Engineer

Localisation

Adaptation d’un logiciel à la langue et aux usages du marché

Ce que c’est

La localisation consiste à adapter un produit à un marché particulier : sa langue et les conventions que ses lecteurs attendent pour les dates, les nombres, les prix, les noms, les adresses et l’ordre d’une liste. Le W3C la définit comme l’adaptation d’un produit, d’une application ou du contenu d’un document à un marché cible pour répondre à ses particularités linguistiques et culturelles ainsi qu’à ses autres besoins ; il appelle ce marché une locale. Le mot anglais s’écrit localization ou localisation et s’abrège en l10n, pour les 10 lettres entre le premier l et le dernier n.

Le W3C ajoute que la localisation est souvent considérée comme synonyme de traduction de l’interface et de la documentation, alors que le travail est souvent nettement plus complexe. Sa liste d’adaptations possibles couvre les formats des nombres, dates et heures ; les devises ; la disposition du clavier ; les règles de classement et de tri ; les symboles, icônes et couleurs ; les textes et images qu’une culture peut mal comprendre ou juger indélicats ; les exigences légales ; les formats d’adresses, de codes postaux et de numéros de téléphone, ainsi que l’ordre des noms ; et les besoins de lisibilité propres à chaque écriture. Si les pratiques commerciales diffèrent, comme la comptabilité dans l’exemple du W3C, l’adaptation peut exiger de repenser la logique, la conception visuelle ou la présentation du produit.

La localisation dépend de l’internationalisation et vient après elle. L’internationalisation se fait une fois, dans le code, pour n’y inscrire aucune particularité d’un marché. La localisation se fait pour chaque locale, dans les données que lit le code : les messages traduits et les conventions qui transforment une date ou un montant stocké en texte attendu par le lecteur. Un produit bien internationalisé peut accueillir une locale sans modification du code ; sinon, il faut le modifier à nouveau pour chaque marché abordé.

L’équipe n’a pas à écrire la plupart de ces données de conventions. Le projet CLDR du Consortium Unicode les rassemble pour plus de 100 langues, avec la vérification de spécialistes et de locuteurs natifs de chaque langue. Son propre exemple est la liste des fichiers d’un téléphone : le format des dates, les nombres, les unités des tailles de fichiers et l’ordre alphabétique varient selon la langue et sont généralement fournis par CLDR. Le projet indique que ses données sont intégrées à tous les systèmes d’exploitation et navigateurs modernes, ainsi qu’à des langages comme Java, C#, Swift et JavaScript. Un produit reçoit la plupart de ses formats de la plateforme, à condition que le code les lui demande au lieu de les écrire lui-même.

Utilisez le concept pour prendre une décision

Utilisez « Qu’est-ce que la localisation ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Un logiciel qui se lit naturellement sur chaque marché où vous vendez, y compris les dates, les devises et le classement des noms.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. Système i18n anglais/danois — 12,027 messages par langue, en anglais et en danois

Système i18n anglais/danois Contactez-nous →

Pourquoi c’est important

Un produit traduit sans être localisé paraît étranger. Tous les libellés sont en danois, mais la date commence encore par le mois, le prix porte encore un signe dollar et la liste des villes classe encore Aarhus à la lettre A. Les lecteurs remarquent aussitôt chacun de ces détails ; chacun leur dit que le produit a été conçu pour quelqu’un d’autre.

Certains écarts changent ce que comprend le lecteur. La chaîne 10/11/2026 signifie le 11 octobre pour un Américain et le 10 novembre pour un Britannique ; les deux dates sont valides, rien n’échoue et l’un des deux agit le mauvais jour. La chaîne 1,234 signifie mille deux cent trente-quatre en anglais et un virgule deux trois quatre en danois, soit un facteur mille sur une quantité ou un prix. Une mauvaise traduction paraît incorrecte. Une mauvaise convention paraît correcte et se lit de travers.

La liste du W3C dépasse aussi l’écran. Les obligations légales diffèrent d’un marché à l’autre, tout comme les pratiques commerciales : une locale peut donc demander d’autres conditions, d’autres informations de facturation ou un autre

parcours d'achat. Ces décisions appartiennent à l'entreprise et doivent figurer dans le plan d'une locale avant la traduction du premier message.

