Engineer

Lokalisering

att anpassa program till en marknads språk och konventioner

Vad det är

Lokalisering är arbetet med att anpassa en produkt till en bestämd marknad: språket, och med det de konventioner som läsarna förväntar sig för datum, tal, priser, namn, adresser och ordningen i en lista. W3C definierar det som anpassning av en produkt, en applikation eller innehållet i ett dokument till de språkliga, kulturella och övriga krav som gäller på en viss målmarknad, och kallar en sådan marknad en locale. På engelska stavas ordet localization eller localisation, och det förkortas l10n, efter de 10 bokstäverna mellan det första l:et och det sista n:et.

W3C tillägger att lokalisering ofta uppfattas som en synonym till översättning av användargränssnittet och dokumentationen, och att uppgiften ofta är betydligt mer komplex. Listan över vad som kan behöva anpassas omfattar format för tal, datum och tid; valuta; tangentbordslayout och hur tangentbordet används; sortering; symboler, ikoner och färger; text och grafik som en kultur kan missförstå eller uppfatta som stötande; lagkrav; formatet på adresser, postnummer och telefonnummer och ordningen mellan namnen; och de krav som varje skriftsystem ställer för att texten ska vara läsbar. Där affärsseden skiljer sig, med bokföring som W3C:s exempel, kan anpassningen innebära att logiken, den visuella utformningen eller presentationen av produkten måste tänkas om.

Lokalisering bygger på internationalisering och kommer efter den. Internationaliseringen görs en gång, i koden, så att ingenting som gäller en enskild marknad byggs in i den. Lokaliseringen görs för varje locale, i data som koden läser: de översatta meddelandena och de konventioner som gör om ett lagrat datum eller belopp till den text som läsaren förväntar sig. En produkt som är väl internationaliserad kan få en ny locale utan att koden ändras; en som inte är det måste ändras på nytt för varje marknad den går in på.

De flesta konventionerna behöver teamet inte ta fram själv. Unicode-konsortiets projekt CLDR samlar in dem för över 100 språk, granskade av modersmålstalare och språkexperter, och projektets eget exempel är fillistan i en telefon: formatet på datumen, talen, enheterna för filstorlekarna och den alfabetiska ordningen varierar alla med språket och kommer oftast från CLDR. Projektet uppger att dess data finns inbyggd i alla moderna operativsystem och webbläsare och i språk som Java, C#, Swift och JavaScript. En produkt får alltså de flesta av sina format från plattformen, förutsatt att koden ber plattformen om dem i stället för att skriva dem själv.

Använd begreppet i ett beslut

Använd ”Vad är lokalisering?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Mjukvara som känns naturlig på varje marknad du säljer på, inklusive datum, valutor och hur namn sorteras.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Byggde ett i18n-system för engelska och danska — 12,027 meddelanden per språkversion, på engelska och danska

Byggde ett i18n-system för engelska och danska Hör av er →

Varför det spelar roll

En produkt som är översatt men inte lokaliserad känns främmande. Varje etikett står på svenska, och ändå kommer månaden först i datumet, priset har dollartecken och ortlistan placerar Örebro bland O:na. Svenska läsare märker var och en av detaljerna direkt, och var och en säger dem att produkten är gjord för någon annan.

Vissa av avvikelserna ändrar vad läsaren förstår. Strängen 10/11/2026 är för en amerikansk läsare den 11 oktober och för en brittisk den 10 november; båda är giltiga datum, så inget fel uppstår, och en av de två läsarna agerar på fel dag. Strängen 1,234 betyder ettusen tvåhundratrettiofyra på engelska och en komma två tre fyra på svenska, en skillnad på en faktor tusen i en mängd eller ett pris. En felaktig översättning ser fel ut. En felaktig konvention ser rätt ut och läses fel.

W3C:s lista når också bortom skärmen. Lagkraven skiljer sig mellan marknader, och det gör även sättet att driva en verksamhet, så en locale kan kräva andra villkor, andra uppgifter på fakturan eller ett annat köpflöde. Det är beslut för verksamheten, och de hör hemma i planen för en locale innan det första meddelandet översätts.

Kostnaden gäller per locale, och den tar aldrig slut. Varje nytt meddelande måste översättas till varje språk, varje release måste läsas igenom i varje locale innan den går ut, och varje marknad har sina egna konventioner att kontrollera. Ett team som lägger till ett språk lägger det arbetet på varje senare release, och därför är beslutet att gå in på en marknad också ett beslut om releaseprocessen.

Så ser det ut i praktiken

Plattformen gör det mesta när koden ger den ett värde och en locale. I JavaScript tar formateringsobjekten i `Intl` emot båda och returnerar text:

```
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));
}
```

I Node.js 25, som har data från CLDR 47, skriver koden ut ett och samma datum och ett pris på 800 danska kronor på sex sätt:

```
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 inte valt något av detta. Den danska läsaren ser kr., den svenska Dkr, eftersom kr i Sverige betyder svenska kronor, och den norska och den ukrainska läsaren ser valutakoden. Ordningen mellan dag och månad, skiljetecknen, decimalkommat och valutans placering hämtas alla ur locale-data. I den faktiska utdatan är tecknet mellan beloppet och valutan ett hårt mellanslag, U+00A0, som håller ihop de två på samma rad.

Sortering fungerar på samma sätt. Sex danska ortnamn, sorterade för 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
```

Engelska behandlar æ, ø och å som varianter av a och o. Danska placerar Æ, Ø och Å efter Z och sorterar den gamla stavningen Aa som Å, så Aabenraa flyttar från första till näst sista plats. Svenska ställer sina egna Å, Ä och Ö efter Z och sorterar Æ tillsammans med Ä och Ø tillsammans med Ö, men räknar Aa som två a, så Aabenraa står först även för en svensk läsare. Varje ordning är rätt för sina läsare, och en dansk läsare som letar efter Aabenraa nära slutet av en lista sorterad för engelska har goda skäl att tro att orten saknas.

