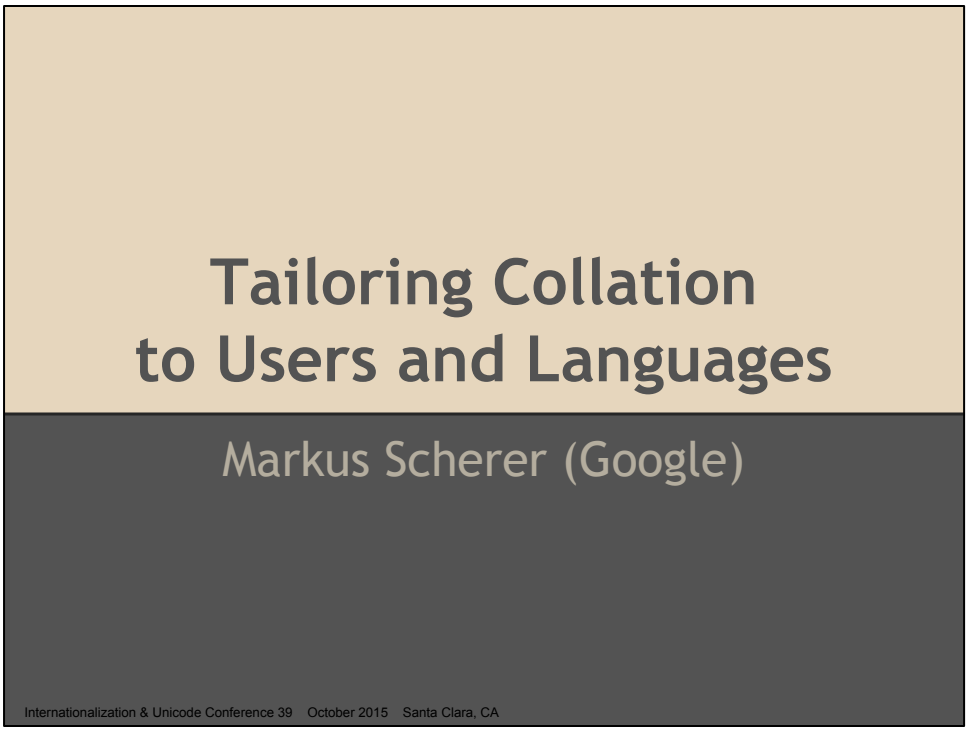



This interactive session shows how to use Unicode and CLDR collation algorithms and data for multilingual sorting and searching. Parametric collation settings - "ignore punctuation", "uppercase first" and others - are explained and their effects demonstrated. Then we discuss language-specific sort orders and search comparison mappings, why we need them, how to determine what to change, and how to write CLDR tailoring rules for them.

We will examine charts and data files, and experiment with online demos. On request, we can discuss implementation techniques at a high level, but no source code shall be harmed during this session.

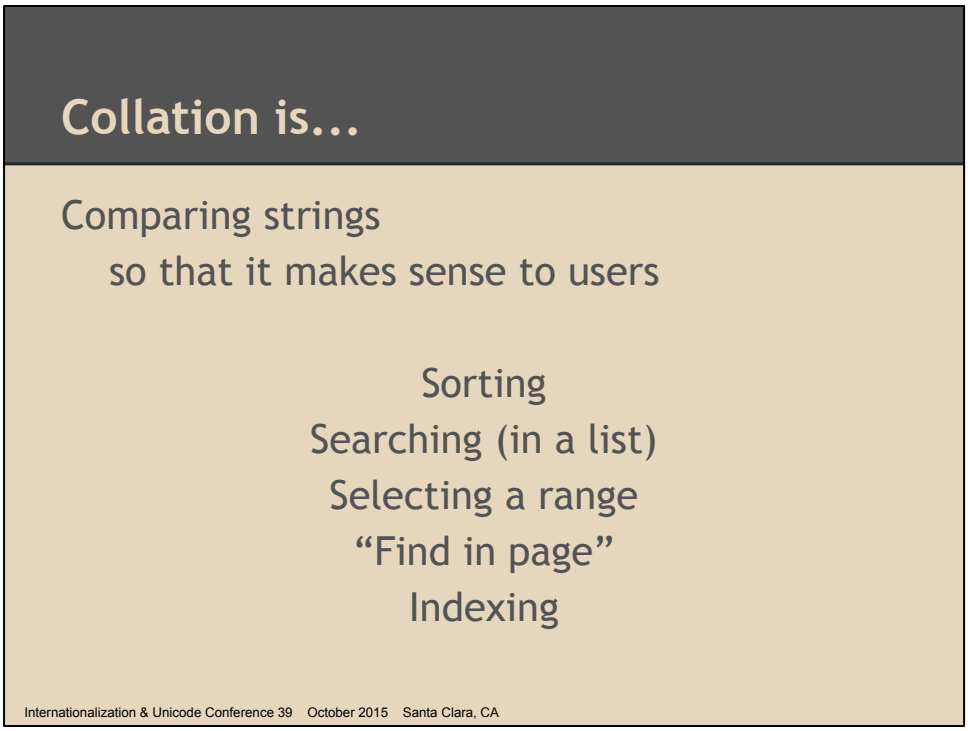
Ask the audience:

- How familiar with Unicode/UCA/CLDR collation?
- More examples from CLDR, or more working on requests/issues from audience members?

