**Engineer**

# Lokalisering

at tilpasse software til ét markeds sprog og konventioner

## Hvad det er

Lokalisering er arbejdet med at tilpasse et produkt til ét bestemt marked: dets sprog og dermed de konventioner, læserne forventer for datoer, tal, priser, navne, adresser og rækkefølgen i en liste. W3C definerer det som tilpasning af et produkt, en applikation eller et dokuments indhold til de sproglige, kulturelle og øvrige krav i et bestemt målmarked, og kalder det marked en locale. På engelsk staves ordet localization eller localisation, og det forkortes l10n, for de 10 bogstaver mellem det første l og det sidste n.

W3C tilføjer, at lokalisering ofte opfattes som et synonym for oversættelse af brugerfladen og dokumentationen, og at opgaven ofte er væsentligt mere kompleks. Listen over, hvad der kan skulle tilpasses, omfatter tal-, dato- og tidsformater; valuta; tastaturlayout og brugen af det; sortering; symboler, ikoner og farver; tekst og grafik, som en kultur kan misforstå eller opfatte som taktløs; lovkrav; formatet for adresser, postnumre og telefonnumre og rækkefølgen af navne; og de krav, hver skrift stiller for at kunne læses. Hvor forretningsskikken er en anden, med regnskab som W3C's eksempel, kan tilpasningen betyde, at produktets logik, visuelle design eller præsentation skal tænkes forfra.

Lokalisering bygger på internationalisering og kommer efter den. Internationaliseringen sker én gang, i koden, så intet ved et enkelt marked er bygget ind i den. Lokaliseringen sker for hver locale, i data, som koden læser: de oversatte beskeder og de konventioner, der gør en gemt dato eller et gemt beløb til den tekst, læseren forventer. Et produkt, der er godt internationaliseret, kan få en ny locale uden en ændring i koden; et, der ikke er, skal ændres igen for hvert marked, det går ind på.

Det meste af konventionsdataene behøver teamet ikke selv at skrive. Unicode-konsortiets projekt CLDR samler dem for over 100 sprog, gennemgået af modersmålstalere og sprogeksperter, og projektets eget eksempel er fillisten på en telefon: formatet på datoerne, tallene, enhederne for filernes størrelse og den alfabetiske rækkefølge afhænger alle af sproget og kommer som regel fra CLDR. Projektet oplyser, at dets data er bygget ind i alle moderne styresystemer og browsere og i sprog som Java, C#, Swift og JavaScript. Et produkt får altså de fleste af sine formater fra platformen, forudsat at koden beder platformen om dem i stedet for at skrive dem selv.

### Brug begrebet i en beslutning

Brug »Hvad er lokalisering?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Software, der læses naturligt på hvert marked, I sælger på — også datoer, valutaer og den måde, navne sorteres på.

**Se vores ydelser** Kontakt os →

### Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har bygget et engelsk/dansk i18n-system — 12,027 beskeder pr. sprog, på engelsk og dansk

**Har bygget et engelsk/dansk i18n-system** Kontakt os →

## Hvorfor det betyder noget

Et produkt, der er oversat, men ikke lokaliseret, virker fremmed. Hver etiket står på dansk, og alligevel kommer måneden først i datoen, prisen har dollartegn, og listen over byer har Aarhus stående under A. Den danske læser bemærker hver af dem med det samme, og hver af dem fortæller, at produktet er lavet til nogle andre.

Nogle af uoverensstemmelserne ændrer, hvad læseren forstår. Strengen 10/11/2026 er 11. oktober for en amerikansk læser og 10. november for en britisk; begge er gyldige datoer, så intet fejler, og den ene af de to læsere handler på den forkerte dag. Strengen 1,234 betyder tusind to hundrede og fireogtredive på engelsk og én komma to tre fire på dansk, en faktor tusind i en mængde eller en pris. En forkert oversættelse ser forkert ud. En forkert konvention ser rigtig ud og bliver læst forkert.

