Engineer

CLDR

Unicodes Sprach- und Regionsdaten für Formate und Sortierung

Was es ist

CLDR ist ein Bestand an Gebietsschemadaten, der Unicode ergänzt. Unicode definiert die Zeichen, in denen ein Text geschrieben wird. CLDR hält fest, welche Schreibweise Leser einer bestimmten Sprache in einer bestimmten Region für ein Datum, eine Zahl, einen Preis, eine Einheit oder eine Liste erwarten und in welcher Reihenfolge Namen sortiert sein sollen. Das Unicode Consortium pflegt beides. Der volle Name von CLDR lautet heute Unicode Common Locale Data & Rules. Früher hieß es Common Locale Data Repository, seine Spezifikation Locale Data Markup Language oder LDML. Laut Spezifikation bleiben Verweise unter den alten Namen gültig.

Die Projektwebsite nennt drei Teile. Der erste sind die Daten selbst, redaktionell gepflegt und für Software strukturiert. Der zweite ist die Spezifikation Unicode Technical Standard #35. Sie dokumentiert die Datenstruktur und ihre Verwendung durch Algorithmen und stellt Konformitätsanforderungen. Der dritte ist der Code, der Daten von Sprachexperten sammelt, auf Gültigkeit und Konsistenz prüft und in die von Entwicklern verwendeten Formate umwandelt. Die Zusammenfassung der Spezifikation liefert eine gute Übersicht über die Daten: Kennungen für Gebietsschemata, Einheiten, Zahlensysteme und Kalender; die Formatierung von Datum, Uhrzeit, Zeitzonen, Zahlen, Währungen und Messgrößen; Sortieren und Suchen. Dazu kommen Pluralregeln, Listenmuster und die Namen von Sprachen und Regionen.

Die Daten sind nach Gebietsschema organisiert: einer Sprache, die durch Schriftsystem und Region enger gefasst werden kann. Jedes Gebietsschema hält nur, was vom übergeordneten abweicht. Das Beispiel der Spezifikation ist Englisch: `en` enthält den Großteil der Daten, `en_IE` , Englisch in Irland, nur wenige Angaben wie die Währung. Über jeder Sprache steht ein Gebietsschema namens root. Das Gebietsschema für die Hauptregion einer Sprache enthält keine eigenen Daten, weil sein übergeordnetes bereits alles Nötige hält. Einige

Angaben gehören unabhängig von der Sprache allein zur Region: Währung, Maßsystem und Wochenkonventionen. Existiert etwa kein Datenpaket für Französisch in den Vereinigten Staaten, erhält eine entsprechende Anfrage ihre Formate aus `fr` und ihre Beträge in US-Dollar.

Die Quelldaten werden zu jeder Version als XML im Format der Spezifikation veröffentlicht, mit Werkzeugen zur Umwandlung in JSON- und POSIX-Formate. Der Veröffentlichungsplan sieht zwei Zyklen jährlich vor. CLDR 49 erschien am 22. Oktober 2026. Neue Daten kommen über das online verfügbare Survey Tool hinzu, das meist einmal jährlich öffnet. Es sammelt Daten für neu hinzugefügte Strukturen und korrigiert Daten früherer Versionen. Muttersprachler und Sprachexperten prüfen die eingehenden Angaben.

Das Konzept für eine Entscheidung nutzen

Nutzen Sie „Was ist CLDR?“ für eine konkrete Entscheidung. Wir können helfen, das Konzept mit dem System in Verbindung zu bringen, das Sie entwickeln oder einkaufen. Software, die in jedem Ihrer Absatzmärkte natürlich klingt — einschließlich Datumsangaben, Währungen und der Sortierung von Namen.

Unsere Leistungen entdecken Kontakt aufnehmen →

Die Annahmen hinter dem Begriff prüfen

Nutzen Sie die Erklärung und die verlinkten Arbeiten, um Fragen für Ihr Projekt zu formulieren. Wir können die Annahmen gemeinsam mit Ihnen prüfen, bevor ein Werkzeug ausgewählt wird. Englisch-dänisches i18n-System aufgebaut — 12,027 Nachrichten je Sprachversion, auf Englisch und Dänisch

Englisch-dänisches i18n-System aufgebaut Kontakt aufnehmen →

Warum es wichtig ist

Die Projektwebsite benennt das Problem klar: Ein Datum, eine Zahl, einen Geldbetrag oder eine Messgröße richtig zu schreiben ist „über verschiedene Sprachen und Regionen hinweg weit komplizierter, als den meisten Menschen bewusst ist“. CLDR gibt es, damit kein Team diese Komplexität Sprache für Sprache lernen muss. Es unterstützt über 100 Sprachen. Seine Daten werden von Muttersprachlern und Sprachexperten geprüft, gründlicher, als ein Produktteam das Datumsformat eines gerade neu erschlossenen Markts prüfen kann.