About myself:

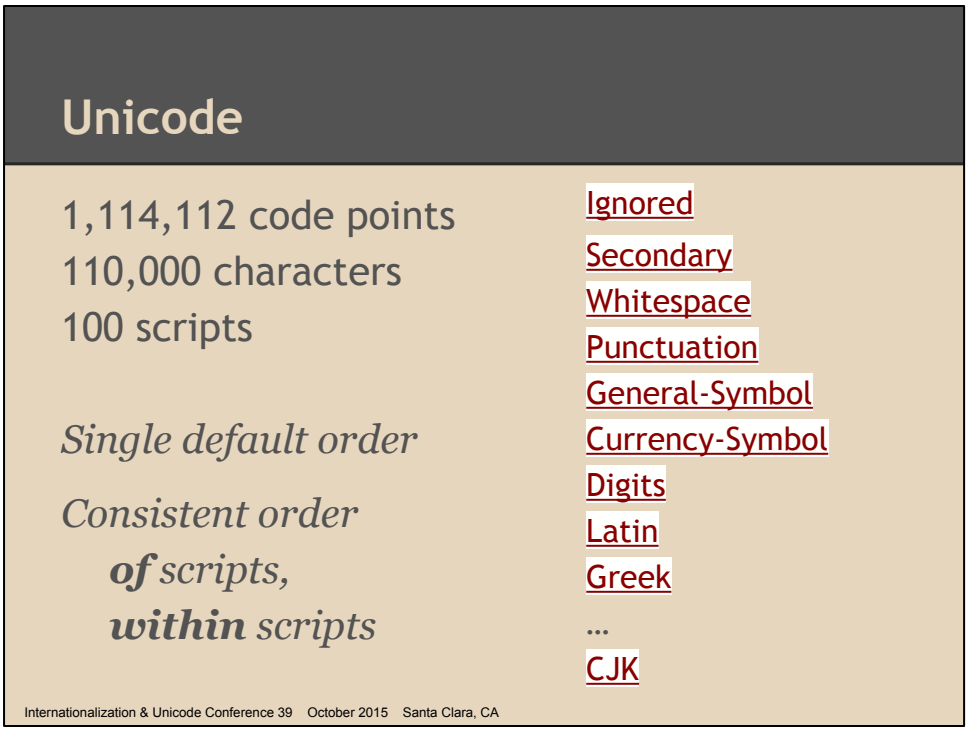
- 16 years ICU team member
- Co-designed data structures for the ICU 1.8 collation implementation (live in 2001)
- Re-wrote ICU collation 2012..2014, live in ICU 53
- Became maintainer of UTS #10 (UCA) and LDML collation spec (CLDR)

- Fixed bugs, clarified spec, added features to LDML



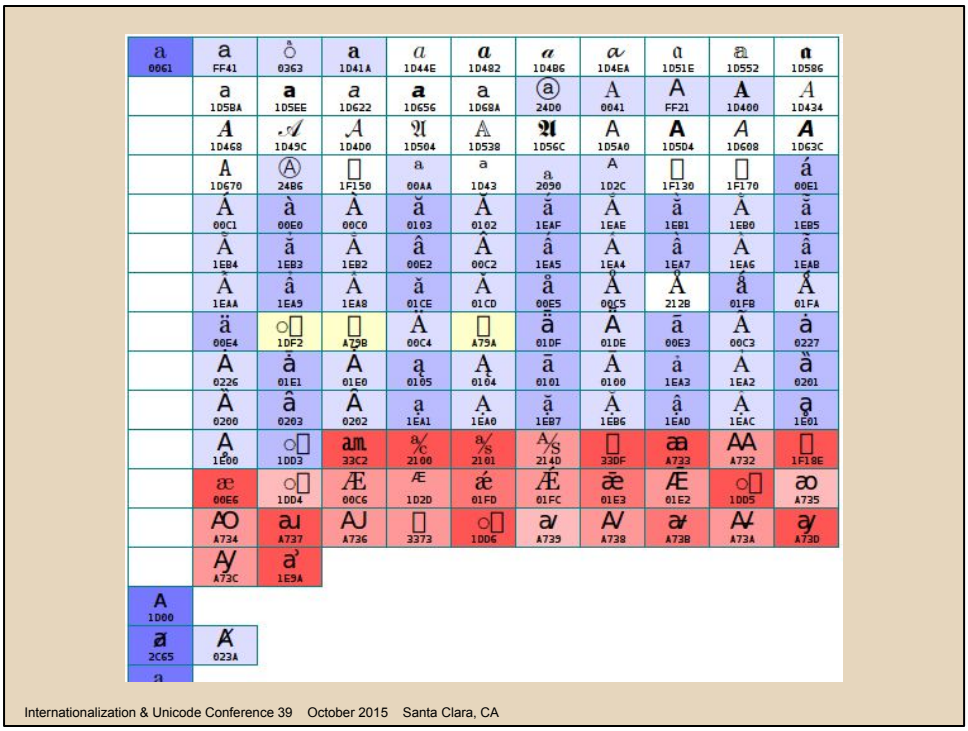

“Collation is the assembly of written information into a standard order. Many systems of collation are based on numerical order or alphabetical order, or extensions and combinations thereof.” (http://en.wikipedia.org/wiki/Collation)

“Collation is the general term for the process and function of determining the sorting order of strings of characters. It is a key function in computer systems; whenever a list of strings is presented to users, they are likely to want it in a sorted order so that they can easily and reliably find individual strings.” (UTS #10 (UCA): http://www.unicode.org/reports/tr10/)




It is relatively easy to define one sort order for one language and its writing system.

