Introduction to Internationalization and Unicode Internationalization Technologies

Joel Sahleen - UTW 2025

Nov. 11, 2025

UNICODE

Overview

Introductions

- I ❤ 🤔 🤯 i18n/l10n
- Engineer, architect, manager
- Adobe, Domo, Spotify
- Classically-trained sinologist
- Early Chinese language and culture
- Researcher and teaching fellow
- Consultant and developer
- Unicode CET member
- Co-Chair, W3C I18n Working Group

Joel Sahleen

About This Tutorial

This tutorial is an introductory course on software internationalization (i18n). It is designed to help you learn the basics of developing international software, and understand the Unicode internationalization technologies that make it possible.

- **Audience**: Persons new to internationalization
- **Purpose**: To prepare you for subsequent tutorials
- **Scope**: A broad, high-level overview
- **Focus**: How everything fits together
- **Goal**: Foundational knowledge
- **Format**: Presentation

Content

The tutorial examines four main topics. These topics are defined in terms of four basic questions that cover the fundamentals.

- What is internationalization?
- What is internationalization about?
- How does internationalization work?
- What does internationalization involve?

Organization

Each question/topic is covered in a separate section. Sections are organized so that later sections build on earlier sections. The content of all the sections will be reviewed during the conclusion.

- Internationalization Concepts and Relationships
- Internationalization Focus Areas and Concerns
- Unicode Internationalization Technologies
- The Internationalization Life Cycle

Structure

Each sections starts with a summary and list of objectives. Each section ends with takeaways and links to resources for learning more about what was covered.

- Summary
- Objective
  - [Section Content...]
- Takeaways
- Learning More

Materials

A PDF of this deck will be available on the conference website after in a few weeks. The website will also have a link to the recording on the Unicode YouTube channel.

- No need to take pictures of the slides
- The slides are overly wordy and boring. They are intended to be an artifact that can be used a reference.
- Most of the important content in this tutorial will be verbal, so it’s best to focus on what is said rather than what is written.

Logistics

We have a lot of ground to cover and only have 90 minutes in which to cover it. This requires some ground rules.

- Questions and comments are welcome during the presentation, but please keep them short.
- There will be a dedicated Q&A session at the end of the tutorial. When in doubt, wait.
- Keep the discussion professional. Be nice, and do not monopolize the available time.

Internationalization Concepts and Relationships

What is internationalization?

Summary

This section introduces the concepts and relationships that define what internationalization is and why it matters. It does this by analyzing the terms we use to talk about internationalization, in English. Doing this serves three main purposes.

1. It forces us to be explicit about how certain terms will be used, which helps avoid misunderstanding.
2. It allows us develop a shared understanding of what these terms mean, which simplifies communication.
3. It provides a way of understanding the concepts and relationships to which the terms refer by using language as a “fish trap.”

Words and Concepts

A fish trap exists for the sake of fish. Once you've caught the fish, you can forget about the trap. A rabbit snare exists for the sake of rabbits. Once you've caught a rabbit, you can forget about the snare. Words exist for the sake of meaning. Once you've caught the meaning, you can forget about the words. Oh, where can I find a person who has forgotten words so I can have a word with them?"

– Zhuangzi

Objectives

- Describe the etymology of the English term **nation**. Explain why the development of this term is significant.
- Clarify the meaning of **local**, **national** and **international**. Explain how the concepts behind these terms are related to each other**.**
- Distinguish between **international software**, **software internationalization** and **internationalization technologies**. Explain how they fit together.
- Explain the relationships between internationalization, **localization**, **translation**, and **globalization**. Situate internationalization in its broader context.

The Etymology of “Nation”

The term “nation” comes from the Latin word *natio*. It means “of common origin or birth.” Over time, this meaning has expanded to include groups of people that have other types of common origins.

- Ethnicity - A nationality
- Country - A territorial nation
- Citizenship - A political nation-state

The members of a nation often use a common language (or set of languages), and share important cultural traditions and social systems.

Local vs. National vs. International

- **Local**. Associated with or involving a user's the immediate environment.
  National > Local > Personal.

- **National**. Associated with or involving the people of an entire nation.
  International > National > Local.

- **International**. Associated with or involving two or more nations.
  Global > International > National.

International Software

International software is software that works internationally, and can adapt to meet the needs and expectations of users in any national or local context. The specific international requirements for an application depend on the nature of the application and its industry. However, certain international requirements are common.

- Complies with international standards
- Not specific to a particular context (generic)
- Has contextual awareness and decision logic
- Supports multiple languages, scripts, regions and cultural traditions
- Can deliver different localized user experiences to different audiences
- Interacts with users in way that is accessible and feels natural

Internationalization (I18n)

Internationalization (i18n) is a set of architectural principles and software development practices design to make an application work internationally, and support adaptation. If international software is a product, software internationalization is the process that produces it.

- Embedded in the software development life cycle
- Requires widespread involvement and support
- Includes both systemic and local feature development
- Intersects with localization, translation and globalization.

Internationalization Technologies

Internationalization technologies are technologies that are used in the software development process by software developers to enable the production of international software. These technologies encapsulate internationalization principles, patterns, and best practices, so that they are available to everyone.