En webbplats byggd med Hugo har samma verktyg i sina mallar. Hugos dokumentation skickar ett belopp genom `lang.FormatCurrency`:

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

En engelsk sida visar det som $512.50, en fransk som 512,50 $US och en tysk som 512,50 $, och `lang.FormatNumber` och `lang.FormatPercent` gör samma sak för vanliga tal och procentsatser.

Lokalisering ingår i en av den här webbplatsens tjänster. Tjänsten Internationalisering och lokalisering beskriver målet som mjukvara som känns naturlig på varje marknad kunden säljer på, inklusive datum, valutor och hur namn sorteras. Bland resultaten finns en innehållskontroll mellan språken. Den säger ifrån när en översättning tappar en siffra som står i den engelska texten, eller när ett språk använder en notation som inte hör hemma där, och den hittade två danska beskrivningar som saknade ett mätvärde och sexton ukrainska textavsnitt som citerade på engelskt sätt.

Var det går fel

Lokalisering budgeteras som översättning. Planen för en ny marknad omfattar meddelandefilerna och inget annat, så produkten lanseras på det nya språket med den gamla marknadens datum, priser, adresser och sorteringsordning, och luckorna upptäcks av kunderna. Planera varje locale utifrån hela W3C:s lista, avgör vilka punkter som gäller just den marknaden, och låt en läsare med språket som modersmål granska den färdiga produkten snarare än meddelandefilerna.

Ett språk förväxlas med en locale. `en` säger inte om den 27 oktober skrivs `10/27/26` eller `27/10/2026`, och utdatan ovan visar att de två engelska locales är oense om både ordningen och årtalets format. En produkt som bara lagrar språket måste gissa resten, och den gissar på en marknads konventioner för alla andra. Lagra en fullständig tagg som `en-GB` eller `sv-SE` för varje användare, hämta den från en uttrycklig inställning där en sådan finns, och skicka den vidare till varje formateringsanrop.

Valutan gissas utifrån locale. CLDR:s specifikation säger det rakt ut: ett belopp får aldrig skickas mellan system utan en känd valutakod, annars riskerar man att en användare läser talet 3,5 som 3,50 dollar och en annan som 3,50 euro. Den tillägger att valutakoden för ett belopp är oberoende av läsarens språk och att bara dåligt utformad programvara saknar koden och tvingas gissa valutan utifrån locale. I Skandinavien blir gissningen fel utan att det märks, eftersom danska, norska och svenska var för sig förkortar sin egen valuta kr, så ett pris som lagrats som 800 blir 800 danska kronor för den ena läsaren och 800 svenska kronor för den andra. Lagra varje belopp med sin ISO 4217-kod och formatera det för läsaren; en svensk läsare ser då ett danskt pris som 800,00 Dkr, precis som i utdatan ovan.

Listor sorteras efter kodpunkt. Standardsorteringen i JavaScript, `sort` utan jämförelsefunktion, jämför UTF-16-kodenheter, vilket placerar Є och І före resten av det ukrainska alfabetet och Ґ efter Я, eftersom deras kodpunkter ligger utanför intervallet från A till я. Sorteras listan Ялта, Ґанок, Гора, Єнот, Іван, Дуб på det sättet blir resultatet Єнот, Іван, Гора, Дуб, Ялта, Ґанок; med `new Intl.Collator("uk").compare` blir det Гора, Ґанок, Дуб, Єнот, Іван, Ялта, vilket är ordningen i det ukrainska alfabetet. Sortera text som ska visas efter läsarens locale, med `Intl.Collator` i koden eller med databasens sorteringsregler, och behåll kodpunktsordningen för nycklar som ingen läsare ser.

Tester jämför med formaterad text som skrivits för hand. Ett test som förväntar sig `800,00 kr` med ett vanligt mellanslag misslyckas mot formateringen, som sätter U+00A0 där, och samma hårda mellanslag skiljer tusentalsgrupperna åt på norska, svenska och ukrainska. Locale-data revideras dessutom från en CLDR-version till nästa, så en sträng som stämde före en uppgradering av plattformen stämmer inte nödvändigtvis efter den. Bygg den förväntade texten med samma formatering och samma locale som koden använder, eller testa det lagrade värdet och den locale som skickades med snarare än den färdiga strängen.

Layouten är dimensionerad för engelska. En knapp som rymmer Save måste också rymma Зберегти, med dubbelt så många bokstäver, och en meny som rymmer Settings måste också få plats med det svenska Inställningar. Fasta bredder klipper av det längre ordet eller trycker ner det på en ny rad, och ofta är den första som ser det en läsare på just den marknaden. Låt elementen växa med sin text, undvik fasta bredder runt översatta ord, och granska varje locale på den smalaste skärm som produkten stöder.

Läs mer

- Localization vs. Internationalization — W3C:s arbetsdefinition av lokalisering, listan över vad en locale kan behöva få anpassat och hur lokalisering förhåller sig till internationalisering.
- Hugo: Multilingual mode — lokaliserade datum, valutor, tal och procentsatser i Hugos mallar, tillsammans med översättningstabeller och URL:er per språk.
- UTS #35, Part 3: Numbers — CLDR:s format för tal och valutor, bland annat regeln att ett belopp alltid åtföljs av sin valutakod.
- Unicode CLDR Project — projektet som samlar in locale-data för över 100 språk och levererar den till operativsystem, webbläsare och programspråk.

Ordlista

Välj utifrån era faktiska förutsättningar

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är lokalisering?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

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

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

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

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/localization/