W3C's liste rækker også ud over skærmen. Lovkravene er forskellige fra marked til marked, og det samme er måden, en virksomhed drives på, så en locale kan kræve andre betingelser, andre oplysninger på fakturaen eller et andet forløb gennem et køb. Det er beslutninger for forretningen, og de hører hjemme i planen for en locale, før den første besked bliver oversat.

Omkostningen er pr. locale, og den slutter ikke. Hver ny besked skal oversættes til hvert sprog, hver udgivelse skal læses igennem i hver locale, før den går ud, og hvert marked kommer med sine egne konventioner, der skal tjekkes. Et team, der tilføjer et sprog, lægger det arbejde oven i hver senere udgivelse, og derfor er beslutningen om at gå ind på et marked også en beslutning om udgivelsesprocessen.

## Sådan ser det ud i praksis

Platformen klarer det meste, når koden giver den en værdi og en locale. I JavaScript tager formatteringsobjekterne i `Intl` imod begge og returnerer tekst:

```
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));
}
```

Kørt i Node.js 25, der har data fra CLDR 47, skriver koden én dato og én pris på 800 danske kroner på seks måder:

```
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
```

Koden har ikke valgt noget af det. Den danske læser ser kr., den svenske Dkr, fordi kr i Sverige betyder svenske kroner, og den norske og den ukrainske læser ser valutakoden. Rækkefølgen af dag og måned, skilletegnene, decimalkommaet og valutaens placering kommer alle fra locale-dataene. I det rigtige output er mellemrummet mellem beløbet og valutaen et hårdt mellemrum, U+00A0, der holder de to sammen på samme linje.