- Standards
- Properties and algorithms
- Structured locale data
- Libraries and packages
- Programming languages
- Platform SDKs

Localization (L10n)

Localization is the process of adapting a software product to meet the needs and expectations of users in a particular local context.

Internationalization enables localization to scale by making it possible to non-destructively localize a software product for users in multiple local contexts.

- Interface localization
- Application localization
- Content localization

Translation (T9n)

Translation is the process of transferring information from one (written) language to another.

Internationalization can have positive (accuracy) and negative (complexity) effects on the translation process. Translation impacts localization quality.

- Decoding the source language information
- Re-encoding the information in a target language
- Translated information is applied as part of localization

Globalization (G11n)

Globalization is the process of international expansion, from fewer to more contexts, within a limited, well-defined whole. It constitutes an increase in scope and share.

Internationalization facilitates globalization by making it easier to expand localization to new features and markets.

- Globalization usually depends on factors that go beyond the scope of software development.
- Internationalization and localization complexity increases as more contexts are added to the process.

Takeaways

- The concept of a nation is complex and ambiguous. This makes it hard to identify nations and national characteristics.
- Local, national and international represent different points along a spectrum ranging from more specific contexts to more general contexts.
- International software works internationally and can adapt to any context.
- Internationalization is the process of developing international software
- Internationalization relies on internationalization technologies.
- Internationalization enables localization to scale.
- Internationalization affects translation accuracy and complexity, which can impact localization quality.
- Internationalization facilitates globalization and expansion, by making localization easier to expand and extend.

Learn More

The definitions of internationalization and internationalization-related terms in these online sources can be compared to the those presented in this tutorial. The important thing is to develop your own understanding.

- Start with the W3C I18n Working Group's About Internationalization page and our Internationalization Glossary.
- A similar, but less accessible, terminology can be found in RFC 3536 from the Internet Engineering Task Force (IETF).
- Brief definitions of internationalization and localization are also provided at the beginning of the Unicode Technical Quick Start Guide.

Internationalization Focus Areas and Concerns

What is internationalization about?

Summary

This section tries to define the focus and scope of internationalization, by using the concept of a locale to identify the dimensions that make up a user's localization preferences. The section starts with a discussion of IETF Language Tags and Unicode Locale Identifiers. The discussion focuses on the standard subtags and Unicode extensions, in order to highlight what these tags and extensions suggest about the main focus of internationalization. This is followed by an analysis of each focus areas and their problems. This is designed to show how operating in an international context makes the development of software applications more complicated, and requires a different approach to working with language, writing, geography and culture.

Objectives

- Introduce the concept of a locale and analyze its definition in the specification for the Locale Data Markup Language (LDML).
- Discuss the composition and syntax of IETF Language Tags, and explain how they are extended by Unicode Locale Identifiers
- Explore the focus areas represented by the different Language Tag subtags. Describe what each is about and what its main problems are.
- Describe the international diversity that exists within each focus area and show how that makes software development more complicated.

Locales

The first issue is basic: *what is a locale?* In this model, **a locale is an identifier (id) that refers to a set of user preferences that tend to be shared across significant swaths of the world**. Traditionally, the data associated with this id provides support for formatting and parsing of dates, times, numbers, and currencies; for measurement units, for sort-order (collation), plus translated names for time zones, languages, countries (regions), and scripts. The data can also include support for text boundaries (character, word, line, and sentence), text transformations (including transliterations), and other services.

https://www.unicode.org/reports/tr35/#what-is-a-locale

IETF BCP 47 Language Tags

A language tag is composed from a sequence of one or more "subtags", each of which refines or narrows the range of language identified by the overall tag. Subtags, in turn, are a sequence of alphanumeric characters (letters and digits), distinguished and separated from other subtags in a tag by a hyphen ("-", [Unicode] U+002D).

https://datatracker.ietf.org/doc/html/rfc5646#section-2.1

Example Language Tags

- zh // Chinese
- zh-cmn // Chinese Mandarin
- zh-Hans // Simplified Chinese
- zh-CN // Chinese from China
- zh-Hans-CN // Simplified Chinese from China
- zh-yue-HK // Cantonese Chinese from Hong Kong
- de-CH-1901 // Swiss German using the 1901 spellings
- ca-ES-valencia // Catalan as spoken in Valencia, Spain
- zh-x-key-value // Chinese with private use tags

Unicode Locale Identifiers

[BCP47] Language Tags provides a mechanism for extending language tags for use in various applications by extension subtags. Each extension subtag is identified by a single alphanumeric character subtag assigned by IANA.

The Unicode Consortium has registered and is the maintaining authority for two BCP 47 language tag extensions: the extension 'u' for Unicode locale extension [RFC6067] and extension 't' for transformed content [RFC6497]. The Unicode BCP 47 extension data defines the complete list of valid subtags.

https://www.unicode.org/reports/tr35/#unicode-bcp-47-u-extension

Examples of Unicode Locale Extensions

- ca – calendar algorithm
- cu – currency type
- dx – dictionary break script exclusions
- em – emoji presentation style
- fw – first day of week
- hc – hour cycle
- ms – measurement system
- sd – regional subdivision

Full list: https://www.unicode.org/reports/tr35/#Unicode_Locale_Extension_Data_Files

Significance

1. The subtags and syntax of IETF Language Tags show that there are four main aspects to a user’s localization preferences: language, script, region and culture.

