Engineer

CLDR

дані Unicode про дати, числа й сортування для кожної мови

Що це таке

CLDR — це зібрання даних про локалі, що доповнює Unicode. Unicode визначає символи, якими написано текст, а CLDR фіксує, як читачі певної мови в певному регіоні очікують побачити записаними дату, число, ціну, одиницю виміру чи перелік і в якому порядку вони очікують побачити відсортовані імена. І те, і інше підтримує Консорціум Unicode. Повна назва CLDR тепер Unicode Common Locale Data & Rules. Раніше проєкт називався Common Locale Data Repository, а його специфікацію називали Locale Data Markup Language, або LDML; специфікація зазначає, що посилання під старими назвами залишаються чинними.

Сайт проєкту називає три складники. Перший складник — це самі дані, відібрані й упорядковані так, щоб з ними могли працювати програми. Другий — це специфікація, технічний стандарт Unicode UTS #35, яка описує, як упорядковано дані і як їх використовують алгоритми, а також вимоги до відповідності стандарту. Третій — це код, який збирає дані від мовних фахівців, перевіряє їх на коректність і узгодженість та перетворює у формати, з якими працюють розробники. Добру карту дає підсумок самої специфікації про те, що охоплюють дані: ідентифікатори локалей, одиниць виміру, систем запису чисел і календарів; форматування дат, часу, часових поясів, чисел, валют і вимірювань; а також сортування й пошук. Поруч із ними стоять правила форм множини, шаблони списків і назви мов та регіонів.

Дані впорядковано за локалями. Локаль — це мова, яку можна уточнити системою письма і регіоном. Кожна локаль містить лише те, чим вона відрізняється від батьківської. Приклад у специфікації — англійська: `en` містить основну частину даних, а `en_IE` , англійська мова в Ірландії, лише кілька пунктів, як-от валюту. Над усіма мовами стоїть локаль root, а локаль основного регіону мови порожня, бо її батьківська локаль уже містить усе потрібне. Деякі відомості належать лише регіонові, хоч би якою була мова:

валюта, система мір і правила календарного тижня. Тож якщо окремих даних для французької у Сполучених Штатах немає, запит на таку локаль отримає формати з `fr`, а суми в доларах США.

Дані в первинному вигляді публікують для кожного випуску як XML у форматі, який визначає специфікація, разом з інструментами, що перетворюють їх у формати JSON і POSIX. Випуски виходять за графіком із двох циклів на рік, і CLDR 49 вийшов 22 жовтня 2026 року. Нові дані надходять через онлайн-інструмент Survey Tool, який зазвичай відкривають раз на рік, щоб зібрати дані для нових структур і виправити вже випущені, а те, що потрапляє до CLDR, перевіряють носії мов і мовознавці.

Застосуйте поняття до рішення

Застосуйте «Що таке CLDR?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Програма, що природно читається на кожному ринку, де ви продаєте, — з правильними датами, валютами й сортуванням імен.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Створила англо-данську систему i18n — 12,027 повідомлень на локаль — англійською та данською

Створила англо-данську систему i18n Зв'язатися з нами →

Чому це важливо

Сайт проєкту прямо називає проблему: форматувати дати, числа, валюти й одиниці виміру для різних мов і регіонів набагато складніше, ніж уявляє собі більшість людей. CLDR існує для того, щоб жодній команді не доводилося опановувати цю складність мова за мовою. Він підтримує понад 100 мов, і його дані перевіряють носії мов і мовознавці, а це більше перевірки, ніж будь-яка продуктова команда може дати формату дати на ринку, куди вона щойно вийшла.

Більшість команд ніколи не працює з CLDR напряму, і так задумано. Серед тих, хто використовує його дані, проєкт називає всі сучасні операційні системи й браузери, а також такі мови, як Java, C#, .NET, Swift і JavaScript. Часто це відбувається опосередковано: застосунок просить службу операційної системи відформатувати дату, служба викликає ICU, бібліотеку промислового рівня від Unicode для C, C++, Java і Rust, а ICU читає CLDR. Node.js будує свої об’єкти `Intl` на ICU і типово постачає повні дані ICU; програми на Python дістаються тих самих даних через Babel, а програми на

Ruby через TwitterCLDR. Коли команда викликає засіб форматування з урахуванням локалі, відповідь майже завжди надходить із CLDR. Команда ж і далі вирішує, яку локаль запитати і з яким випуском працювати.

