本代码点字符串可以唯一表示更大集合中的任何代码点字符串。 https://tools.ietf.org/html/rfc3492
IDN RFC	
RFC 5890	**国际化域名应用 (IDNA)：定义和文档框架** 此 RFC 介绍了国际化域名应用 (IDNA) 修订版本的使用环境和协议。 https://tools.ietf.org/html/rfc5890
RFC 5891	**国际化域名应用 (IDNA) 协议** 此 RFC 为以不需要更改 DNS 本身的方式注册和查询 IDN 指定了协议机制，这称为国际化域名应用 (IDNA)。 https://tools.ietf.org/html/rfc5891
RFC 5892	**Unicode 点和国际化域名应用 (IDNA)** RFC 5892 指定了一些规则，可用于确定在孤立或结合情景考虑时，代码点是否为国际化域名 (IDN) 的候选项。 https://tools.ietf.org/html/rfc5892
RFC 5893	**国际化域名应用 (IDNA) 的从右至左文字** 此 RFC 为国际化域名应用 (IDNA) 标签制定了新的 Bidi 规则，以用于国际化域名中的从右至左文字。 https://tools.ietf.org/html/rfc5893
RFC 5894	**国际化域名应用 (IDNA)：背景、说明和理由** 本介绍文档概述了用于处理更新版本的 Unicode 的修订系统，并对该系统组件进行了说明。 https://tools.ietf.org/html/rfc5894

RFC 5895	**国际化域名应用 (IDNA) 2008 的映射字符** 此 RFC 介绍了实施者在接收用户输入与向新 IDNA 协议 (2008) 传送允许的代码点期间可以执行的操作。其中说明了将对用户输入执行的操作，以便在关于网络的协议中使用用户输入。另外还介绍了映射的常规实施流程。 https://tools.ietf.org/html/rfc5895
EAI RFC	
RFC 6530	**国际化电子邮件概述和框架** 此标准引入了一系列规范，它们定义了完全支持国际化电子邮件地址所需的机制和协议扩展。本文档说明了如何整合电子邮件国际化的各个要素，以及与消息传递、标头格式和处理相关的主要规范之间的关系。 https://tools.ietf.org/html/rfc6530
RFC 6531	**国际化电子邮件的 SMTP 扩展** 本文档定义了简单邮件传输协议扩展，使得服务器能够公布其能够接受和处理国际化电子邮件地址及国际化电子邮件标头。 https://tools.ietf.org/html/rfc6531
RFC 6532	**国际化电子邮件标头** 本文档详细说明了互联网邮件格式和 MIME 的改进，这项改进使得在电子邮件地址和大多数标头字段内容中使用 Unicode 成为可能。此文档说明了互联网邮件格式 (RFC 5322) 和 MIME 的改进，它允许在标头字段值（包括电子邮件地址）中直接使用 UTF-8，而不仅仅是 ASCII。其中为这种扩展格式定义了一种新的媒体 — message/global。该规范还加强了在消息顶级类型的任何子类型中使用非同一性内容传输编码 (content-transfer-encoding) 的 MIME 限制，以便在现有邮件基础架构间安全传送 message/global 部分。 https://tools.ietf.org/html/rfc6532
RFC 6533	**国际化投递状态和处理通知** 此规范为国际电子邮件地址添加了一种新地址，使得即使在降级后仍然能够正确保护包含非 ASCII 字符的原始收件人地址。它还为投递状态通知和消息处理通知提供了经过更新的内容返回媒体类型，以支持使用新类型的地址。 https://tools.ietf.org/html/rfc6533

主要标准

ISO 10646 (Unicode)	为了给处理各种语言的电子信息奠定常用技术基础，国际标准化组织 (ISO) 制定了国际编码标准 ISO 10646。ISO 10646 为对世界上所有主要语言的字符（包括繁体和简体中文字符）进行编码提供了统一的标准。这个大型字符集称为通用字符集 (UCS)。定义了该字符集的Unicode 标准也进一步定义了其它字符属性和对实施者有用的应用详情。