Unicode has a large number of code points, and a large number of assigned characters for a large number of varied writing systems. The standard defines one sort order that covers all of them.

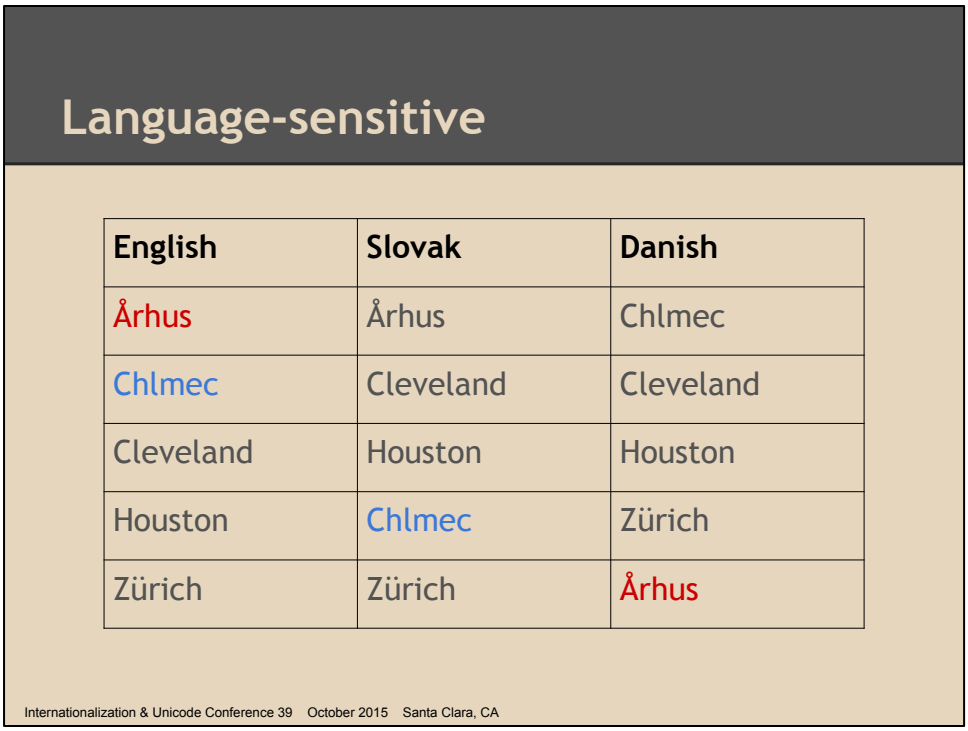



Default Unicode Collation Element Table
http://www.unicode.org/reports/tr10/#Default_Unicode_Collation_Element_Table

CLDR Root collation
http://www.unicode.org/reports/tr35/tr35-collation.html#Root_Collation

Charts
http://www.unicode.org/charts/collation/
http://www.unicode.org/charts/collation/chart_Latin.html

Note: The sort order is *independent of the character codes*. Code point order is never useful for presenting lists to users.



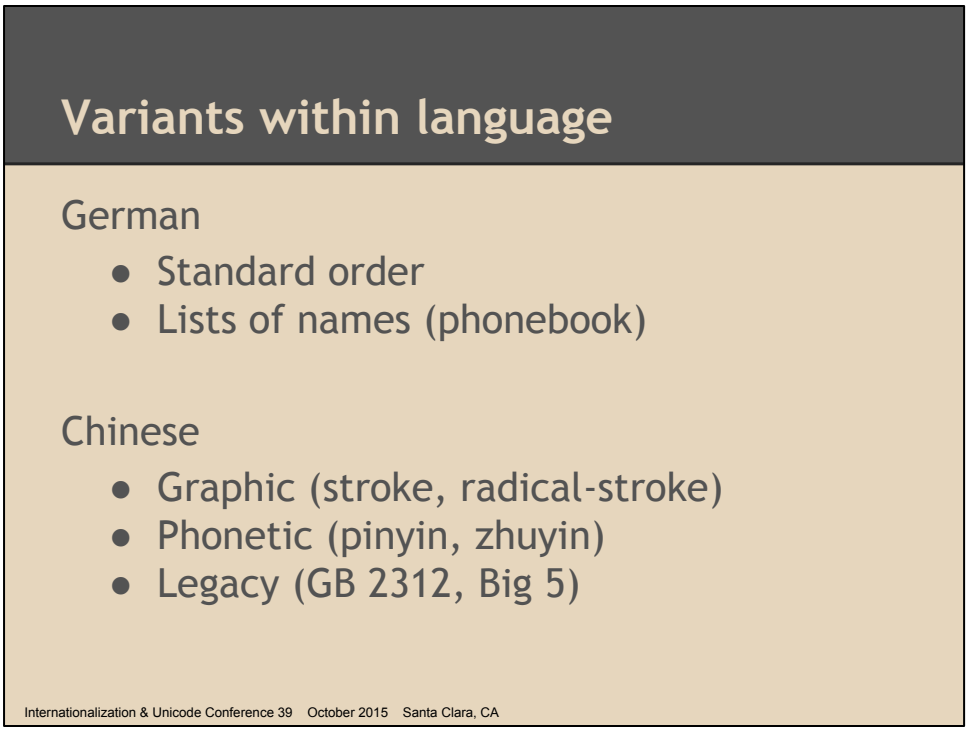

This table shows a list of city names, and how the list is ordered differently for different languages.