Le coût est propre à chaque locale et ne s'arrête pas. Chaque nouveau message doit être traduit dans chaque langue, chaque version relue dans chaque locale avant livraison, et chaque marché apporte ses conventions à vérifier. Ajouter une langue, c'est ajouter ce travail à chaque version ultérieure : entrer sur un marché est donc aussi une décision sur le processus de livraison.

À quoi cela ressemble en pratique

La plateforme fait l'essentiel du travail lorsque le code lui transmet une valeur et une locale. En JavaScript, les outils de formatage d' `Intl` prennent les deux et renvoient du texte :

```
const when = new Date("2026-10-27T12:00:00Z");
const money = { style: "currency", currency: "DKK" };
for (const locale of ["en-US", "en-GB", "da", "nb", "sv", "uk"]) {
  const date = new Intl.DateTimeFormat(locale, { dateStyle: "short" });
  const price = new Intl.NumberFormat(locale, money);
  console.log(locale, date.format(when), price.format(800));
}
```

Exécuté dans Node.js 25, qui intègre les données de CLDR 47, ce code affiche une date et un prix de 800 couronnes danoises de six façons :

```
en-US 10/27/26 DKK 800.00
en-GB 27/10/2026 DKK 800.00
da 27.10.2026 800,00 kr.
nb 27.10.2026 800,00 DKK
sv 2026-10-27 800,00 Dkr
uk 27.10.26 800,00 DKK
```

Le code n'a choisi aucun de ces détails. Le lecteur danois voit kr., le Suédois Dkr, car kr désigne la couronne suédoise en Suède, et les lecteurs norvégien et ukrainien voient le code de devise. L'ordre du jour et du mois, les séparateurs, la virgule décimale et la position de la devise viennent des données de locale. Dans la sortie réelle, l'espace entre le montant et sa devise est insécable, U+00A0, ce qui les maintient sur une même ligne.

Le tri fonctionne de même. Voici six noms de lieux danois, triés pour trois locales :

```
const towns = ["Aabenraa", "Ærø", "Åkirkeby", "Odense", "Østerbro", "Vejle"];
for (const locale of ["en", "da", "sv"]) {
  const order = new Intl.Collator(locale).compare;
  console.log(locale, towns.toSorted(order).join(", "));
}
```

```
en Aabenraa, Ærø, Åkirkeby, Odense, Østerbro, Vejle
da Odense, Vejle, Ærø, Østerbro, Aabenraa, Åkirkeby
sv Aabenraa, Odense, Vejle, Åkirkeby, Ærø, Østerbro
```

L'anglais traite æ, ø et å comme des formes de a et o. Le danois place Æ, Ø et Å après Z et classe l'ancienne graphie Aa comme Å : Aabenraa passe donc de la première à l'avant-dernière place. Le suédois place ses propres Å, Ä et Ö après Z, et classe Æ avec Ä et Ø avec Ö. Chaque ordre convient à ses lecteurs ; un Danois qui cherche Aabenraa vers la fin d'une liste triée pour l'anglais a de bonnes raisons de croire que le nom manque.

Un site Hugo dispose des mêmes outils dans ses modèles. La documentation de Hugo fait passer un montant par `lang.FormatCurrency` :

```
{{ 512.5032 | lang.FormatCurrency 2 "USD" }}
```

Une page anglaise l'affiche sous la forme $512.50, une française sous la forme 512,50 $US et une allemande sous la forme 512,50 $; `lang.FormatNumber` et `lang.FormatPercent` font de même pour les nombres ordinaires et les pourcentages.

La localisation fait partie d'une prestation proposée sur ce site. La prestation d'internationalisation et de localisation vise des logiciels qui paraissent naturels sur chaque marché où le client les vend, y compris les dates, les devises et le classement des noms. Parmi ses résultats figure un contrôle de contenu entre langues qui échoue lorsqu'une traduction perd un chiffre indiqué en anglais ou qu'une langue emploie une notation qui lui est étrangère. Il a trouvé deux descriptions danoises où manquait une métrique et seize passages ukrainiens utilisant les guillemets anglais.

Où les problèmes surviennent

La localisation est budgétée comme de la traduction. Le plan d'un nouveau marché ne couvre que les fichiers de messages : le produit sort donc dans la nouvelle langue avec les dates, prix, adresses et règles de tri de l'ancien marché, et les clients découvrent les lacunes. Planifiez chaque locale à partir de la liste complète du W3C, décidez quels éléments concernent ce marché et faites relire le produit construit par un locuteur natif de la langue concernée, plutôt que les seuls fichiers de messages.