	Unicode 是统一域名编码协会设计的字符编码体系，旨在为交换、处理和显示世界上所有主要语言的书写文本提供支持。ISO 10646 和 Unicode 为其常用字符库定义了几种编码形式：UTF-8、UCS-2、UTF-16、UCS-4 和 UTF-32。 http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=63182
GB18030（中国）	GB 18030-2000 是中国政府制定的标准，除 UTF-8 以外，它还指定了一个扩展代码页以便在中国市场使用。字符库的内部处理代码可以并且应该为 Unicode；但是，该标准规定，软件提供商必须保证能够在 GB18030 与内部处理代码之间成功切换。当前或未来在中国市场销售的产品必须规划代码页迁移，以完全支持 GB18030。GB18030 是一项“强制性标准”，中国政府规定了认证流程来实施 GB18030 部署。 http://icu-project.org/docs/papers/unicode-gb18030-faq.html
Unicode 技术标准 46：Unicode IDNA 兼容性处理	此规范定义了与 IDNA 2008 协议的标准要求相一致的映射，并且它尽可能兼容 IDNA 2003。对于客户端软件，该规范提供了利用现有数据处理域名时最符合用户预期的行为。 http://unicode.org/reports/tr46/

在线资源

API	Windows API https://www.msdn.microsoft.com/enus/library/windows/desktop/ff818516%28v=vs.85%29.aspx SharePoint API https://msdn.microsoft.com/en-us/library/office/jj860569.aspx 公共后缀列表 https://publicsuffix.org/list/public_suffix_list.dat ICANN 权威 TLD 列表 http://data.iana.org/TLD/tlds-alpha-by-domain.txt Android API http://developer.android.com/guide/index.html MAC IOS API https://developer.apple.com/library/mac/navigation .Net Framework https://msdn.microsoft.com/en-us/library/system.text.encoding(v=vs.110).aspx
Unicode 安全性	Unicode 安全性注意事项 http://www.unicode.org/reports/tr36 Unicode 安全机制 http://www.unicode.org/reports/tr39

Unicode 字符分组	Unicode 代码平面 http://en.wikipedia.org/wiki/Mapping_of_Unicode_character_planes GB18030 概述 http://en.wikipedia.org/wiki/GB_18030 GB18030-2000 与 Unicode 之间的权威映射表 http://source.icu-project.org/repos/icu/data/trunk/charset/data/xml/gb-18030-2000.xml Unicode 规范化 https://en.wikipedia.org/wiki/Unicode_equivalence
Unicode 漏洞利用	Unicode 技术报告 36 第 3.1 节“UTF-8 漏洞利用” http://unicode.org/reports/tr36/#UTF-8_Exploit M3AAWG Unicode 滥用预防最佳实践 https://www.m3aawg.org/sites/default/files/m3aawg-unicode-best-practices-2016-02.pdf M3AAWG Unicode 滥用概述和教程 https://www.m3aawg.org/sites/default/files/m3aawg-unicode-tutorial-2016-02.pdf 另见： http://www.unicode.org
其他资源	URI http://tools.ietf.org/html/rfc3986 域名系统：非技术性说明 — 普遍可解析性为什么如此重要 http://www.internic.net/faqs/authoritative-dns.html ICANN 术语表 https://www.icann.org/resources/pages/glossary-2014-02-03-en

致谢

作者衷心感谢以下人员为撰写本文档所做的贡献和协作：

伊莉莎・阿格匹安 (Eleeza Agopian)

格温・卡尔森 (Gwen Carlson)

钟宏安 (Edmon Chung)

萨曼莎・迪克森 (Samantha Dickinson)

唐・霍兰德 (Don Hollander)

香塔尔・勒布鲁芒 (Chantal Lebrument)

安东尼亚塔・曼吉亚科蒂 (Antonietta Mangiacotti)

理查德・默丁杰 (Richard Merdinger)

拉姆・莫罕 (Ram Mohan)

戴维・莫里森 (David Morrison)

卡洛琳・阮 (Carolyn Nguyen)

迈克尔・派拉格 (Michael D. Palage)

库尔特・普里茨 (Kurt Pritz)

安德烈・夏坡 (André Schappo)

宋峭 (Zheng Song)

拉斯・斯蒂芬 (Lars Steffen)

安德鲁・苏利文 (Andrew Sullivan)

丹尼斯・谭 (Dennis Tan)

维尼・余 (Winnie Yu)（音译）

ICANN北京合作中心衷心感谢以下中国专家审校本文档中文译文：

王伟

马迪

姚健康