The first column is sorted as in English, German, and many other languages, and as in the Unicode default order.

The second column is sorted as in Slovak where the pair “ch” is considered a separate “letter” which sorts between ‘h’ and ‘i’. (See http://en.wikipedia.org/wiki/Slovak_orthography#Alphabet)

The third column is sorted as in Danish where a-ring sorts as a separate letter at the end of the alphabet. (http://en.wikipedia.org/wiki/Danish_and_Norwegian_alphabet)

If a long list (imagine a phone book, or a list of hundreds of contacts on a phone) is not sorted according to a user’s expectations, then a user might not be able to find what they are looking for.

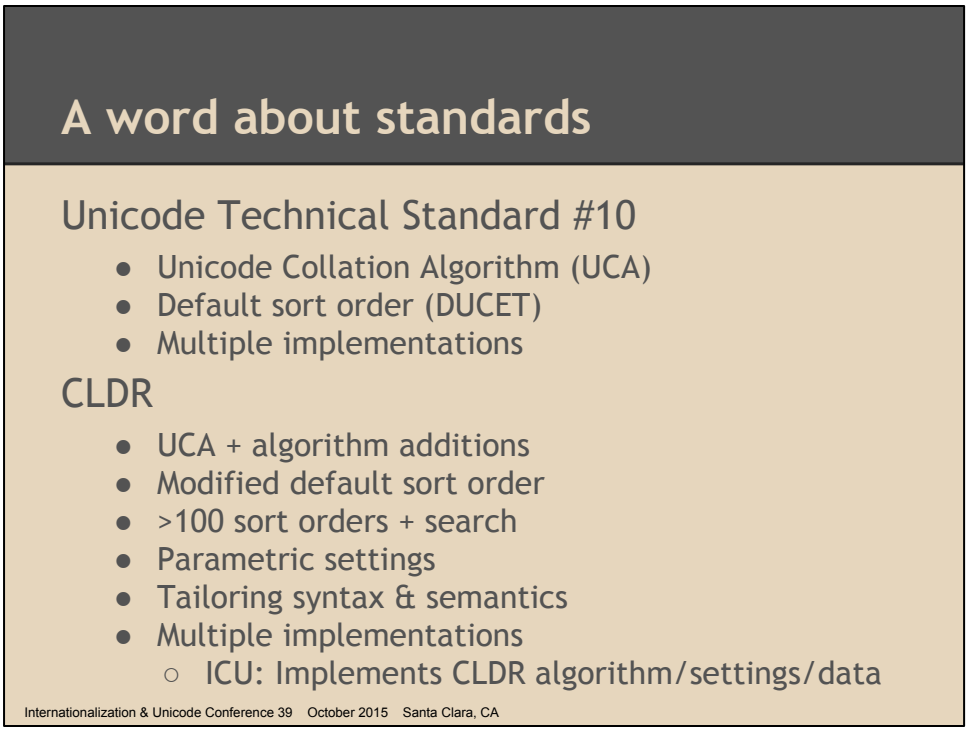



Sometimes there is more than one sorting convention for a single language.

For example, German dictionaries treat letters with umlauts (äöü) as minor variants of the base letters, but in lists of names, which are historically spelled unpredictably, the umlauts are treated as base letter + ‘e’. (http://en.wikipedia.org/wiki/German_orthography#Sorting)

In Chinese, there are several common ways of ordering Han ideographs by appearance or by pronunciation. Japanese and Korean use yet different ways of ordering those characters.

In some languages, the convention has changed over time, so that there may be a “modern” and a “traditional” sort order.




Unicode Collation Algorithm: http://www.unicode.org/reports/tr10/

This defines the algorithm and data for the default Unicode sort order. It is useful as is for many languages and writing systems. For others, it serves as a base for tailoring. Only those characters and sequences that need to change from the default need to be defined specifically.

The DUCET is synchronized with the default data for the older, less capable ISO 14651 sorting standard.

http://www.unicode.org/reports/tr35/tr35-collation.html

The CLDR collation spec adds useful elements to the UCA, modifies the default sort order somewhat, defines parametric settings, defines a concrete mechanism for tailoring via human-readable rule strings, and provides tailoring data for sort orders for many languages. It also provides data for collations that are optimized for searching (e.g., ctrl-F in a browser) rather than sorting.

The algorithms do not prescribe any particular implementation. There are several different implementations of the UCA, and several of the CLDR collation spec. The ICU library implements the CLDR collation spec, and is widely used.

## Multi-level comparison

Compare character by character
- If there is a primary (base letter) difference, then return with that.

Else look for lower-level differences.

aaB > ÄÅá
aaB > ÄÅ



Users expect the order of *strings* to be determined first by the sequence of “letters”; and only when that is the same, then by minor distinctions.

When comparing two strings, look first for primary (base letter) differences across the full lengths of the two strings being compared. Only if there is no primary difference, that is, both strings contain the same sequence of base characters, then look for lower-level diffs.

## Accents, case, variants

- If same base letters, is there a secondary (accent) difference?
- Otherwise, is there a tertiary (case/variant) difference?

aaá > A𝒶𝕒

aaá̧ > A𝒶á

aaA > aa𝒶 > aaa



In many writing systems, the secondary level considers accents/diacritics and ligatures. The third (tertiary) level distinguishes between lowercase and uppercase and (in Unicode collation) also between other minor variations.

