Engineer

国際化

容易にローカライズできるようソフトウェアを設計すること

どのようなものか

国際化は、言語や地域ごとにコードを変更せず、一つの製品を多くの言語や地域で使えるようにする技術上の取り組みです。W3Cは、対象者の文化、地域、言語を問わず、後で容易にローカライズできるよう、製品、アプリケーション、文書を設計し、構築することと定義しています。英語ではinternationalizationのほか、internationalisationともつづります。最初のiと最後のnの間に18文字あるため、i18nと略します。同じ考え方をグローバル化（globalization）と呼ぶ人もいます。

W3Cは、役割の分担を一文で示しています。製品を設計し、構築する人は、「翻訳者やローカライズの担当者が容易にローカライズできるように」、製品を国際化します。国際化は、コードとテンプレートの中で一度行います。ローカライズは、コードが読み込むデータを通して、市場ごとに行います。完全に国際化した製品でも、一つの言語だけで出荷することはできます。

両方の作業の単位となるのが、ロケールです。CLDRの仕様では、世界の広い範囲で、利用者が共通して好む傾向のある設定を表す識別子と定義しています。言語を基本に、多くの場合は地域でさらに区別します。たとえば `en-GB` と `en-US` は言語を共有しますが、日付の書き方が異なります。この識別子は、翻訳と一連の表記慣習を同時に選びます。

W3Cは、国際化の作業を四種類に分けています。第一は、障壁の除去です。製品はテキストをUnicodeとして扱い、古い文字エンコーディングが必要な箇所では、意識して対応します。断片をつなぎ合わせて文を作らず、利用者に見せる文言にコードを依存させません。第二は、後で追加する言語で初めて必要になる機能への対応です。右から左へ書く文字と左から右へ書く文字が混在するテキストのマークアップ、文章の言語を宣言する方法、縦書きのスタイル規則などが含まれます。第三は、地域の慣習をコードから使えるようにすることです。通常は、チームが独自の表を作るより、既存のライブラリーを使います。日付と時刻の形

式、暦、数値の書式と数字の体系、一覧の並べ替え順、人名の書き方や呼びかけ方が対象です。第四は、翻訳できるものをすべてソースコードの外に置き、閲覧者ごとに適切な版を読み込めるようにすることです。

概念を判断に生かす

「国際化とは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 日付、通貨、名前の並び順まで含め、販売先のそれぞれの市場で自然に読めるソフトウェアにします。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 英語・デンマーク語のi18nシステムを構築 — 12,027 英語・デンマーク語の言語ごとのメッセージ

英語・デンマーク語のi18nシステムを構築 お問い合わせ →

なぜ重要なのか

W3Cが重視するのは、着手の時期です。一つの言語、一つの文化を前提に作った製品を世界市場向けに作り直すには、最初から世界に向けて設計するより、はるかに多くの労力と時間がかかります。W3Cは、その作業を、年はすべて19で始まると仮定した二桁の年フィールドを改修する、Y2K対応になぞらえています。国際化を後回しにすると、難しく高価な再設計になりがちです。早く始めれば、取り入れた時点ではほとんど費用のかからない、いくつかの習慣で済みます。

こうした習慣を欠いた場合の負担は、言語を増やすたびに大きくなります。コード内で三つの断片から組み立てる文は、英語の製品なら一つの不具合ですが、五つのロケールを持つ製品では五つの不具合になります。どの場合も、よりよい翻訳では直せず、ほかの変更と同様にレビューしてリリースする、コードの修正が必要です。手作業で書式を整えた日付、バイトを比較して並べ替える一覧、英語の表示文言でテストが見つけるラベルも同じです。

一度も翻訳しなくても、こうした習慣は役に立つと、W3Cは明言しています。英語だけで販売する製品にも、英語では使わない文字を含む名前、他国の慣習で配置した住所、月より先に日を書くシステムの日付が届きます。Unicodeとして保存するテキストと、表示の瞬間まで日付を日付の値のまま保つデータなら、どれも特別扱いせずに処理できます。

実際の活用例

製品が国際化されているかを最も手早く試せるのは、数を含む文です。国際化されていない実装は、コード内で文を組み立てます。

```
const label = count + (count === 1 ? " day" : " days");
```

これは英語としては正しくても、翻訳には対応できません。ウクライナ語では、整数に三つの形が必要で、小数には第四の形が要ります。1、21、101の後はдень、2、3、4、22の後はдні、0、5から20、25の後はднів、小数の後はдняです。翻訳者が二つ目の分岐にどの一語を入れても、大半の数では誤りになります。21には1と同じ形を使い、2には5と違う形を使うためです。

国際化すると、文をキーに置き換え、各言語の表に、その言語で必要なすべての形を用意します。Hugoでは、翻訳表を `i18n` ディレクトリーに置きます。言語ごとに一つのファイルを用意し、言語タグをファイル名にします。

```
# i18n/en.toml
[days]
one = '{{ . }} day'
other = '{{ . }} days'
```

```
# i18n/uk.toml
[days]
one = '{{ . }} день'
few = '{{ . }} дні'
many = '{{ . }} днів'
other = '{{ . }} дня'
```

テンプレートはキーを指定し、数を渡します。どちらの言語の単語も、複数形の規則も含みません。

```
{{ T "days" 21 }}
```

英語のページでは `21 days` 、ウクライナ語のページでは `21 день` と表示されます。2なら `2 days` と `2 дні` 、5なら `5 days` と `5 днів` です。 `one` 、 `few` 、 `many` 、 `other` という名前は、Hugo独自のものではありません。Unicode Consortiumが集めたロケールデータであるCLDRが定義する、複数形のカテゴリーです。Hugoの文書も、各言語のカテゴリーについて、CLDRの複数形規則の一覧を参照しています。新しい言語はファイルを一つ追加すれば導入でき、テンプレートの変更は不要です。