Інший шлях — таблиця, написана вручну: дванадцять назв місяців для кожної мови, десятковий розділювач, розділювач тисяч, порядок абетки. У кращому разі така таблиця правильна в день, коли її написали, і відтоді лишається незмінною, тоді як Survey Tool проєкту щороку відкривається знову, щоб виправити вже випущені дані. До того ж пише її зазвичай людина, яка знає лише деякі з мов у ній. Порядок, у якому данська і шведська сортують ті самі шість назв населених пунктів, показаний нижче, не вгадає розробник, який не знає цих мов.

Як це виглядає на практиці

Запитайте в середовища виконання, який випуск CLDR воно має, потім дайте цим даним відсортувати шість данських і шведських назв населених пунктів, об’єднати в список три назви данських міст і визначити перший день тижня для чотирьох локалей:

```
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 виводить:

```
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
```

Перший рядок показує випуск, з яким зібрано це середовище виконання. Це 47.0, хоча CLDR 49 уже вийшов: середовище виконання має дані своєї збірки, а не найновіші дані, які існують.

Звичайний `sort()` без функції порівняння ставить Ä, Å, Æ і Ø після всіх літер від A до Z, у порядку їхніх кодів символів, і це не відповідає жодній із трьох мов. Англійська сортує кожну назву за базовою літерою, тож чотири назви на A йдуть першими, а Ørsted іде після Odense. Данська закінчує абетку літерами Æ, Ø і Å, вважає Ä варіантом Æ і ставить Aabenraa, назву, записану за старим правописом через подвійне a, серед назв на Å; частина специфікації про сортування називає це данською контракцією «aa». Шведська закінчує абетку літерами Å, Ä і Ö, вважає Æ варіантом Ä, а Ø варіантом Ö, і сортує Aabenraa як назву, що починається з двох a. Норвезький букмол, `nb`, дає той самий порядок, що й данська. Нічого з цього в коді немає: кожен порядок береться з правил сортування CLDR для відповідної мови.

Списки працюють так само. Кожна локаль має шаблон списку з окремими частинами для початку, середини й кінця списку, а також для списку з двох елементів. Британська англійська не ставить кому перед «and», американська ставить, данська з'єднує елементи словом «og», а українська словом «і», і все це дає один виклик, у якому немає жодного слова з цих мов.

Останні чотири рядки показують дані, які належать регіонові, а не мові. Кожен рядок містить тег, повну форму, яку `maximize()` доповнює з даних CLDR про найімовірніші систему письма й регіон, і перший день тижня як число за ISO 8601, де 1 означає понеділок, а 7 неділю. Тег `en` без регіону середовище виконання тлумачить як англійську мову латинським письмом у

варіанті США, тож тиждень починається з неділі. Для англійської у Великій Британії чи в Данії тиждень починається з понеділка, як і для данської: тиждень залежить від регіону, а не від мови.

Ті самі норми, що їх фіксує CLDR, доводиться вручну витримувати й у перекладеному тексті. Один із проєктів на цьому сайті, міжмовна перевірка чисел, вибирає з англійського рядка кожне число, відсоток, множник та одиницю виміру й вимагає, щоб кожен із цих елементів був у кожному перекладі. Вона також перевіряє кожну мову на норми, яких треба дотримуватися, і на ті, яких дотримуватися не можна, зокрема щодо десяткового розділювача, групування розрядів і лапок.

Де виникають проблеми

Відформатований текст порівнюють як незмінний рядок. Тест, що очікує `"1 500"` від `new Intl.NumberFormat("sv").format(1500)`, падає, хоча результат виглядає саме так, бо шведська, норвезька й українська розділяють тисячі символом U+00A0 NO-BREAK SPACE, нерозривним пробілом, який редактор показує як звичайний. Той самий тест може зламатися й після оновлення, бо новий випуск CLDR може виправити дані, які постачав попередній. Порівнюйте відформатований результат із результатом того самого засобу форматування, перевіряйте його будову через `formatToParts`, який позначає кожну частину як integer, group, decimal чи fraction, і ніколи не розбирайте відформатований текст назад у число. Специфікація радить зберігати й передавати дані в мовно-нейтральному вигляді і форматувати їх для читача.