Sortering virker på samme måde. Seks danske stednavne, sorteret for tre 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
```

Engelsk behandler æ, ø og å som varianter af a og o. Dansk placerer Æ, Ø og Å efter Z og sorterer den gamle stavemåde Aa som Å, så Aabenraa rykker fra første til næstsidste plads. Svensk sætter sine egne Å, Ä og Ö efter Z og sorterer Æ sammen med Ä og Ø sammen med Ö. Hver rækkefølge er rigtig for sine læsere, og en dansk læser, der leder efter Aabenraa nederst i en liste sorteret til engelsk, har god grund til at tro, at byen mangler.

Et websted bygget med Hugo har de samme værktøjer i sine skabeloner. Hugos dokumentation sender ét beløb gennem `lang.FormatCurrency`:

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

En engelsk side viser det som $512.50, en fransk som 512,50 $US og en tysk som 512,50 $, og `lang.FormatNumber` og `lang.FormatPercent` gør det samme for almindelige tal og procenter.

Lokalisering er en del af en af dette websteds ydelser. Ydelsen Internationalisering & lokalisering beskriver målet som software, der læses naturligt på hvert marked, kunden sælger på, også datoer, valutaer og den måde, navne sorteres på. Blandt resultaterne er et tværsprogligt indholdstjek, der fejler, når en oversættelse udelader et tal, som den engelske tekst angiver, eller når et sprog bruger en notation, der ikke hører til det, og som fandt to danske beskrivelser, der manglede en metrik, og seksten ukrainske tekststykker med citationstegn på engelsk vis.

# Hvor det går galt

**Lokalisering budgetteres som oversættelse.** Planen for et nyt marked dækker beskedfilerne og intet andet, så produktet lanceres på det nye sprog med det gamle markeds datoer, priser, adresser og sorteringsorden, og hullerne bliver fundet af kunderne. Planlæg hver locale ud fra hele W3C's liste, afgør, hvilke punkter der gælder for netop det marked, og lad en læser med sproget som modersmål gennemgå det færdige produkt frem for beskedfilerne.

**Et sprog forveksles med en locale.** `en` siger ikke, om 27. oktober skrives `10/27/26` eller `27/10/2026`, og outputtet ovenfor viser, at de to engelske locales er uenige om både rækkefølgen og årstallets format. Et produkt, der kun gemmer sproget, må gætte resten, og det gætter på ét markeds konventioner for alle de andre. Gem et fuldt tag som `en-GB` eller `da-DK` for hver bruger, tag det fra en udtrykkelig indstilling, hvor der er en, og giv det videre til hver eneste formatering.

**Valutaen gættes ud fra locale.** CLDR's specifikation siger det ligeud: et beløb må aldrig sendes mellem systemer uden en kendt valutakode, for ellers risikerer man, at én bruger læser tallet 3,5 som 3,50 dollar og en anden som 3,50 euro. Den tilføjer, at et beløbs valuta er uafhængig af læserens sprog, og at kun dårligt designet software udelader koden og udleder valutaen af locale. I Skandinavien går gættet galt i stilhed, fordi dansk, norsk og svensk hver forkorter deres egen valuta kr, så en pris gemt som 800 læses som 800 danske kroner af én læser og 800 norske kroner af den næste. Gem hvert beløb med sin ISO 4217-kode, og formatér det til læseren; en dansk læser ser så en norsk pris som 800,00 NOK.

**Lister sorteres efter kodepunkt.** JavaScripts standard-`sort` sammenligner UTF-16-kodeenheder, hvilket placerer Є og І før resten af det ukrainske alfabet og Ґ efter Я, fordi deres kodepunkter ligger uden for intervallet fra А til я. Sorteret på den måde bliver `["Ялта", "Ґанок", "Гора", "Єнот", "Іван", "Дуб"]` til Єнот, Іван, Гора, Дуб, Ялта, Ґанок; med `new Intl.Collator("uk").compare` bliver det Гора, Ґанок, Дуб, Єнот, Іван, Ялта, som er det ukrainske alfabets rækkefølge. Sortér tekst, der skal vises, efter læserens locale, med `Intl.Collator` i koden eller med databasens collation, og behold kodepunktsorden til nøgler, som ingen læser ser.

**Tests sammenligner med formateret tekst, der er skrevet i hånden.** En test, der forventer `800,00 kr.` med et almindeligt mellemrum, fejler mod formatteringen, som sætter U+00A0 der, og det samme hårde mellemrum

adskiller tusinderne på norsk, svensk og ukrainsk. Locale-data bliver også revideret fra én CLDR-udgave til den næste, så en streng, der passede før en opgradering af platformen, ikke nødvendigvis passer efter den. Byg den forventede tekst med den samme formatering og locale, som koden bruger, eller test på den gemte værdi og den locale, der blev givet med, frem for på den færdige streng.

**Layoutet er tegnet til engelsk.** En knap, der passer til Save, skal også kunne rumme Зберегти, med dobbelt så mange bogstaver, og en menu, der passer til Settings, skal kunne rumme det danske Indstillinger. Faste bredder skærer det længere ord af eller skubber det ned på en ny linje, og ofte er den første, der ser det, en læser på netop det marked. Lad containere vokse med deres tekst, undgå faste bredder omkring oversatte ord, og gennemgå hver locale på den smalleste skærm, produktet understøtter.

## Læs mere

- Localization vs. Internationalization — W3C's arbejdsdefinition af lokalisering, listen over, hvad en locale kan kræve tilpasset, og hvordan lokalisering forholder sig til internationalisering.
- Hugo: Multilingual mode — lokaliserede datoer, valutaer, tal og procenter i Hugos skabeloner sammen med oversættelsestabeller og URL'er pr. sprog.
- UTS #35, Part 3: Numbers — CLDR's tal- og valutaformater, herunder reglen om, at et beløb altid følges af sin valutakode.
- Unicode CLDR Project — projektet, der samler locale-data for over 100 sprog og leverer dem til styresystemer, browsere og programmeringssprog.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

**Se, hvad det koster** Kontakt os →

# Gør begreberne praktiske

Spørg, hvordan »Hvad er lokalisering?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/localization/