2. The extensions defined for Unicode Locale Identifiers show that a user’s localization preferences may include a preference for using specific linguistic, scriptographic, regional, and cultural systems.

Implications

1. By looking at the four main aspects of a user's localization preferences, and examining each aspect's core features and concerns, we can begin to understand what internationalization, in general, is about.

2. By looking at the diversity that exists within each of these aspects, and examining what is required to support this diversity in a technological setting, we can begin to understand why internationalization is necessary and what it is meant to accomplish.

Languages: Definition

A language is a conventional system of communication used by the members of a society or nation. A language consists of two main elements: a vocabulary, or system of words, and a grammar, or system of rules for using words. The study of language is called linguistics. It encompasses several different sub-disciplines.

- **Semiotics and Lexicology**: The study of signs and words
- **Phonology and Orthography**: The sonic or visual composition of words
- **Morphology and Syntax**: The grammatical forms and functions of words
- **Semantics and Pragmatics**: The communication of meaning and intent
- **Taxonomy and Phylogeny**: The classification and evolution languages
- **Translation and Interpretation**: The transfer of information between languages

Languages: Diversity

According to Ethnologue, there are **7,159 languages** currently in use in the world today and many more historical and extinct languages. Current languages can be categorized into **142 different language families**. 81.9% of languages are institutional or formally adopted for use, while 16.9% are not formally adopted but have a stable population of speakers. 1.2% of languages are endangered and at risk of becoming extinct. Languages can vary in terms of vocabulary, pronunciation, orthography and grammar. Languages vary by origin, region, social rank, and individual. Some languages are inflected and use morphological changes to signal grammatical function, while others are non-inflected and rely on sentence position or other mechanisms.

Languages: Technology

Computer systems are historically text-based, but voice-based interfaces are increasingly common. Certain grammatical features apply to both speech and writing and must be supported by both types of interfaces. This introduces complexity when composing messages with dynamic content.

- **Pluralization**. Zero, one, two, few, many, other
- **Tense**. Past, present and future
- **Aspect**. Simple, progressive, perfect, progressive perfect.
- **Gender**. Male, female, neutral
- **Person**. First, second, third

Scripts: Definition

A script is a conventional set of graphical symbols (**graphemes**) that can be used to write one or more languages. Graphemes are also called characters, and so a script can also be called a character set. Scripts are the graphical elements that make up one part of a writing system. The other part consists of a set of rules (an **orthography**) that governs how scripts are used to represent languages. The study of scripts is called **scriptology** or **scriptography**, and has given rise to several sub-fields.

- **Graphology**. The study of written characters.
- **Paleography**. The study of ancient scripts.
- **Epigraphy**. The study of inscriptions.
- **Calligraphy**. The study and practice of handwriting.
- **Typography**. The style and appearance of printed text.

Scripts: Diversity

Recent estimates suggest that there are around **293 known writing systems**, of which **156 are currently in use**. The remaining **137 writing systems are historical** in nature. Some writing systems, like Japanese, use multiple scripts, which some scripts, like Latin, are used to write multiple languages. Scripts can be alphabetic (Cyrillic), syllabic (Hiragana), logographic (Han) or featural (Hangul). Some traditional scripts, like Mongolian, are written vertically in columns. Most modern scripts are written horizontally in rows. Most scripts are written from left to right, but there are several scripts that are written from right to left. When right-to-left and left-to-write scripts appear in the same text, the result is called bidirectional text.

Scripts: Technology

Computer systems store information using bits and bytes. To represent text in software, characters must be encoded into bytes, and arranged into a linear sequences called strings. To display text, these encodings must be mapped to corresponding glyphs within a font. Fonts include data tables that specify things like glyph substitution and glyph positioning. In some writing systems, multiple characters can be combined to make a single glyph called a ligature. Character and ligature boundaries are tracked by font rendering and shaping engines, in order to allow for features like cursor positioning and text selection. These boundaries are also used for sentence, word and line-breaking, with different writing systems having different line-breaking rules. Variable fonts include different weights and styles of glyphs that can be used for decoration and emphasis. Text direction is controlled by an algorithm, that uses character properties, directional markers and isolates to determine the proper sequence of characters.

Regions: Definition

A region is a territorial unit that may or may not be associated with specific nationalities or a particular nation-state. Regional localization preferences take the form of variations in language, writing system and the use of cultural traditions. For example, the locales defined for the Common Locale Data Repository (CLDR) include **305 regional variants**. The following are examples of other localization preferences that vary according to region.

- Toponyms, addresses and phone numbers.
- Time zones and meta zones
- Currencies and currency formats.
- Date, time, and datetime formats
- Calendar, number, and measurement systems.

Regions: Diversity

Many regions include multiple nationalities and use multiple languages and cultural traditions. Conversely, some nationalities, languages and cultural traditions extend across multiple regions. Regions have different forms of administrative divisions, and these divisions often correspond to subregions or territorial units like states and provinces. The names used to refer to particular places are different in different regions, and different regions may have different address and phone number formats. Addresses may be geocoded into coordinates, and regional boundaries are used to define different time zone and meta zones. Different countries may use different monetary, calendrical, numbering and measurement systems, and even when the same systems are used, the formats used to write currency amounts, dates, times, numbers and units may be different.