Die meisten Teams berühren CLDR nie direkt. Das ist beabsichtigt. Das Projekt zählt jedes moderne Betriebssystem und jeden modernen Browser zu den Nutzern seiner Daten sowie Sprachen wie Java, C#, .NET, Swift und JavaScript. Häufig ist die Verwendung indirekt: Eine Anwendung bittet einen Betriebssystemdienst, ein Datum zu formatieren. Dieser ruft ICU auf, Unicodes Bibliothek für den Produktiveinsatz mit C, C++, Java und Rust, und ICU liest CLDR. Node.js baut seine `Intl` -Objekte auf ICU auf und liefert standardmäßig

die vollständigen ICU-Daten mit. Python-Programme erreichen dieselben Daten über Babel, Ruby-Programme über TwitterCLDR. Ruft ein Team einen Formatter mit Gebietsschemaunterstützung auf, kommt die Antwort fast immer von CLDR. Das Team entscheidet weiterhin, welches Gebietsschema es anfordert und welche Version es verwendet.

Die Alternative ist eine selbst geschriebene Tabelle: zwölf Monatsnamen pro Sprache, ein Dezimaltrennzeichen, ein Tausendertrennzeichen, eine Alphabetreihenfolge. Bestenfalls stimmt sie am Tag ihrer Erstellung und bleibt danach eingefroren. CLDRs Survey Tool öffnet dagegen jedes Jahr erneut, um bereits veröffentlichte Daten zu korrigieren. Außerdem schreibt eine solche Tabelle meist jemand, der nur einige der enthaltenen Sprachen spricht. Wie Dänisch und Schwedisch dieselben sechs unten gezeigten Ortsnamen sortieren, würde ein Entwickler ohne diese Sprachkenntnisse nicht erraten.

Wie es in der Praxis aussieht

Fragen Sie die Laufzeitumgebung nach ihrer CLDR-Version. Lassen Sie deren Daten dann sechs dänische und schwedische Ortsnamen sortieren, drei dänische Städtenamen zu einer Liste verbinden und die Angaben zur Woche für vier Gebietsschemata nachschlagen:

```
console.log("CLDR", process.versions.cldr);

const towns = ["Ørsted", "Odense", "Åsa", "Aabenraa", "Ærø", "Ängelholm"];
console.log("code point", [...towns].sort().join(", "));
for (const tag of ["en", "da", "sv"]) {
  const sorted = [...towns].sort(new Intl.Collator(tag).compare);
  console.log(tag.padEnd(10), sorted.join(", "));
}

const three = ["Aarhus", "Odense", "Aalborg"];
for (const tag of ["en-GB", "en-US", "da", "uk"]) {
  console.log(tag.padEnd(10), new Intl.ListFormat(tag).format(three));
}

for (const tag of ["en", "en-GB", "en-DK", "da"]) {
  const locale = new Intl.Locale(tag);
  const { firstDay } = locale.getWeekInfo();
  console.log(tag.padEnd(10), locale.maximize().toString(), firstDay);
}
```

Node 25.2.1 gibt aus:

```
CLDR 47.0
code point Aabenraa, Odense, Ängelholm, Åsa, Ærø, Ørsted
en         Aabenraa, Ærø, Ängelholm, Åsa, Odense, Ørsted
da         Odense, Ängelholm, Ærø, Ørsted, Aabenraa, Åsa
sv         Aabenraa, Odense, Åsa, Ängelholm, Ærø, Ørsted
en-GB      Aarhus, Odense and Aalborg
en-US      Aarhus, Odense, and Aalborg
da         Aarhus, Odense og Aalborg
uk         Aarhus, Odense i Aalborg
en         en-Latn-US 7
en-GB      en-Latn-GB 1
en-DK      en-Latn-DK 1
da         da-Latn-DK 1
```

Die erste Zeile nennt die Version, mit der diese Laufzeitumgebung gebaut wurde: 47.0, obwohl CLDR 49 bereits erschienen ist. Eine Laufzeitumgebung enthält die Daten ihres eigenen Builds, nicht die neuesten verfügbaren Daten.