## More levels

Case (when turned on)

- Case alone trumps other tertiary diffs
- Untailorable letter case

Quaternary

- “Ignore punctuation”: “ ” < . < any other
- Japanese: か<カ, き<キ

Identical

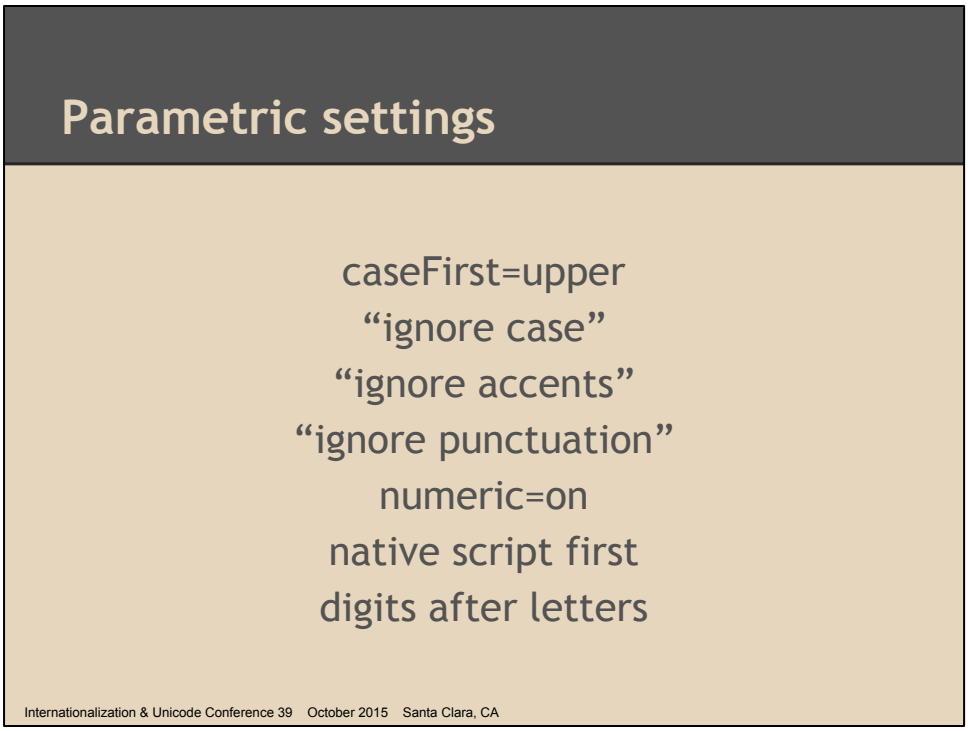
- Tie-breaker if no other diffs
- Untailorable NFD



Further levels can be distinguished as necessary for some use cases or languages.

Ignore Punctuation: http://www.unicode.org/reports/tr10/#Variable_Weighting

http://www.unicode.org/charts/collation/chart_Katakana_Hiragana.html
The default order distinguishes Hiragana from Katakana on tertiary level; the CLDR Japanese tailoring moves this distinction to quaternary level, based on JIS X 4061.



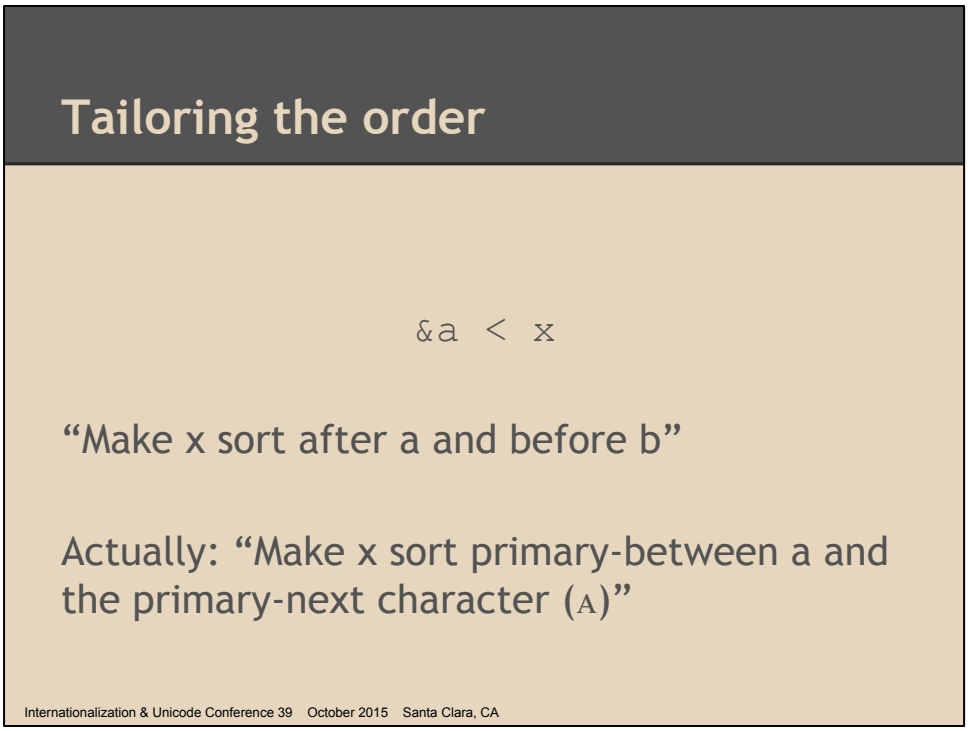

Systematic changes to the sort order that affect many similar characters are best done via parametric settings. For example, there are some 1750 uppercase characters; when they are to be sorted before their lowercase equivalents, it is much simpler and more efficient to use the appropriate setting, rather than reorder them all explicitly. The parametric setting will also work automatically for case pairs that might be added in future versions of the Unicode Standard.