Une langue est prise pour une locale. `en` ne dit pas si le 27 octobre s'écrit `10/27/26` ou `27/10/2026` ; la sortie ci-dessus montre que les deux locales anglaises diffèrent tant sur l'ordre que sur l'année. Un produit ne stockant que la langue doit deviner le reste et impose les conventions d'un marché à tous les autres. Conservez une balise complète telle que `en-GB` ou `da-DK` pour chaque utilisateur, tirez-la d'un réglage explicite lorsqu'il existe et transmettez-la à chaque outil de formatage.

La devise est déduite de la locale. La spécification CLDR est catégorique : un montant ne doit jamais circuler entre systèmes sans code de devise connu, sinon le même nombre 3,5 peut être « interprété comme $3.50 par un utilisateur et €3.50 par un autre ». Elle ajoute que la devise d'un montant est indépendante de la langue du lecteur et que seuls les logiciels mal conçus omettent le code pour le déduire de la locale. En Scandinavie, l'erreur est discrète, car le danois, le norvégien et le suédois abrègent chacun leur monnaie en kr : un prix stocké comme 800 signifie 800 couronnes danoises pour un lecteur et 800 couronnes norvégiennes pour le suivant. Stockez chaque montant avec son code ISO 4217 et formatez-le pour le lecteur : un Danois verra alors un prix norvégien sous la forme 800,00 NOK.

Les listes sont triées par point de code. Le `sort` par défaut de JavaScript compare les unités de code UTF-16 : il place Є et І avant le reste de l'alphabet ukrainien, et Ґ après Я, car leurs points de code sortent de la plage allant de A à я. Trié ainsi, `["Ялта", "Ґанок", "Гора", "Єнот", "Іван", "Дуб"]` donne Єнот, Іван, Гора, Дуб, Ялта, Ґанок ; avec `new Intl.Collator("uk").compare`, il donne Гора, Ґанок, Дуб, Єнот, Іван, Ялта, dans l'ordre de l'alphabet ukrainien. Triez le texte à afficher avec un comparateur de la locale du lecteur, dans le code ou par la collation de la base, et réservez l'ordre des points de code aux clés que personne ne voit.

Les tests comparent un texte formaté saisi à la main. Un test attendant `800,00 kr.` avec une espace ordinaire échoue face à l'outil de formatage, qui y place U+00A0 ; la même espace insécable sépare les milliers en norvégien, suédois et ukrainien. Les données de locale changent aussi d'une version de CLDR à l'autre : une chaîne conforme avant une mise à jour de plateforme peut ne plus l'être après. Construisez le texte attendu avec le même outil et la même locale que le code, ou vérifiez la valeur stockée et la locale transmise plutôt que la chaîne rendue.

La mise en page est dimensionnée pour l'anglais. Un bouton assez large pour afficher Save (« Enregistrer ») doit aussi afficher son équivalent ukrainien Зберегти, deux fois plus de lettres, et un menu prévu pour Settings (« Paramètres ») doit accueillir le danois Indstillinger. Une largeur fixe coupe le mot plus long ou le repousse sur une deuxième ligne ; souvent, un lecteur du marché concerné est le premier à le voir. Laissez les conteneurs grandir avec leur texte, évitez les largeurs fixes autour des mots traduits et examinez chaque locale à la largeur d'écran minimale prise en charge.

Pour aller plus loin

- Localisation et internationalisation — la définition de travail du W3C, sa liste d'adaptations possibles pour une locale et le rapport entre localisation et internationalisation.
- Hugo : mode multilingue — dates, devises, nombres et pourcentages localisés dans les modèles Hugo, avec les tables de traduction et les URL par langue.
- UTS #35, partie 3 : nombres — formats des nombres et devises de CLDR, dont la règle imposant qu'un montant circule toujours avec son code de devise.
- Projet Unicode CLDR — le projet qui rassemble les données de locale de plus de 100 langues et les fournit aux systèmes d'exploitation, navigateurs et langages de programmation.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le processus de travail, les interfaces existantes et les utilisateurs qui ont besoin du changement. Nous pouvons définir une livraison testable et une transmission permettant à votre équipe d'assurer la maintenance des éléments livrés.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce que la localisation ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/localization/