Engineer

CLDR

datos de Unicode de formato y ordenación por idioma y región

Qué es

CLDR es un conjunto de datos de configuración regional que complementa Unicode. Unicode define los caracteres con los que se escribe un texto; CLDR registra cómo esperan los lectores de un idioma y una región que se escriban una fecha, un número, un precio, una unidad o una lista, y en qué orden esperan que se ordenen los nombres. Unicode Consortium mantiene ambos. El nombre completo actual de CLDR es Unicode Common Locale Data & Rules. Antes se llamaba Common Locale Data Repository y su especificación se llamaba Locale Data Markup Language, o LDML; la especificación indica que las referencias con los nombres antiguos siguen siendo válidas.

El sitio del proyecto distingue tres partes. La primera son los propios datos, seleccionados y estructurados para que los utilice el software. La segunda es la especificación, Unicode Technical Standard #35, que documenta cómo se estructuran los datos y cómo los utilizan los algoritmos, con requisitos de conformidad. La tercera es el código que recoge los datos de especialistas en idiomas, comprueba su validez y coherencia y los convierte a los formatos que utilizan los desarrolladores. El resumen de la propia especificación ofrece un buen mapa de lo que cubren los datos: los identificadores de configuraciones regionales, unidades, sistemas de numeración y calendarios; el formato de fechas, horas, zonas horarias, números, monedas y medidas; y la ordenación y búsqueda. Las reglas de plural, los patrones de listas y los nombres de idiomas y regiones los acompañan.

Los datos se organizan por configuración regional: un idioma que puede concretarse mediante una escritura y una región. Cada configuración solo contiene lo que difiere de su configuración superior. El ejemplo de la especificación es el inglés: `en` contiene la mayor parte de los datos y `en_IE`, el inglés utilizado en Irlanda, solo algunos elementos, como la moneda. Por encima de todos los idiomas hay una configuración llamada root, y la configuración de la

región principal de un idioma no tiene datos propios porque su configuración superior ya contiene lo que necesita. Algunos datos pertenecen únicamente a la región, sea cual sea el idioma: la moneda, el sistema de medidas y las convenciones de la semana. Por eso, si no existe un paquete para el francés utilizado en Estados Unidos, una solicitud de esa configuración recibe los formatos de `fr` y los importes en dólares estadounidenses.

Los datos de origen se publican en cada versión como XML en el formato que define la especificación, con herramientas para convertirlos a JSON y POSIX. Las publicaciones siguen un calendario de dos ciclos al año, y CLDR 49 se publicó el 22 de octubre de 2026. Los datos nuevos llegan mediante una herramienta en línea llamada Survey Tool, que suele abrirse una vez al año para recoger datos de estructuras recién añadidas y corregir los de versiones anteriores, y hablantes nativos y especialistas lingüísticos revisan lo que se incorpora.

Utilice el concepto en una decisión

Aplique «¿Qué es CLDR?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Software que se lee con naturalidad en cada mercado donde vende, incluidas las fechas, las monedas y la forma de ordenar los nombres.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. i18n en inglés y danés — 12,027 mensajes por configuración regional en inglés y danés

i18n en inglés y danés Póngase en contacto →

Por qué importa

El sitio del proyecto expresa el problema con claridad: escribir correctamente una fecha, un número, una cantidad de dinero o una medida es «mucho más complicado entre distintos idiomas y regiones de lo que la mayoría de las personas imagina». CLDR existe para que ningún equipo tenga que aprender esa complejidad idioma por idioma. Admite más de 100 idiomas y sus datos los revisan hablantes nativos y especialistas lingüísticos, una revisión mayor de la que cualquier equipo de producto puede dedicar al formato de fecha de un mercado al que acaba de entrar.

La mayoría de los equipos nunca toca CLDR directamente, y eso es deliberado. El proyecto incluye entre quienes utilizan sus datos a todos los sistemas operativos y navegadores modernos, además de lenguajes como Java, C#, .NET, Swift y JavaScript. El uso suele ser indirecto: una aplicación pide a un servicio del sistema operativo que dé formato a una fecha, el servicio llama a ICU, la biblioteca de

Unicode para uso en producción con C, C++, Java y Rust, e ICU lee CLDR. Node.js construye sus objetos `Intl` sobre ICU e incluye de forma predeterminada todos los datos de ICU; los programas Python acceden a los mismos datos mediante Babel y los de Ruby mediante TwitterCLDR. Cuando un equipo llama a un formateador que tiene en cuenta la configuración regional, la respuesta casi siempre procede de CLDR. Lo que aún decide el equipo es qué configuración solicitar y qué versión está ejecutando.