Ein einfaches `sort()` ohne Vergleichsfunktion stellt Ä, Å, Æ und Ø hinter jeden Buchstaben ohne diakritisches Zeichen, in der Reihenfolge ihrer Zeichencodes. Das passt zu keiner der drei Sprachen. Englisch sortiert jeden Namen nach seinem Grundbuchstaben. Die vier mit A beginnenden Namen kommen deshalb zuerst, Ørsted folgt auf Odense. Im Dänischen endet das Alphabet mit Æ, Ø und Å. Ä gilt als Variante von Æ, und Aabenraa mit seiner älteren Doppel-a-Schreibung wird unter den Å-Namen einsortiert. Der Sortierungsteil der Spezifikation nennt das die dänische Kontraktion „aa“. Im Schwedischen endet das Alphabet mit Å, Ä und Ö. Æ gilt als Variante von Ä und Ø als Variante von Ö. Aabenraa wird als Name mit zwei a am Anfang sortiert. Norwegisch Bokmål, `nb`, liefert die dänische Reihenfolge. Nichts davon steht im Code. Jede Reihenfolge kommt aus den CLDR-Sortierregeln für die jeweilige Sprache.

Die Listen funktionieren genauso. Jedes Gebietsschema hat ein Listenmuster mit getrennten Teilen für Anfang, Mitte und Ende einer Liste sowie für eine Zweierliste. Britisches Englisch setzt vor „and“ kein Komma, amerikanisches schon. Dänisch verbindet mit „og“, Ukrainisch mit „i“. Dafür reicht ein Aufruf ohne ein Wort aus einer dieser Sprachen.

Die letzten vier Zeilen zeigen Daten, die zu einer Region statt zu einer Sprache gehören. Jede nennt die Kennung, die von `maximize()` aus CLDRs wahrscheinlichen Unterkennungen (likely subtags) ergänzte Vollform und den ersten Wochentag als Zahl nach ISO 8601: 1 steht für Montag, 7 für Sonntag. Die Laufzeitumgebung liest ein bloßes `en` als Englisch in lateinischer Schrift in den

Vereinigten Staaten. Die Woche beginnt deshalb am Sonntag. Englisch in Großbritannien oder Dänemark beginnt wie Dänisch am Montag. Die Regeln für die Woche kommen unabhängig von der Sprache aus der Region.

Auch in einem übersetzten Text müssen die von CLDR erfassten Konventionen von Hand korrekt umgesetzt werden. Eines der Projekte dieser Website, eine sprachübergreifende Zahlenprüfung, extrahiert jede Zahl, jeden Prozentwert, jeden Multiplikator und jede Einheit aus einer englischen Zeichenfolge und verlangt sie in jeder Übersetzung. Es prüft jede Sprache außerdem auf vorgeschriebene und verbotene Konventionen, darunter Dezimaltrennzeichen, Tausendergruppierung und Anführungszeichen.

Wo es schiefgeht

Formatierter Text wird als feste Zeichenfolge verglichen. Ein Test, der `"1 500"` von `new Intl.NumberFormat("sv").format(1500)` erwartet, scheitert trotz genau so aussehender Ausgabe. Schwedisch, Norwegisch und Ukrainisch trennen Tausender mit U+00A0 NO-BREAK SPACE, das ein Editor wie ein gewöhnliches Leerzeichen anzeigt. Derselbe Test kann auch bei einer Aktualisierung scheitern, weil eine neue CLDR-Version Daten einer älteren korrigieren kann. Vergleichen Sie formatierte Ausgabe mit der Ausgabe desselben Formatters. Prüfen Sie ihre Struktur mit `formatToParts`, das die Teile als integer, group, decimal oder fraction kennzeichnet. Parsen Sie formatierten Text nie zurück in eine Zahl. Die Spezifikation empfiehlt, Daten sprachneutral zu speichern und auszutauschen und erst für den Leser zu formatieren.

Zwei Laufzeitumgebungen enthalten zwei Versionen. Node 25.2.1 enthält CLDR 47.0, während das Projekt 49 veröffentlicht hat. Ein Browser enthält, was sein eigener Build mitgebracht hat. Formatiert eine Seite ein Datum auf dem Server und nochmals im Browser, können für einen Wert zwei Zeichenfolgen erscheinen. Der Vergleich mit einem auf einer Maschine aufgezeichneten Snapshot kann auf einer anderen fehlschlagen. Formatieren Sie jeden Wert an einer Stelle, halten Sie `process.versions.cldr` neben den Test-Snapshots fest und aktualisieren Sie die Laufzeitumgebung bewusst.