Два середовища виконання мають два різні випуски. Node 25.2.1 постачається з CLDR 47.0, тоді як проєкт уже випустив 49, а браузер має той випуск, з яким його зібрали. Сторінка, що форматує дату на сервері, а потім ще раз у браузері, може показати для одного значення два різні рядки, а знімок, записаний на одній машині, може не пройти на іншій. Форматуйте кожне значення в одному місці, записуйте `process.versions.cldr` поруч зі знімками тестів і оновлюйте середовище виконання свідомо.

Відсутня локаль непомітно підмінюється іншою. `new Intl.NumberFormat("tlh")` форматує як американська англійська і не кидає винятку, бо середовище виконання не має даних для клінгонської і переходить до своєї типової локалі. Документація Node описує ще менш помітний випадок: двійковий файл, зібраний з опцією small-icu, зазвичай містить лише англійські дані, і там іспанська назва місяця виводиться як

«M01» або «January» замість «enero». Перевіряйте `resolvedOptions().locale` або `supportedLocalesOf`, перш ніж довіряти локалі, і збирайте з повними даними ICU, які в Node є типовими і які мають офіційні двійкові файли.

Тег без регіону отримує найімовірніший. Середовище виконання тлумачить `en` як `en-Latn-US`, тож читач в Ірландії, Великій Британії чи Данії, якому віддали `en`, побачить тиждень, що починається з неділі, і дати у вигляді 10/27/2026, з місяцем на першому місці. Специфікація описує дані likely subtags, на яких це ґрунтується, як виведені евристично й зазначає, що вони можуть змінюватися з часом. Передавайте регіон читача разом із мовою, коли він відомий, а коли відома лише мова, викликайте `maximize()`, щоб побачити, який регіон припустить середовище виконання.

Ідентифікатори CLDR передають в API, що чекає BCP 47. Файли й приклади CLDR записують локалі з підкресленням, як-от `en_IE`, і специфікація вважає `_` і `-` рівнозначними. `Intl` у JavaScript так не вважає: `new Intl.NumberFormat("da_DK")` кидає `RangeError: Invalid language tag: da_DK`. Перетворюйте ідентифікатор на межі так само, як це робить власне перетворення специфікації на BCP 47, замінюючи кожне `_` на `-`, і пропускайте результат через `Intl.getCanonicalLocales`, який повертає `da-DK`.

Що почитати далі

- Unicode CLDR Project — що таке CLDR, три його складники, хто й як використовує його дані, і графік випусків.
- UTS #35, Part 1: Core — специфікація: ідентифікатори локалей, колишні назви, успадкування між локалями, likely subtags і порада тримати дані мовно-нейтральними.
- UTS #35, Part 2: General — назви мов і регіонів для показу, системи мір і шаблони списків, використані вище.
- UTS #35, Part 5: Collation — як CLDR пристосовує типовий порядок сортування до кожної мови, з данською як прикладом.
- Node.js: Internationalization support — як Node будує свої об’єкти `Intl` на ICU, параметри збирання і як дізнатися, що підтримує двійковий файл.
- ECMAScript Internationalization API Specification — стандарт, на якому стоїть `Intl`, зокрема методи `Locale`, використані вище, і дані CLDR про тиждень та likely subtags, на які вони спираються.

Словник

Обирайте за фактичними умовами

Опишіть робочий процес, наявні інтерфейси та користувачів, яким потрібна зміна. Ми можемо визначити перевірюваний результат і спосіб передачі, який дасть вашій команді змогу підтримувати його.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке CLDR?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

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

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/cldr/