Depending on the implementation, available parametric settings may be specified

- in tailoring rules
- via API on the Collator object
- via a language tag or Unicode Locale ID which includes appropriate -u- extensions

For details about the options defined by CLDR see http://www.unicode.org/reports/tr35/tr35-collation.html#Setting_Options and http://www.unicode.org/reports/tr35/tr35-collation.html#Common_Settings

(Show the effects of (some of) the examples on the slide via the ICU Collation Demo: http://demo.icu-project.org/icu-bin/collation.html)




In the context of Unicode collation, defining a sort order means building on the Unicode default order and changing the order of a few characters relative to the rest. This is called a tailoring.

This is usually done via rules that are relatively human-readable and express how characters sort relative to others. A piece of software (a “builder”) interprets those rules and creates the data tables for string comparisons.

It would not be practical to define the mappings between characters and collation data directly:

- It would be hard to understand and verify.
- The numerical collation weights of the default table change with version and implementation.
- The UCA default table does not provide for gaps between weights.

(ᴀ = U+1D00 Latin Letter Small Capital A)

(Show the collation chart again.)

## Multiple rule chains

```
&a < x &x < y
```

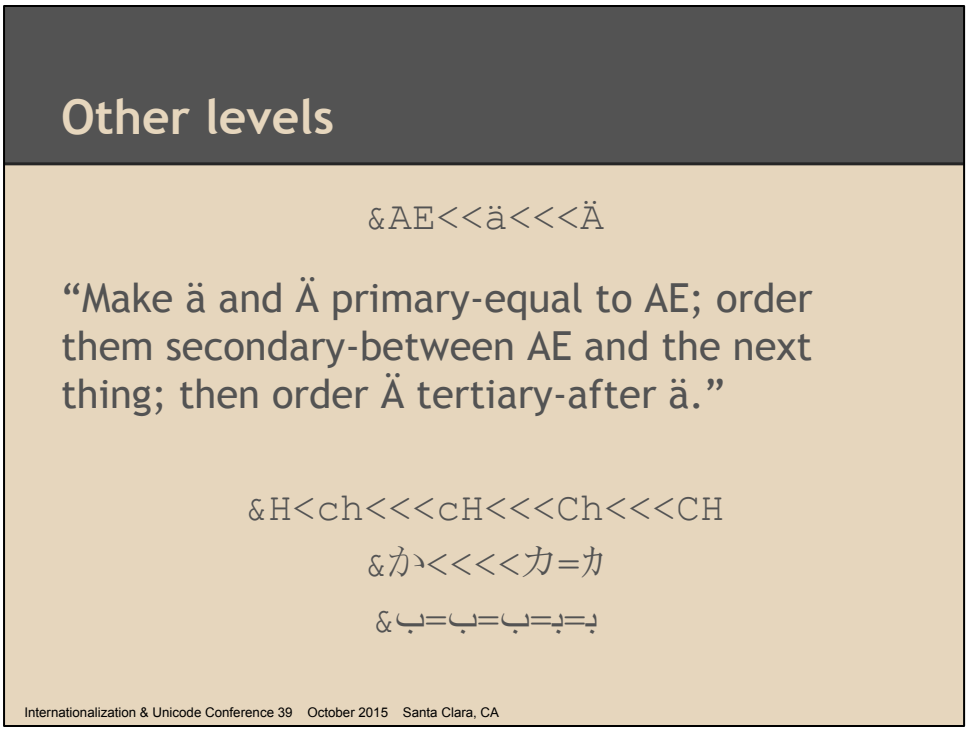
→ a, x, y, A, …, b

```
&a < x &a < y
```

→ a, y, x, A, …, b



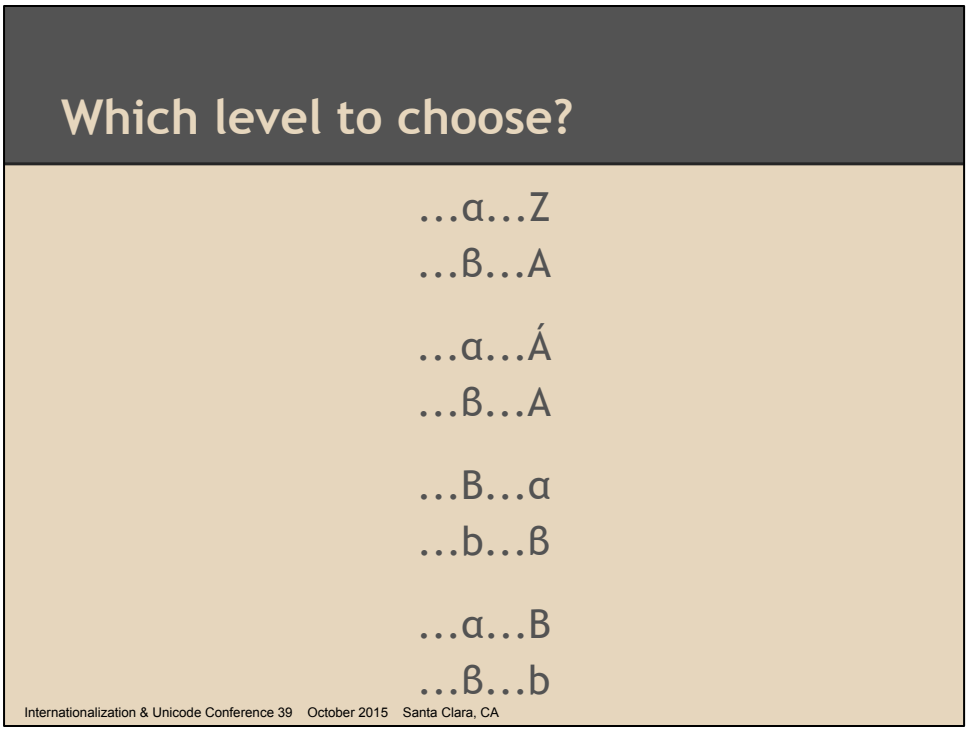
Each rule changes the order established by the default plus the previous rules.