規模が大きくなると、翻訳表は製品で最も大量のテキストを持つ部分になり、コードと同じ規律が必要です。このサイトが紹介するプロジェクトの一つ、英語とデンマーク語の国際化システムは、458の名前空間ファイルに、ロケールごとに12,027件のメッセージを持ちます。各ファイルは400行を上限とし、リンターが二つのロケールのキーを比較して、一方にあるメッセージが他方に欠けることを防ぎます。デンマーク語には、くだけた呼びかけの「du」に至るまで固定した語彙があり、ツールがコミットごとに守らせます。リンク先は、この上限によって保守性を保てたと説明しています。ファイルごとの上限がなければ、458ファイル分を十二ファイルにまとめていたと述べています。

問題が起きるところ

文を断片からつなぎ合わせます。 `"Delete " + n + " files?"` として作るメッセージは、翻訳者には文全体の文脈を持たない二つの断片として届きます。その順序が必要なのは、英語だけです。ほかの言語では、数を末尾に置いたり、名詞によって動詞の形を変えたりする必要があるかもしれません。W3Cは、文字列の連結への配慮も、国際化の一部に挙げています。各文を一つのメッセージにし、翻訳者が移動できる名前付きプレースホルダーを使います。Hugoの文書では、`{{ T "age" (dict "name" "Will" "count" 1) }}` で、一つの翻訳文に名前と数の両方を埋め込む例を示しています。

コードが画面の文言を読み取ります。 「Save」というラベルでボタンを探すテストや、状態を「Paid」という文字列と比較する分岐は、最初の翻訳までは動き、その後はエラーも出さずに機能しなくなります。W3Cは、ユーザーインターフェースの文字列にコードを依存させないことを、国際化の最初の項目の一つに挙げています。キー、識別子、列挙値を比較し、テストが要素を見つけるための、安定した属性を用意します。

すべての言語に一つの複数形規則を使います。 `count === 1` は、英語の規則をコードにしたもので、製品に追加し得る言語の大半には合いません。CLDRの仕様は、zero、one、two、few、many、otherの六つのカテゴリーを定義し、必須なのはotherだけとしています。どれを選ぶかは言語ごとに規則があり、英語は二つ、ウクライナ語は四つを使います。数をメッセージに渡し、その言語で使うすべてのカテゴリーを翻訳表に持たせます。

翻訳の欠落に気付かないまま公開します。 Hugoの `T` は、現在の言語にキーがなければ既定の言語を使い、そちらにもなければ空文字列を返します。そのため、翻訳が抜けた箇所や、デンマーク語の途中に英語が交じった箇所があるまま、ページが公開されかねません。`--printI18nWarnings` を付けてビルドし、欠落

した翻訳と代替言語を使った翻訳をすべて列挙します。プレビュー環境で `enableMissingTranslationPlaceholders` を有効にして、欠落をページ上に表示し、継続的インテグレーションですべてのロケールのキーを比較します。

日付と数値の書式を手で整えます。 `MM/DD/YYYY` を書くテンプレートや、三桁ごとにコンマを入れる数値は、一つの国の慣習を固定しています。同じ日は、米国英語では `10/27/26` 、英国英語では `27/10/2026` 、デンマーク語では `27.10.2026` 、スウェーデン語では `2026-10-27` です。12,027.5は、デンマーク語では `12.027,5` と書き、ノルウェー語、スウェーデン語、ウクライナ語では、桁区切りに空白、小数点の位置にコンマを使います。CLDRの仕様は、データを言語に依存しない形で保存、交換し、閲覧者に合わせて表示形式を整えることを勧めています。日付と金額は、最後の段階まで素の値として保ち、そこでプラットフォームのロケール対応関数を使います。Hugoの `time.Format ":date_full"` は、同じ日付を、英語のページでは `Wednesday, November 3, 2021` 、ドイツ語のページでは `Mittwoch, 3. November 2021` と表示します。

テキストをUnicodeで保存しません。 「blåbærsyltetøj」をUTF-8で書き、Windows-1252として読み直す製品では、「blÃ¥bÃ¦rsyltetÃ¸j」と表示されます。古い単一言語用のエンコーディングで保存する製品では、ウクライナ語の名前をそもそも格納できません。Unicodeへの対応は、W3Cの一覧の最初の項目です。データベースからブラウザーまで、一つのUnicodeエンコーディング、通常はUTF-8を使い、バイトをテキストに変えるすべての境界で、それを明示します。

参考資料

- ローカライズと国際化 — W3Cによる両用語の実務上の定義、国際化に含む四種類の作業、最初に取り組むべき理由です。
- Hugo：多言語モード — 翻訳表、地域に応じた日付と通貨、翻訳の欠落をHugoが報告する方法です。
- Hugo：lang.Translate — 言語をまたぐ検索順序、翻訳表の複数形カテゴリー、名前付きプレースホルダーを持つメッセージです。
- UTS #35、第1部：基本事項 — CLDRの仕様です。ロケール識別子、ロケール間の継承、表示するまでデータを言語に依存させないという助言を扱います。
- UTS #35、第3部：数値 — ロケールごとの数値記号と書式、複数形カテゴリーと、その選択規則です。

用語集

実際の制約に合わせて選ぶ

業務の流れ、既存のインターフェース、変更を必要とする利用者をお知らせください。検証可能な納品内容と、お客様のチームが保守を続けられる引き継ぎ内容を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「国際化とは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/internationalization/