Regions: Technology

Linguistic specificity can be important for effective localization, so digital technologies must support levels of linguistic identification that are more granular and less granular than countries. Specific locations in digital systems are defined in terms of geographical coordinates, while regions are defined as vector polygons. The existence of region-specific toponyms and address formats make the conversion from coordinates and polygons to human-readable maps more difficult. Formatting different types of primitive values typically involves finding the right localized elements, defining the right pattern or rules, and then using an algorithm to produce value strings through token substitution. Parsing does the opposite. It converts value strings into other value types. The number of languages, regions, units, and format styles that exist can make managing all the permutations difficult.

Culture: Definition

Culture refers to the shared way of life of group of people. It is both an aggregate of individual characteristics and a source of personal identity. Cultural traditions are systems that groups of people have developed over time, in order to facilitate or improve their way of life. Traditions usually take the form of shared beliefs, values, institutions and/or practices, and they can be found in many different areas of social life. The cultural traditions that are of most interest to internationalization are:

- **Onomastic systems**. Personal and place names.
- **Numbering systems**. Digits, numbers, values and math.
- **Calendrical systems**. Eras, years, months, days, and days of the week.
- **Time-keeping systems**. Hour cycles, day periods.
- **Measurement systems**. Volume, length, weight, etc.

Culture: Diversity

A lot of cultural traditions are now shared across peoples and cultures, but many differences remain to this day. For example, kinship systems influence how you refer to and converse with other people in a society, and so there are correspondingly many different forms of personal names. Although most people today use a base ten number system, different peoples have different cultural traditions about how numbers should be grouped, and what symbols are used to represent the groupings. Similarly, although most peoples use the modern Gregorian calendar, there are still many other solar, luni-solar, seasonal, cyclic, and tabular calendars in use. Some societies keep time in terms of two twelve-hour cycles with am and pm indicators, while other use a twenty-three or twenty-four hour clock to keep time. Finally, while most countries now use the metric system, there are still some hold outs like the United States, and of course, different languages have different names and abbreviations for different units within the metric system.

Culture: Technology

Digital systems do not do well with ambiguity. For this reason, the creators of digital systems have taken several steps to remove ambiguity from their systems. They have created or adopted international standards (e.g, Unicode) that assign universal values and/or identifiers, and they have specified and implemented culturally agnostic systems (e.g., like UTC) that operate independently of any particular regional or cultural tradition. Having international standards and culturally agnostic systems that can serve as an intermediary makes it possible to convert between different cultural systems in cases when a direct conversion is not possible. Furthermore, widespread adoption of these standards and systems makes conversion less necessary.

Takeaways

- Internationalization is about creating support for different users’ localization preferences.
- The four main aspects of a user’s localization preferences are language, script, region and culture.
- Systems are the product of traditions. Language and writing are forms of systems.
- Developing software systems in an international environment requires supporting multiple sets of localization preferences
- Digital systems can support multiple sets of localization preferences by adopting international standards and culturally agnostic systems.
- This model allows an application to be multilingual and multicultural, so it can adapt to each user’s linguistic, scriptographic, regional and cultural preferences.

Learn More

- Start by looking at the RFCs for composing and matching IETF Language Tags (BCP 47) in IETF RFC 5646. Tags for Identifying Languages and IETF RFC 4647. Matching of Language Tags.

- Then take a look at how locales are defined and extended from language tags in UTS 35. Locale Data Markup Language.

- Finally, go beyond the core LDML specification and look at the specifications for different linguistic and cultural traditions.

Unicode Internationalization Technologies

How does internationalization work?

Summary

This section surveys the internationalization technologies developed by the Unicode Consortium and others. The purpose of this section is to explain how internationalization works by examining the technologies that make international software possible, and make the process of internationalization faster, easier, and generally better. Because the goal of this section is to provide a holistic view of the entire Unicode tech stack, the focus will be on explaining what each internationalization technology is, what it does, and how it contributes to the development of international software. By the end of this section, you should have a basic understanding of Unicode internationalization technologies, and be ready to learn about how to apply them in your own development work.

Objectives

- Introduce the Unicode Consortium and explain the organization’s role in the development of international software.
- Provide an overview of the technologies that make up the Unicode tech stack, and explain how they fit together.
- Describe each Unicode internationalization technology, highlight what it does, and explain how it contributes to the overall goal of making software international.
- Show how Unicode internationalization technologies are used by programing languages, platform APIs and application SDKs to provide internationalization functionality to a broad spectrum of developers.

The Unicode Consortium

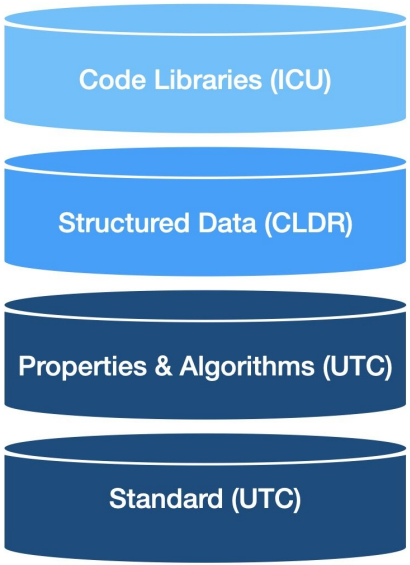
What began in 1988 as a standard for character encoding has grown into a powerful portfolio of open source standards, code, tools, libraries, and products that ensure global language support, interoperability, and resiliency across billions of devices.