Ein fehlendes Gebietsschema fällt ohne Fehler auf ein anderes zurück. `new Intl.NumberFormat("tlh")` formatiert wie amerikanisches Englisch und löst keinen Fehler aus, weil die Laufzeitumgebung keine Daten für Klingonisch besitzt und auf ihr Standardgebietsschema zurückfällt. Nodes Dokumentation beschreibt einen unauffälligeren Fall: Ein mit der Option small-icu gebautes Binärprogramm

enthält typischerweise nur englische Daten. Ein spanischer Monatsname erscheint dort als „M01“ oder „January“ statt „enero“. Prüfen Sie `resolvedOptions().locale` oder `supportedLocalesOf`, bevor Sie einem Gebietsschema vertrauen. Verwenden Sie Builds mit vollständigen ICU-Daten, wie es bei Node standardmäßig und in den offiziellen Binärpaketen der Fall ist.

Eine Kennung ohne Region bekommt die wahrscheinlichste. Die Laufzeitumgebung liest `en` als `en-Latn-US`. Ein Leser in Irland, Großbritannien oder Dänemark erhält mit `en` deshalb eine am Sonntag beginnende Woche und Datumsangaben wie 10/27/2026, mit dem Monat zuerst. Die Spezifikation beschreibt die zugrunde liegenden wahrscheinlichen Unterkennungen als heuristisch abgeleitet und weist darauf hin, dass sie sich ändern können. Geben Sie die Region des Lesers zusammen mit der Sprache weiter, sobald sie bekannt ist. Ist nur die Sprache bekannt, zeigt `maximize()`, welche Region die Laufzeitumgebung annehmen wird.

CLDR-Kennungen werden einer BCP-47-API übergeben. CLDR-Dateien und -Beispiele schreiben Gebietsschemata mit Unterstrichen, etwa `en_IE`. Die Spezifikation behandelt `_` und `-` als gleichwertig. JavaScripts `Intl` tut das nicht: `new Intl.NumberFormat("da_DK")` löst `RangeError: Invalid language tag: da_DK` aus. Konvertieren Sie an der Schnittstelle wie bei der in der Spezifikation beschriebenen Umwandlung nach BCP 47: Ersetzen Sie jedes `_` durch `-` und geben Sie das Ergebnis an `Intl.getCanonicalLocales` weiter. Es liefert `da-DK` zurück.

Weiterführende Lektüre

- Unicode CLDR Project — was CLDR ist, seine drei Bestandteile, wer es wie nutzt und der Veröffentlichungsplan.
- UTS #35, Teil 1: Kern — die Spezifikation: Gebietsschemakennungen, frühere Namen, Vererbung, wahrscheinliche Unterkennungen und die Empfehlung, Daten sprachneutral zu halten.
- UTS #35, Teil 2: Allgemeines — Anzeigenamen, Maßsysteme und die oben verwendeten Listenmuster.
- UTS #35, Teil 5: Sortierung — wie CLDR die Standardsortierung für jede Sprache anpasst, mit Dänisch als Beispiel.
- Node.js: Internationalisierungsunterstützung — wie Node seine `Intl`-Objekte auf ICU aufbaut, die Buildoptionen und die Erkennung unterstützter Funktionen eines Binärpakets.

- ECMAScript Internationalization API Specification — der Standard hinter `Intl`, einschließlich der oben verwendeten `Locale`-Methoden und ihrer CLDR-Wochendaten und wahrscheinlichen Unterkennungen.

Glossar

Für Ihre tatsächlichen Rahmenbedingungen auswählen

Beschreiben Sie den Arbeitsablauf, die vorhandenen Schnittstellen und die Nutzer, die die Veränderung benötigen. Wir können eine prüfbare Lieferung definieren und die Übergabe so gestalten, dass Ihr Team die gelieferten Bestandteile warten kann.

Preise ansehen Kontakt aufnehmen →

Die Begriffe praktisch nutzbar machen

Fragen Sie, wie „Was ist CLDR?“ auf Ihre Organisation zutrifft. Ein konkretes Beispiel aus Ihrem bestehenden System reicht aus, um das Gespräch zu beginnen.

Unsere Leistungen entdecken

welcome@engineer.company +45 50 10 61 84 Kopenhagen, Dänemark

Nehmen Sie diese Seite mit

Kontaktkarte

Mit Sorgfalt und Liebe gestaltet von D.

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

Copyright © 2025 – heute · Engineer ApS · Alle Rechte vorbehalten.

Gedruckt von https://engineer.company/de/glossary/cldr/