La alternativa es una tabla escrita a mano: doce nombres de mes por idioma, un separador decimal, uno de miles y un orden para el alfabeto. En el mejor caso, es correcta el día en que se escribe y queda congelada desde entonces, mientras la Survey Tool de CLDR vuelve a abrirse cada año para corregir datos ya publicados. Además, suele escribirla alguien que solo habla algunos de los idiomas que contiene. Los órdenes en que el danés y el sueco clasifican los mismos seis topónimos que aparecen a continuación no son algo que adivinaría un desarrollador ajeno a esos idiomas.

Cómo se aplica en la práctica

Pregunte al entorno de ejecución qué versión de CLDR incluye y deje que esos datos ordenen seis topónimos daneses y suecos, unan tres nombres de ciudades danesas en una lista y consulten la semana de cuatro configuraciones regionales:

```
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 imprime:

```
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
```

La primera línea indica la versión con la que se construyó ese entorno de ejecución. Es 47.0, aunque CLDR 49 ya está publicado: un entorno incorpora los datos de su propia compilación, no los más recientes que existen.

Un `sort()` simple sin comparador sitúa Ä, Å, Æ y Ø después de todas las letras sin marcas, en el orden de sus códigos de carácter, que no coincide con ninguno de los tres idiomas. El inglés ordena cada nombre por su letra base, de modo que los cuatro nombres que empiezan por A van primero y Ørsted sigue a Odense. El danés termina su alfabeto con Æ, Ø y Å, trata Ä como una variante de Æ y coloca Aabenraa, escrito con la antigua grafía de doble a, entre los nombres con Å; la parte de ordenación de la especificación lo llama contracción danesa «aa». El sueco termina su alfabeto con Å, Ä y Ö, trata Æ como una variante de Ä y Ø como una variante de Ö, y ordena Aabenraa como un nombre que empieza por dos aes. El bokmål noruego, `nb`, devuelve el orden danés. Nada de esto está en el código: cada orden procede de las reglas de ordenación de CLDR para ese idioma.

Las listas funcionan igual. Cada configuración regional tiene un patrón de lista con partes separadas para el principio, el medio y el final de una lista, y para una lista de dos elementos. El inglés británico no pone coma antes de «and» y el estadounidense sí, el danés enlaza con «og» y el ucraniano con «i», todo a partir de una llamada que no contiene ninguna palabra de esos idiomas.

Las últimas cuatro líneas muestran datos que pertenecen a una región, no a un idioma. Cada una da la etiqueta, la forma completa que `maximize()` rellena a partir de las subetiquetas probables de CLDR y el primer día de la semana como número ISO 8601, donde 1 es lunes y 7 domingo. El entorno interpreta un `en` sin más como inglés en escritura latina utilizado en Estados Unidos, por lo que la semana comienza en domingo. El inglés de Gran Bretaña o de Dinamarca comienza la semana en lunes, igual que el danés: la semana procede de la región, sea cual sea el idioma.

Las convenciones que registra CLDR también son las que un texto traducido debe respetar a mano. Uno de los proyectos de este sitio, una comprobación de cifras entre idiomas, extrae de una cadena en inglés cada número, porcentaje, multiplicador y unidad y exige su presencia en todas las traducciones. También comprueba cada idioma según las convenciones que debe utilizar y las que debe evitar, incluidos los separadores decimales, la agrupación de miles y las comillas.

Dónde surgen los problemas

El texto formateado se compara como una cadena fija. Una prueba que espera `"1 500"` de `new Intl.NumberFormat("sv").format(1500)` falla aunque la salida tenga exactamente ese aspecto, porque el sueco, el noruego y el ucraniano separan los miles con U+00A0 NO-BREAK SPACE, que un editor muestra como un espacio normal. La misma prueba también puede romperse tras una actualización porque una versión nueva de CLDR puede corregir datos de una anterior. Compare la salida formateada con la salida del mismo formateador, compruebe su estructura con `formatToParts`, que etiqueta cada parte como integer (entero), group (separador de grupos), decimal (separador decimal) o fraction (parte fraccionaria), y nunca convierta el texto formateado de nuevo en un número. La especificación aconseja almacenar e intercambiar datos neutrales respecto al idioma y darles formato para quien los lea.