Examples from CLDR tailorings: [http://unicode.org/cldr/trac/browser/trunk/common/collation/](http://unicode.org/cldr/trac/browser/trunk/common/collation/)

- German phonebook order treats “Umlauts” like vowel+’e’ except for a secondary difference
  - This shows an “expansion”: One character sorts similar to a sequence.
- Slovak sorts ch (and its case variants) like a single letter after h
  - This shows a “contraction”: A sequence sorts similar to a single character.
- Japanese treats Hiragana and Katakana as tertiary-equal, with Hiragana quaternary-before Katakana; fullwidth/halfwidth forms sort equal to regular forms (example: Hiragana/Katakana letter Ka)
- CLDR 26 Arabic changes presentation forms so that they sort equal to regular letters (since properly chosen presentation forms should be indistinguishable from properly shaped letters) (example: U+0628 Arabic Letter Beh with its presentation forms)

“Secondary-between AE and the next thing”: Depends on the implementation, but ideally ä should sort secondary-between AE and Aé. ä must “expand” to at least two collation elements, and the last one should have its secondary weight incremented. See [http://www.unicode.org/reports/tr35/tr35-collation.html#Orderings](http://www.unicode.org/reports/tr35/tr35-collation.html#Orderings) and [http://www.unicode.org/reports/tr35/tr35-collation.html#Expansions](http://www.unicode.org/reports/tr35/tr35-collation.html#Expansions) for details.




If you are not sure about the right level for the difference between two characters, then check the expected order of similar words.
The difference is at level x if

- It is trumped by a *later* difference at level x-1
- It trumps a later difference at level x
- It trumps an earlier difference at level x+1

See http://cldr.unicode.org/index/cldr-spec/collation-guidelines#TOC-Determining-Level-Differences

## Advanced rules

```
&[before 2]a<<ā<<<Ā<<á<<<Á...
&[before 3]か゛<<<か゛|ゝ

&[last regular]<*亜唖娃阿哀愛挨姶...

<suppress_contraction>[เ-ไ เ-ไ]
```

Internationalization & Unicode Conference 39 October 2015 Santa Clara, CA

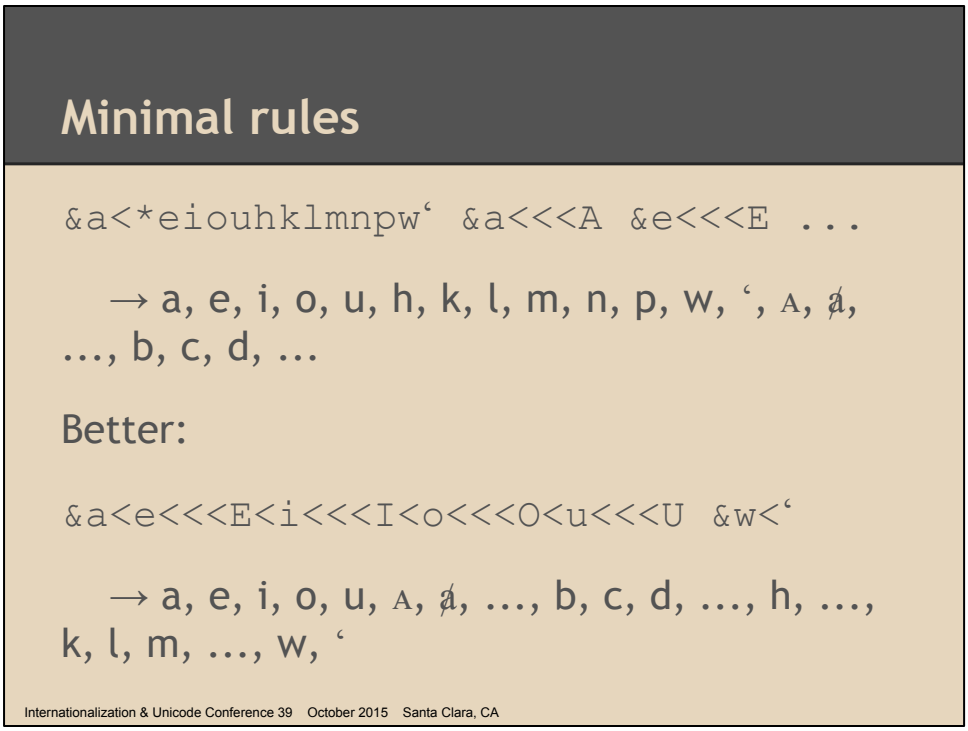
Sometimes we need to order a character between a reference point and the *preceding* item.

