Engineer

Lokalisering

å tilpasse programvare til ett markeds språk og konvensjoner

Hva det er

Lokalisering er arbeidet med å tilpasse et produkt til ett bestemt marked: språket, og med det de konvensjonene leserne venter seg for datoer, tall, priser, navn, adresser og rekkefølgen i en liste. W3C definerer det som tilpasning av et produkt, en applikasjon eller innholdet i et dokument til de språklige, kulturelle og øvrige kravene i et bestemt målmarked, og kaller et slikt marked en locale. På engelsk staves ordet localization eller localisation, og det forkortes l10n, etter de 10 bokstavene mellom den første l-en og den siste n-en.

W3C legger til at lokalisering ofte blir oppfattet som et synonym for å oversette brukergrensesnittet og dokumentasjonen, og at jobben ofte er vesentlig mer sammensatt. Listen over hva som kan måtte tilpasses, omfatter tall-, dato- og tidsformater; valuta; tastaturoppsett og bruken av tastaturet; sortering; symboler, ikoner og farger; tekst og grafikk som en kultur kan misforstå eller oppfatte som upassende; lovkrav; formatet på adresser, postnumre og telefonnumre og rekkefølgen på navn; og kravene hver skrift stiller for å kunne leses. Der forretningsskikken er en annen, med regnskap som W3Cs eksempel, kan tilpasningen bety at logikken, den visuelle utformingen eller presentasjonen av produktet må tenkes på nytt.

Lokalisering bygger på internasjonalisering og kommer etter den. Internasjonaliseringen gjøres én gang, i koden, slik at ingenting ved et enkelt marked er bygd inn i den. Lokaliseringen gjøres for hver locale, i data som koden leser: de oversatte meldingene og konvensjonene som gjør en lagret dato eller et lagret beløp om til den teksten leseren venter seg. Et produkt som er godt internasjonalisert, kan få en ny locale uten en endring i koden; et som ikke er det, må endres på nytt for hvert marked det går inn i.

Teamet trenger ikke skrive det meste av konvensjonsdataene selv. Unicode-konsortiets prosjekt CLDR samler dem for over 100 språk, gjennomgått av morsmålsbrukere og språkeksperter, og prosjektets eget eksempel er fillisten på en telefon: formatet på datoene, tallene, enhetene for filstørrelsene og den alfabetiske rekkefølgen varierer alle med språket og kommer som regel fra CLDR. Prosjektet opplyser at dataene er bygd inn i alle moderne operativsystemer og nettlesere og i språk som Java, C#, Swift og JavaScript. Et produkt får altså de fleste formatene sine fra plattformen, forutsatt at koden ber plattformen om dem i stedet for å skrive dem selv.

Bruk begrepet i en beslutning

Bruk «Hva er lokalisering?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Programvare som leses naturlig i alle markeder du selger i, også datoene, valutaene og måten navn sorteres på.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Bygde et engelsk/dansk i18n-system — 12,027 meldinger per språkversjon, på engelsk og dansk

Bygde et engelsk/dansk i18n-system Ta kontakt →

Hvorfor det er viktig

Et produkt som er oversatt, men ikke lokalisert, virker fremmed. Hver etikett står på norsk, og likevel kommer måneden først i datoen, prisen har dollartegn, og stedslisten fører Ålesund opp blant A-ene. Norske lesere legger merke til hver av dem med en gang, og hver av dem forteller at produktet er laget for noen andre.

Noen av avvikene endrer hva leseren forstår. Strengen 10/11/2026 er 11. oktober for en amerikansk leser og 10. november for en britisk; begge er gyldige datoer, så ingenting feiler, og den ene av de to leserne handler på feil dag. Strengen 1,234 betyr tusen to hundre og trettifire på engelsk og én komma to tre fire på norsk, en faktor på tusen i en mengde eller en pris. En feil oversettelse ser feil ut. En feil konvensjon ser riktig ut og blir lest feil.

W3Cs liste rekker også ut over skjermen. Lovkravene er forskjellige fra marked til marked, og det samme er måten en virksomhet drives på, så en locale kan kreve andre vilkår, andre opplysninger på fakturaen eller et annet kjøpsforløp. Dette er beslutninger for forretningen, og de hører hjemme i planen for en locale før den første meldingen blir oversatt.

Kostnaden er per locale, og den tar ikke slutt. Hver ny melding må oversettes til hvert språk, hver utgivelse må leses gjennom i hver locale før den går ut, og hvert marked kommer med sine egne konvensjoner som må sjekkes. Et team som legger til et språk, legger det arbeidet oppå hver senere utgivelse, og derfor er beslutningen om å gå inn i et marked også en beslutning om utgivelsesprosessen.

Slik ser det ut i praksis

Plattformen gjør det meste når koden gir den en verdi og en locale. I JavaScript tar formateringsobjektene i `Intl` imot 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));
}
```

Kjørt i Node.js 25, som har data fra CLDR 47, skriver koden ut én dato og én pris på 800 danske kroner på seks måter:

```
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 noe av dette. Den danske leseren ser kr., den svenske Dkr, fordi kr i Sverige betyr svenske kroner, og den norske og den ukrainske leseren ser valutakoden. Rekkefølgen på dag og måned, skilletegnene, desimalkommaet og plasseringen av valutaen kommer alle fra locale-dataene. I det faktiske resultatet er mellomrommet mellom beløpet og valutaen et hardt mellomrom, U+00A0, som holder de to sammen på samme linje.