Behind what most take for granted on screens today is the Unicode Consortium, the non-profit open source, open standards body for the internationalization of software and services. Unicode [technology] is embedded in every major operating system and used on more than 20 billion devices worldwide. It may be the most widely deployed technology ever.

https://unicode.org/about.html

The Unicode Internationalization Tech Stack

The standards body for the internationalization of software and services



Allow easy integration into operating system, applications, services, and more

Language-specific data for sorting, formatting, and processing of dates, times, numbers, currencies, and more

Provide character/emoji properties and behavior to ensure correct and consistent use across all platforms

Encoding of nearly 150,000 characters covering the world's languages (and emoji!)

UNICODE

The Unicode Standard

The Unicode Standard is the universal character encoding standard for written characters and text. It defines a consistent way of encoding multilingual text that enables the exchange of text data internationally and creates the foundation for global software. As the default encoding of HTML and XML, the Unicode Standard provides the underpinning for the World Wide Web and the global business environments of today. Required in new Internet protocols and implemented in all modern operating systems and computer languages such as Java and C#, Unicode is the basis of software that must function all around the world.

https://www.unicode.org/versions/Unicode17.0.0/core-spec/chapter-1/

Unicode Code Points

The Unicode Standard specifies a numeric value (code point) and a name for each of its characters. In this respect, it is similar to other character encoding standards from ASCII onward.

The Unicode Standard contains 1,114,112 code points, most of which are available for encoding of characters. The majority of the common characters used in the major languages of the world are encoded in the first 65,536 code points, also known as the Basic Multilingual Plane (BMP). The overall capacity for more than 1 million characters is more than sufficient for all known character encoding requirements, including full coverage of all minority and historic scripts of the world.

https://www.unicode.org/versions/Unicode17.0.0/core-spec/chapter-1/

UTF-8, UTF-16, UTF-32

- UTF-32 is the simplest Unicode encoding form. Each Unicode code point is represented directly by a single 32-bit code unit. Because of this, UTF-32 has a one-to-one relationship between encoded character and code unit; it is a fixed-width character encoding form.
- In the UTF-16 encoding form, non-surrogate code points in the range U+0000..U+FFFF are represented as a single 16-bit code unit; code points in the supplementary planes, in the range U+10000..U+10FFFF, are represented as pairs of 16-bit code units. These pairs of special code units are known as *surrogate pairs*.
- To meet the requirements of byte-oriented, ASCII-based systems, a third encoding form is specified by the Unicode Standard: UTF-8. This variable-width encoding form preserves ASCII transparency by making use of 8-bit code units.

https://www.unicode.org/versions/Unicode17.0.0/core-spec/chapter-2/#G13708

Unicode Character Properties

The Unicode Standard specifies a numeric value (code point) and a name for each of its characters. In this respect, it is similar to other character encoding standards from ASCII onward. In addition to character codes and names, other information is crucial to ensure legible text: a character's case, directionality, and alphabetic properties must be well defined. The Unicode Standard defines these and other semantic values, and it includes application data such as case mapping tables and character property tables as part of the Unicode Character Database. Character properties define a character's identity and behavior; they ensure consistency in the processing and interchange of Unicode data. See *Section 4.1, Unicode Character Database*.

https://www.unicode.org/versions/Unicode17.0.0/core-spec/chapter-1/

Unicode Character Algorithms

Unicode character algorithms are sets of rules for handling Unicode text. They rely on Unicode character properties defined in the Unicode Character Database. Algorithms are defined in Unicode Standard Annexes and Unicode Technical Reports.

- Normalization Forms.
- Line Breaking Algorithm.
- Bidirectional Algorithm.
- Collation Algorithm.

Locale Data Markup Language (LDML)

This document specifies an XML format for the communication of locale data: the Unicode Locale Data Markup Language (LDML). This provides a common format for systems to interchange locale data so that they can get the same results in the services provided by internationalization libraries. It also provides a standard format that can allow users to customize the behavior of a system.... Unicode LDML is also used in the Unicode Common Locale Data Repository (CLDR).

https://www.unicode.org/reports/tr35/48/tr35.html#Introduction

The Common Locale Data Repository (CLDR)

The Unicode Common Locale Data Repository (CLDR) provides key building blocks for software to support the world's languages with the largest and most extensive standard repository of locale data available.

CLDR is used by a wide spectrum of companies for their software internationalization and localization, adapting software to the conventions of different languages for such common software tasks.