- In Chinese Pinyin, the fifth (neutral) tone is written with unmarked Latin letters but sorts after the other tones which are marked with diacritics. (http://en.wikipedia.org/wiki/Pinyin#Tones)
- In Japanese, the iteration mark sorts tertiary-between small and large Kana.
    - This also shows a prefix rule: The iteration mark sorts tertiary-before Ka *if in the string it is preceded by a Ka*.

The “<*” syntax is simply an abbreviation; the beginning of the Japanese Kanji order could also be written as &[last regular]<亜<唖<娃<阿<哀<愛<挨<姶<...
The “&[last regular]” places the characters at the beginning of the Han-script range (after all non-Han scripts).

Sometimes the default order includes contractions that are not desirable for a tailoring. For example, the CLDR search tailorings suppress the contractions for Thai and Lao vowels (and similar) which are written before consonants but where the contractions make them sort as if they follow the consonants. In a search tailoring, where equality matters but order does not, this yields tighter match boundaries.

For more details see http://www.unicode.org/reports/tr35/tr35-collation.html#Rules




“Minimal rules” basically means “Do not duplicate parts of the default order.”

CLDR ticket “minimize Hawaiian collation tailoring” http://unicode.org/cldr/trac/ticket/6257
Changes there: http://unicode.org/cldr/trac/changeset/9241

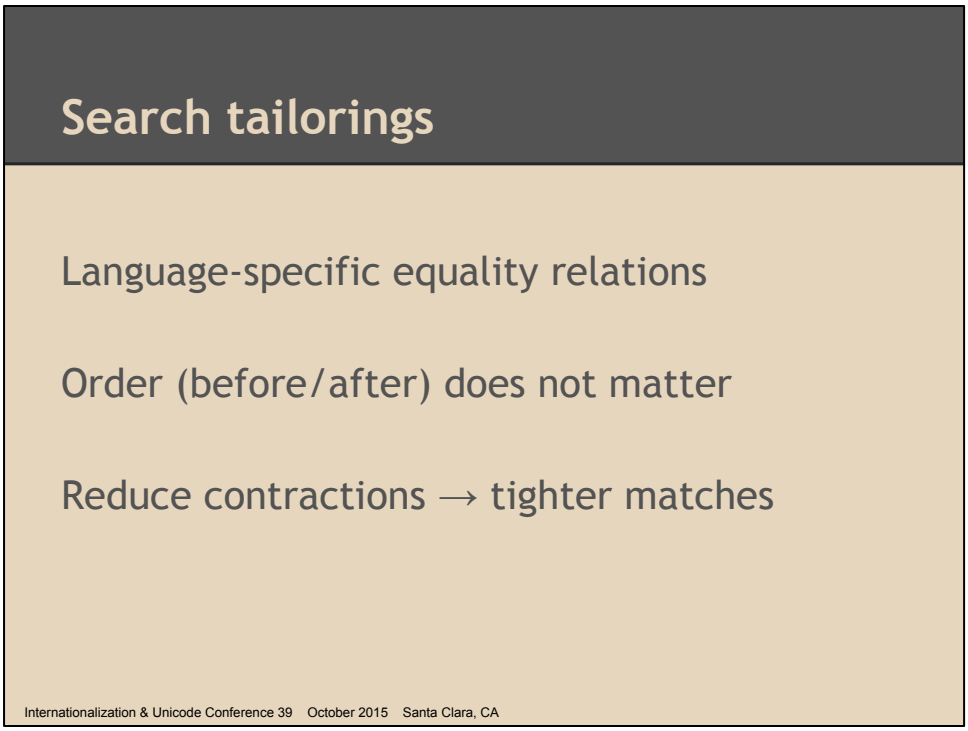
(Show the older rules and ask the audience why they are not good.)

The resulting order shown on the slide only shows primary differences (for brevity).

The old Hawai‘ian tailoring moved the entire Hawai‘ian alphabet between a and b (really between a and ᴀ [U+1D00 Small Capital A]), which made the tailoring unnecessarily long, and in implementations with a highly optimized default table it would result in slower comparisons and longer sort keys.

The improved order only makes the minimal changes necessary to put the letters of the Hawai‘ian alphabet into the desired order (vowels first, ‘okina last), without changing anything else.

See http://cldr.unicode.org/index/cldr-spec/collation-guidelines




Language-sensitive string search can also use the collation mechanisms and data. For searching, only equality relationships matter, together with the levels of differences. The actual order (before/after) does not matter. For example, for searching, the order of Han characters is not relevant and need not be tailored. For most languages and scripts, the sort order serves as a good search comparison as well.

The CLDR search tailorings also remove the default order's loglical-order-exception contractions for Thai, Lao, and Tai Viet which move differences between prevowels to after their consonants. For searching, this is not relevant, and removing those special contractions allows for tighter match boundaries.

These contractions are very visible in the Thai etc. default collation charts: http://www.unicode.org/charts/collation/chart_Thai.html

For examples see the CLDR collation root search rules and language-specific search tailorings.

## Large examples

CLDR 26 Arabic change
(CLDR ticket #4207)

Other CLDR data



CLDR 26 Arabic change see http://unicode.org/cldr/trac/ticket/4207#comment:22
Current data: view-source:http://unicode.org/repos/cldr/trunk/common/collation/ar.xml

All current CLDR collation data: http://unicode.org/repos/cldr/trunk/common/collation/
(Use “view source” for .xml files.)

(Ask audience)

## Hands-on

ICU Collation Demo

site.icu-project.org

demo.icu-project.org/icu-bin/collation.html



Take audience requests

These slides: https://goo.gl/HuRsLU