Sortering fungerer på samme måte. Seks danske stedsnavn, sortert 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 av a og o. Dansk plasserer Æ, Ø og Å etter Z og sorterer den gamle skrivemåten Aa som Å, så Aabenraa flytter fra første til nest siste plass. Med `nb` i stedet for `da` blir rekkefølgen den samme, Aa medregnet. Svensk setter sine egne Å, Ä og Ö etter Z og sorterer Æ sammen med Ä og Ø sammen med Ö. Hver rekkefølge er riktig for sine lesere, og en dansk eller norsk leser som leter etter Aabenraa nederst i en liste sortert for engelsk, har god grunn til å tro at byen mangler.

Et nettsted bygd med Hugo har de samme verktøyene i malene sine. Hugos dokumentasjon sender ett beløp gjennom `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` gjør det samme for vanlige tall og prosenter.

Lokalisering inngår i en av dette nettstedets tjenester. Tjenesten Internasjonalisering og lokalisering beskriver målet som programvare som leses naturlig i alle markeder kunden selger i, også datoene, valutaene og måten navn sorteres på. Blant resultatene er en innholdskontroll på tvers av språk som feiler når en oversettelse mister et tall den engelske teksten oppgir, eller når et språk bruker en notasjon som ikke hører hjemme i det, og som fant to danske beskrivelser som manglet en måleverdi, og seksten ukrainske tekstbiter med engelske anførselstegn.

Hvor det går galt

Lokalisering budsjetteres som oversettelse. Planen for et nytt marked dekker meldingsfilene og ingenting annet, så produktet lanseres på det nye språket med det gamle markedets datoer, priser, adresser og sorteringsrekkefølge, og hullene blir funnet av kundene. Planlegg hver locale ut fra hele W3Cs liste, avgjør hvilke punkter som gjelder for akkurat det markedet, og la en leser med språket som morsmål gå gjennom det ferdige produktet heller enn meldingsfilene.

Et språk forveksles med en locale. `en` sier ikke om 27. oktober skrives `10/27/26` eller `27/10/2026`, og resultatet ovenfor viser at de to engelske locales er uenige om både rekkefølgen og formatet på årstallet. Et produkt som bare lagrer språket, må gjette resten, og det gjetter på ett markeds konvensjoner for alle de andre. Lagre en full tagg som `en-GB` eller `nb-NO` for hver bruker, hent den fra en uttrykkelig innstilling der det finnes en, og send den videre til hver eneste formatering.

Valutaen gjettes ut fra locale. CLDRs spesifikasjon sier det rett ut: et beløp skal aldri sendes mellom systemer uten en kjent valutakode, ellers risikerer man at én bruker leser tallet 3,5 som 3,50 dollar og en annen som 3,50 euro. Den legger til at valutaen for et beløp er uavhengig av leserens språk, og at bare dårlig utformet programvare utelater koden og utleder valutaen av locale. I Skandinavia går gjetningen galt i det stille, fordi dansk, norsk og svensk hver forkorter sin egen valuta kr, så en pris lagret som 800 leses som 800 danske kroner av én leser og 800 norske kroner av den neste. Lagre hvert beløp med ISO 4217-koden sin, og formater det for leseren; en norsk leser ser da en dansk pris som 800,00 DKK, slik resultatet ovenfor viser.

Lister sorteres etter kodepunkt. JavaScripts standard-`sort` sammenligner UTF-16-kodeenheter, noe som plasserer Є og І foran resten av det ukrainske alfabetet og Ґ etter Я, fordi kodepunktene deres ligger utenfor intervallet fra А til я. Sortert på den måten blir `["Ялта", "Ґанок", "Гора", "Єнот", "Іван", "Дуб"]` til Єнот, Іван, Гора, Дуб, Ялта, Ґанок; med `new Intl.Collator("uk").compare` blir det Гора, Ґанок, Дуб, Єнот, Іван, Ялта, som er rekkefølgen i det ukrainske alfabetet. Sorter tekst som skal vises, etter leserens locale, med `Intl.Collator` i koden eller med databasens collation, og behold kodepunktrekkefølgen for nøkler ingen leser ser.

Tester sammenligner med formatert tekst skrevet for hånd. En test som forventer `800,00 kr` med et vanlig mellomrom, feiler mot formateringen, som setter U+00A0 der, og det samme harde mellomrommet skiller tusengruppene på norsk, svensk og ukrainsk. Locale-data blir også revidert fra én CLDR-utgave til den neste, så en streng som stemte før en oppgradering av plattformen, stemmer ikke nødvendigvis etter den. Bygg den forventede teksten med den samme formateringen og den samme locale som koden bruker, eller test på den lagrede verdien og den locale som ble sendt med, heller enn på den ferdige strengen.

Layouten er dimensjonert for engelsk. En knapp som har plass til Save, må også ha plass til Зберегти, med dobbelt så mange bokstaver, og en meny som har plass til Settings, må også romme det norske Innstillinger. Faste bredder kutter det lengre ordet eller skyver det ned på en ny linje, og ofte er den første som ser det, en leser i akkurat det markedet. La elementene vokse med teksten sin, unngå faste bredder rundt oversatte ord, og gå gjennom hver locale på den smaleste skjermen produktet støtter.

Les mer

- Localization vs. Internationalization — W3Cs arbeidsdefinisjon av lokalisering, listen over hva en locale kan trenge tilpasset, og hvordan lokalisering forholder seg til internasjonalisering.
- Hugo: Multilingual mode — lokaliserte datoer, valutaer, tall og prosenter i malene til Hugo, sammen med oversettelsestabeller og URL-er per språk.
- UTS #35, Part 3: Numbers — CLDRs tall- og valutaformater, blant annet regelen om at et beløp alltid følges av valutakoden sin.
- Unicode CLDR Project — prosjektet som samler locale-data for over 100 språk og leverer dem til operativsystemer, nettlesere og programmeringsspråk.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er lokalisering?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/localization/