- **Locale-specific patterns for formatting and parsing**
- **Translations of names**
- **Language & script information**
- **Country information**
- **Validity**

Internationalization Components for Unicode (ICU4C/ICU4J)

ICU is a mature, widely used set of C/C++ and Java libraries providing Unicode and Globalization support for software applications. ICU is widely portable and gives applications the same results on all platforms and between C/C++ and Java software. ICU is released under a nonrestrictive open source license that is suitable for use with both commercial software and with other open source or free software.

https://icu.unicode.org/home#h.i33fakvpjb7o

Internationalization Components for Unicode (ICU4C/ICU4J)

- **Code Page Conversion**. From Unicode to other encodings.
- **Collation**. Sorting according to locale-specific rules using the Unicode Collation Algorithm.
- **Formatting**. Numbers, units, currencies, dates and times, units, relative time, and so much more!
- **Time Calculations**. Several calendars available for use.
- **Unicode Support**. Characters, properties and algorithms are all tracked.
- **Regular Expressions**. Full Unicode support with improved performance.
- **Bidi**. Supports a mixture of left to right and right to left text.
- **Text Boundaries**. Boundary detection and segmentation for graphemes, words and sentences.

ICU4X

ICU4X is an implementation of Internationalization Components of Unicode (ICU) intended to be modular, performant and flexible.

The library provides a layer of APIs for all software to enable internationalization capabilities.

https://icu4x.unicode.org/2_1/tutorials/quickstart/

For the ICU4X design principles see:

https://icu4x.unicode.org/principles/

Companies and Organizations

ABAS Software, Adobe, Amazon (Kindle), Amdocs, Apache, Appian, Apple, Argonne National Laboratory, Avaya, BAE Systems Geospatial eXploitation Products, BEA, BluePhoenix Solutions, BMC Software, Boost, BroadJump, Business Objects, caris, CERN, CouchDB, Debian Linux, Dell, Eclipse, eBay, EMC Corporation, ESRI, Facebook (HHVM), Firebird RDBMS, FreeBSD, Gentoo Linux, Google, GroundWork Open Source, GTK+, Harman/Becker Automotive Systems GmbH, HP, Hyperion, IBM, Inktomi, Innodata Isogen, Informatica, Intel, Interlogics, IONA, IXOS, Jikes, Library of Congress, LibreOffice, Mathworks, Microsoft, Mozilla, Netezza, Node.js, Oracle (Solaris, Java), Lawson Software, Leica Geosystems GIS & Mapping LLC, Mandrake Linux, OCLC, Progress Software, Python, QNX, Rogue Wave, SAP, SIL, SPSS, Software AG, SuSE, Sybase, Symantec, Teradata (NCR), ToolAware, Trend Micro, Virage, webMethods, Wikimedia Foundation [Wikipedia] MediaWiki application servers, Wine, WMS Gaming, XyEnterprise, Yahoo!, Vuo, and many others.

Programming Language APIs

- Objective C
- C#
- D
- Erlang
- Cobol
- Go
- Haskell
- Lua
- Pascal
- PHP
- Python
- R
- Ruby
- Rust
- Smalltalk

Platform SDKs

- WWW (ECMAScript)
- Microsoft Windows
- Apple iOS, iPadOS, macOS
- Google Android
- IBM products
- Adobe products
- Apache open source projects

Takeaways

- The internationalization technologies developed by Unicode Consortium make internationalization possible and make international software work.
- The fundamental layer of the Unicode internationalization tech stack is the Unicode Standard. (Current version 17.0.0)
- The Unicode Character Database contains data files with Unicode character properties and other data.
- Unicode Technical Reports include annexes and specifications that define important algorithms that use Unicode Character Properties to perform character and text related functions.
- The Common Locale Data Repository (CLDR) collects locale data using the CLDR Survey Tool and publishes it using the Locale Data Markup Language (LDML)
- The Internationalization Components for Unicode (ICU) libraries use the Unicode Standard and consume the data in CLDR in order to support internationalization functions.

Learn More

- Get involved with the Unicode Consortium!
- Look through the Unicode Standard and Unicode Technical Reports to learn more about Unicode character encodings, Unicode character properties and Unicode character algorithms.
- Download the Common Locale Data Repository in LDML or view the data on github in JSON format.
- Take a look at the ICU User Guide or work through the ICU4X Quickstart.
- Learn how to leverage the ICU wrapper APIs for your platform, SDK or programming language. (JavaScript, iOS, and Android).

The Internationalization Life Cycle

What does internationalization involve?

Summary

This section examines the phases in the Software Development Life Cycle (SDLC) and outlines the internationalization work that needs to be done during each phase. The goal is to show that software internationalization is a set of architectural principles and engineering practices that must be embedded in every phase of the SDLC in order to be effective. The section begins with an overview of the SDLC and the phases included in it. It then goes through each phase and explains what each phase involves from an internationalization perspective. The section ends with a discussion of continuous internationalization and localization, and how to manage internationalization and localization expansion. By the end of this section, students should understand what it takes to actually produce international software.

Objectives

- Define the seven stages of the software development life cycle and explain the tasks typically performed in each stage.
- Step through each stage, highlighting the internationalization work that needs to be done and the reasoning behind it.
- Discuss what is meant by continuous internationalization and localization and the importance of having a internationalization program that supports it.
- Show how globalization and international expansion involves expansion in terms of languages, or expansion in terms of features, or both.

The Software Development Life Cycle

The Software Development Life Cycle (SDLC) is an idealized view of the software development process. Many different versions of the SDLC exist, but most versions include the following seven steps:

1. Research and Discovery
2. Analysis and Planning
3. Design and Architecture
4. Development and Implementation
5. Testing and Validation
6. Deployment and Release
7. Maintenance and Support

Research and Discovery

The research and discovery phase focuses on understanding the core value of the product, and how it needs to be adapted in order to satisfy the requirements of users in local contexts. Although the requirements for internationalization are always unique, there is a common process that can be used to identifying the requirements.