Dos entornos de ejecución incorporan dos versiones. Node 25.2.1 incluye CLDR 47.0 mientras el proyecto ya ha publicado 49, y un navegador incorpora lo que incluyera su propia compilación. Una página que da formato a una fecha en el servidor y de nuevo en el navegador puede mostrar dos cadenas para un valor, y una instantánea registrada en una máquina puede fallar en otra. Dé formato a cada valor en un solo lugar, registre `process.versions.cldr` junto a las instantáneas de prueba y actualice el entorno de ejecución de forma deliberada.

Una configuración regional ausente recurre a otra sin dar error. `new Intl.NumberFormat("tlh")` utiliza inglés estadounidense sin lanzar ninguna excepción porque el entorno no tiene datos de klingon y vuelve a su configuración predeterminada. La documentación de Node describe un caso más discreto: un binario construido con la opción small-icu suele incluir solo datos del inglés, y allí un nombre de mes español se muestra como «M01» o «January» en lugar de «enero». Compruebe `resolvedOptions().locale` o `supportedLocalesOf` antes de confiar en una configuración y ejecute compilaciones con todos los datos de ICU, que es el valor predeterminado de Node y lo que utilizan sus binarios oficiales.

Una etiqueta sin región recibe la más probable. El entorno interpreta `en` como `en-Latn-US`, de modo que un lector en Irlanda, Gran Bretaña o Dinamarca al que se le sirve `en` recibe una semana que empieza en domingo y fechas como 10/27/2026, con el mes primero. La especificación describe los datos de subetiquetas probables que lo sustentan como derivados mediante heurísticas y señala que pueden cambiar con el tiempo. Pase la región del lector junto con el idioma cuando la conozca y, si solo conoce el idioma, llame a `maximize()` para ver qué región supondrá el entorno.

Se pasan identificadores de CLDR a una API BCP 47. Los archivos y ejemplos de CLDR escriben las configuraciones regionales con guiones bajos, como `en_IE`, y la especificación considera equivalentes `_` y `-`. `Intl` de JavaScript no lo hace: `new Intl.NumberFormat("da_DK")` lanza `RangeError: Invalid language tag: da_DK`. Convierta en el límite entre ambos formatos, como hace la propia conversión de la especificación a BCP 47, sustituyendo cada `_` por `-`, y pase el resultado por `Intl.getCanonicalLocales`, que devuelve `da-DK`.

Lecturas adicionales

- Proyecto Unicode CLDR — qué es CLDR, sus tres componentes, quién lo utiliza y cómo, y el calendario de versiones.
- UTS #35, parte 1: núcleo — la especificación: identificadores de configuración regional, nombres anteriores, herencia entre configuraciones, subetiquetas probables y el consejo de mantener los datos neutrales respecto al idioma.
- UTS #35, parte 2: general — nombres mostrados, sistemas de medidas y los patrones de listas utilizados arriba.
- UTS #35, parte 5: ordenación — cómo adapta CLDR el orden predeterminado a cada idioma, con el danés como ejemplo.

- Node.js: soporte de internacionalización — cómo construye Node sus objetos `Intl` sobre ICU, las opciones de compilación y cómo detectar lo que admite un binario.
- Especificación de la API de internacionalización de ECMAScript — el estándar en que se basa `Intl`, incluidos los métodos de `Locale` utilizados arriba y los datos de CLDR sobre la semana y las subetiquetas probables.

Glosario

Elija según sus restricciones reales

Describa el flujo de trabajo, las interfaces existentes y los usuarios que necesitan el cambio. Podemos definir una entrega comprobable y un traspaso que permita a su equipo mantener lo entregado.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es CLDR?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

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

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/cldr/