1. **Value**. Identify the main business goals of the product and evaluate whether the value it delivers is international or market-specific.
2. **Market research**. Gather information about different markets and user segments, in order to determine which of these might be receptive to the core value.
3. **Preferences**. Collect user and market information, so that you understand how the product would need to be adapted in order to appeal users in different contexts.
4. **Technology**. Analyze the product's current capabilities. Determine what technologies can be used to support user preferences and business requirements.

Analysis and Planning

In the analysis and planning stage, the focus is on analyzing the requirements specified in the previous phase and estimating what is needed to satisfy these requirements. For internationalization, this involves looking at the application architecture and development processes, in order to determine the feasibility, effort and time required to do the work, given the available resources.

- **Architecture**. Does the application support Unicode and other international standards like BCP 47?
- **Development process**. How big is the development organization? Are the necessary skills and resources available?
- **Work estimation**. How much effort will internationalization take? What does this mean in terms of duration and deadlines?
- **Buy in**. Does internationalization have support at the executive level? Does leadership understand what internationalization entails and why it is necessary?

Design and Architecture

The design and architecture phase focuses on specifying what will be built, and how it will operate. For internationalization, this is primarily a matter of designing the systems, frameworks and tooling that are needed to deliver a localized experience.

- **Locale awareness**. Specify how the application will understand and match each user's localization preferences, leveraging internationalization standards.
- **Runtime translation**. Figure out how the application will identify and load translated resources, and how the information will be displayed.
- **Developer experience**. Develop a framework for translation and localized formatting, that hides the details from developers, and allows them to focus on their work.
- **Tooling and infrastructure**. Design systems that manage the collection, transmission, retrieval and integration of translations. Integrate with the build process.

Development and Implementation

The development and implementation phase is where actual code is written based on the designs created in the previous phase. Internationalization requires that localization systems, frameworks and tooling must be developed before features are written or refactored.

- **Systems development**. Implement locale awareness and runtime translation management. Write mechanisms for plugging into these systems.
- **Tooling development**. Create tooling and infrastructure in parallel with system development, in order to provide a complete solution.
- **Framework development**. Wrap internationalization technologies in code that removes the need to understand internationalization in order to do it.
- **Feature development/refactoring**. Educate and guide development teams in how to refactor or write code for internationalization.

Testing and Validation

Testing and validation ensures the product functions as needed and produces the desired experience. For internationalization, several levels of testing and validation are required.

- **Unit testing**. Ensure the individual components in the application function as required, and can handle errors and edge cases.
- **Integration testing**. Ensure the localization systems and frameworks support the required end-to-end use cases and play nicely with other systems.
- **Localization testing**. Use pseudo translation to test internationalization, but conduct additional testing on actual translations.
- **User testing**. Put the complete experience in front of users and collect feedback. Iterate as necessary to achieve the best result.

Deployment and Release

Deployment involves pushing code to production, while releasing involves exposing it to users. There should be mechanisms in place for deploying without releasing. Observability and analytics should be in place to track any problems and determine impact.

- **System deployment**. The internationalization and localization systems should be deployable without causing regressions.
- **Infrastructure deployment**. Tooling and infrastructure should run independently of the build cycle but integrate with it. Avoid making localization a release bottleneck.
- **Language deployment**. It should be possible to turn localization for a particular language on or off using feature switches. Languages should not be released before they are needed.
- **Market release**. Market releases should be carefully monitored, with the ability to roll out languages and features to particular markets in incremental steps.

Maintenance and Support

Releasing code and languages is not the end of the story. Systems, frameworks, and infrastructure all require maintenance and updates as necessary. The internationalization team should provide support for developers, localizers and end users.

- **System and framework maintenance**. Keep internationalization libraries and dependencies up to date. Test each version thoroughly before integrating.
- **Infrastructure maintenance**. Perform regular audits of tooling and infrastructure, in order to detect problems early and find opportunities for optimization.
- **Internationalization support**. Support developers in the internationalization of their work. Provide regular education support and training.
- **Localization support**. Support localization program managers in localizing what developers create. Serve as a bridge between developers and non-technical localization specialists. Pay close attention to incoming user issues.

Continuous Internationalization

Once languages have been released to markets, those markets must be kept in sync with new feature development. A key element of internationalization success is ensuring that an internationalization program is in place.

- **Standards**. Establish internationalization standards and hold developers to them. Make internationalization a regular part of the development process.
- **Practices**. Ensure internationalization is done at the right time and in the right way, by defining best practices and educating developers about them.
- **Processes**. Set up internationalization processes that are easy to understand and follow. Don’t make internationalization more complicated than it needs to be.
- **Program**. Create a programs that support internationalization as business as usual. Make sure someone has ownership and is responsible for this program.

Continuous Localization

Continuous localization is all about data and automation. Automate whatever makes sense to automate, and use metrics to identify gaps and demonstrate value.

- **Localization automation**. With the right automation in place, localization can be a low-touch or no-touch effort, freeing localization program managers to focus on quality and strategy.
- **Localization analytics**. Having reliable information about localization usage and impact will make it easier to decide where to invest in order to maximize return on investment.
- **Localization quality**. Establish reasonable quality standards for different types of content. Don’t waste time over-testing the quality of materials with low user engagement and impact.

Internationalization Expansion

Internationalization facilitates international expansion by making it easier to extend localization to new features and surfaces. Expansion may require new internationalization feature development, or closer attention to particular internationalization issues.

- **Strategic internationalization**. There is more ROI in internationalizing features with high usage before features with low usage, but internationalization systems should allow for quick expansion to additional features.
- **Data-driven expansion**. To determine which features should be internationalized when, use analytics and usage data to determine where there is international demand.
- **Systematic internationalization**. Develop a repeatable playbook for internationalization expansion. This reduces time to market and saves money and resources.

Localization Expansion

Localization expansion involves adding new languages and markets to an application's localization offerings. As with internationalization, this should be done strategically, with an emphasis on metrics, usage and ROI.

- Strategic localization. Avoid the shotgun approach. Identify languages and markets that are in demand and likely to be successful.
- Data-driven expansion. Target languages used in markets where the data suggests there is demand. Compare user preferences to delivered localized experiences.
- Systematic localization. Have a clear go-to-market strategy that includes cost and usage thresholds. In B2B environments, use potential deal size and cost of localization estimates to justify adding new languages.

Takeaways

- To be effective, internationalization must be integrated into the software development life cycle and not separate from it.
- Conduct the necessary research before starting, and design systems and processes that make internationalization easier for developers.
- Be aware of the different type of internationalization and localization testing. Make sure developers and localizers understand how to test new features.
- Develop scalable systems that don’t require extensive maintenance, but keep things up to date, and test dependencies before using in production.
- Make data-driven decisions about internationalization and localization expansion. Use metrics to determine the best opportunities and learn how to estimate internationalization and localization ROI.

Learn More

There is unfortunately not a lot of good material that focuses on the practical aspects of internationalization, and how it impacts localization and globalization. The best approach is to find a mentor, and analyze successful and unsuccessful case studies.

- Learn about how internationalization and localization happens on the ground. Success is as much about studying product development as it is about studying languages.
- Participate in forums and workshops to learn about new internationalization technologies and localization processes. Don’t be afraid to ask questions. The worst that can happen is you learn something different than what you intended.
- Get involved with Unicode and other internationalization organizations. The best way to stay aware of new developments in the field, is to be there when they happen.

Review

What a long, strange trip it’s been.

Internationalization Concepts and Relationships

- The concept of a nation is ambiguous and problematic, but however you define it, international means involving two or more nations.
- International software is software that works internationally, and therefore can adapt to any user's localization preferences.
- Internationalization is the process of developing international software using Unicode internationalization technologies.
- Internationalization enables localization to scale, and can have a positive or negative impact on translation complexity and localization quality.
- Globalization is facilitated by internationalization because it makes it easier to extend localization to new features and languages.

Internationalization
Focus Areas and Concerns

- A locale is an identifier used to identify a set of user preferences that is common to a significant group of people.
- The definition of locales shows that internationalization is primarily about language, script, region and culture.
- A user’s localization preferences may extend to a preference for using particular linguistic, scriptographic, regional and cultural systems.
- Developing software in an international context makes the process more complicated, but there are recognized problems with established approaches.
- Understanding what the problems are and how to approach them makes internationalization more likely to be successful.

Unicode Internationalization Technologies

- The Unicode Consortium develops internationalization technologies that not only make internationalization possible, but also make it faster and easier.
- The Unicode internationalization tech stack is based on the Unicode character encoding standard, but includes properties, algorithms, data and libraries that are used in the majority of software platforms and products.
- Unicode technologies build on each other to create internationalization solutions that are exposed to end users through programming language API and platform SDKs
- Understanding how Unicode technologies work makes it easier to understand how internationalization in general works, but there is still a lot of additional work that goes into producing international software.

The Internationalization Life Cycle

- The software development life cycle is an idealized view of the process of developing software and has seven main stages.
- In order to be effective, internationalization needs to be an integral part of the software development life cycle. It should not be something that is done after software is already developed.
- There is internationalization work to be done in every phase of the software development life cycle, and internationalization does not stop with the release of new languages for new markets.
- The best way to learn about internationalization as it happens on the ground is to find a mentor and analyze internationalization case studies.

Conclusions

- This tutorial has tried to provide a basic understanding of what internationalization is, what it is about, how it works, and what it involves.
- Now that you have this basic understanding, you can dive deeper into Unicode internationalization technologies and determine how best to use them in your software development projects.
- Practical internationalization involves many different roles within product development and need widespread participation and support in order to be successful.
- The Unicode Technology Workshop provides an opportunity to learn from the best in the business, and take your international development to the next level.

Q&A

Thank you!

【 U+3010
シ U+30B7
メ U+30E1
ð U+10DB

☎️ U+260E
デ U+30C7
👽 U+1F47D
ق U+0642

☂️ U+2602
个 U+4E2A
 U+1F60C
Ψ U+03A8
ع U+063A
ミ U+30DF
鏡 U+93E1

ε U+03B5
○ U+3007
♺ U+267A
ζ U+03B6
く U+304F
ß U+0D26
ৰ U+09F0