ICU design: “Mixed locales”

2022q1

Introduction

“Mixed locales”: Locales for a combination of language and region for which we do not have an actual locale in ICU, typically for languages that are not a primary language of the region, algorithmically generated by combining data from both a language locale and a relevant locale for the region.

Mixed locales would be used when a regional-variant locale is requested, but ICU has data only for the written language, not for that specific locale.

Background

At Apple, we’ve been trying to account for a situation where the user has his system locale set so that their preferred language wasn’t one of the normally-recognized languages for their preferred country, such as `en_JP` or `ru_US`. This often happens when the user is working in some country other than their native one and wants their system to reflect that, but still wants to use their computer in their native language.

In cases like this, ICU just takes everything from the default locale data for the user’s language (`en` and `ru` in my examples). Over time, we got many bug reports complaining about this— it seemed that in a lot of cases, users wanted some of the individual resources to be based on the user’s country instead of their language.

A simple locale ID such as the above also doesn’t work when the user speaks a regional variant of a language and is in a different country– for example, if they speak Austrian German but are located in France. To express this, you have to use the `rg` subtag: `de_AT@rg=FR`. If the rg subtag is present, that would be the country to use for any resources that we determine to be country-dependent; otherwise, we’d just use the locale ID’s normal region code. (And in cases where the user doesn’t speak a specific regional variant of a language, the `rg` subtag is superfluous: `en@rg=JP` is equivalent to `en_JP`.

For some time, we’d dealt with this problem by adding new locale data files for the combinations we saw most often. Most of these were English or Spanish in various countries, but I think we might have had a few oddball combinations. At this point, we have TONS of English and Spanish locales because of this (it looks like CLDR has added locales to get around this problem too), and the general feeling was that that was getting out of hand and that this approach would never work for every possible combination of language and country, so we started looking for an algorithmic solution.

So we now have an algorithmic solution, but I’m not all that proud of it (and we’ve gotten our share of bug reports complaining about it too). This is a difficult problem, and I’d like to think there’s a better approach out there somewhere. But whether it’s what I have now or something better, I’d like to see us tackle this problem in ICU and not have it be an Apple-specific addition.

Things that already fall back by country

It’s worth pointing out that ICU already chooses some resources based on the user’s country rather than their language:

- Default currency
- Default calendar
- Hour cycle for time formatting (but not in all places)
- Shane: Unit preferences
- Mark:
    - weekData (minDays, firstDay)
    - weekendStart, weekendEnd
    - measurementSystem
    - See also rgScope.xml
- [are there more?]

Proposed additions to the behavior

In other words, there’s precedent for giving priority to the user’s region setting for choosing certain localized resources. I think we might want to consider adding more to the list. The ones I think we should discuss are:

- Decimal and grouping separators in formatted numbers
- Currency symbols
- Currency symbol position
- Hour cycle in standard time formats
- Field order in all-numeric date formats
- Separator character in time and all-numeric date formats
- Time zone abbreviations

But most of these have subtleties that need to be thought through. I cover all of these topics in more detail below.

Detailed considerations

Number formatting

Default numbering system (digit characters)

Unlike default calendar and currency, default numbering system depends primarily on language. For some languages, such as Arabic, there are countries that override the default numbering system for the language (many Arabic-speaking countries use European digits).

The question is what makes sense when you’re using a nonstandard language/country combination: If you’re an English speaker in Saudi Arabia, you probably want to use European digits, even though Arabic-Indic digits are standard in Saudi Arabia. But if you’re an Arabic speaker in the United States, you might want to use Arabic-Indic digits (the default for Arabic) or European digits. It might depend on whether your home country is Saudi Arabia (which uses Arabic-Indic digits) or the United Arab Emirates (which uses European digits).

This might be worth talking about, but my gut feeling is that we don't want to make a change here – if the user's locale is `ar_US` and they want European digits, they have to set their locale to `ar_US@numbers=latn`.

Decimal and grouping separators

We got a lot of bug reports from users who wanted their decimal and grouping separators to be the standard ones for their country, not the default ones for their language: for example, a French speaker in the US might want a period for the decimal separator and a comma for the thousands separator, where right now they get the default for French: comma for the decimal separator and space for the thousands separator.

We (Apple) specifically got bug reports about this for `fr_US`, `fr_CN`, `es_IE`, `fr_419`, `fr_GB`, and many others.

That said, after we changed this, we got a lot of bug reports (but a smaller number than before) from people who liked the old behavior, including some particularly angry ones from Russian and Ukrainian speakers.

We're never going to get this right for everybody, so some part of solving this involves allowing users to set their own preferences by adding a new locale ID subtag, but we should discuss the default behavior and when it should take the user's region setting into account.

Grouping distance

[TBD] Should grouping distance and minimum grouping digits follow the language, as it does now, or should the user's country play some role?

Sign characters and position

The characters to use for the sign should follow the user's numbering system, rather than either the language or country (although I think we're using the same sign characters in all locales right now).

This is complicated by the RTL languages putting bidi controls into their sign-character resources. We've had all kinds of problems due to assumptions in number and date patterns about the directionality of surrounding text. I think we need to find a better solution to handling bidi in number and date formats than putting bidi control characters into the symbols.

Percent character and position

The percent character should probably follow the user's numbering system, rather than either the language or country.

The position of the percent sign relative to the number should probably follow the language (in Turkish, it's grammatically wrong to put the percent sign after the number).

Whether the percent sign should be separated from the number by a space or not, on the other hand, might depend on the country. This might need discussion.

Scientific notation

[TBD] Any reason to change anything here?

Other formatting symbols

[TBD] Do we want to do anything different with the items in the NumberElements/xxxx/symbols resource that aren't discussed above (infinity, NaN, list separator, etc.)?

Currency formatting

Default currency

This already follows the country; we're not proposing to change this.

Currency symbol

Currency *names* should definitely follow the language; we're not proposing to change this.

Currency *symbols* are more complicated. In some cases, they contain alphabetic characters that may not make sense outside of the appropriate language (and in some situations, like the Saudi rial, can mess up the text around them if it's in a different language). In some cases, they contain alphabetic characters that *are* okay in other languages.

Then there are the cases that involve real symbols. There are many symbols that are used to represent many different currencies: Which currency is meant by $ or ¥ depends explicitly on where you are, and the way in which other currencies that use that symbol are rendered in each of those countries can change too.

I think we're doing a lousy job of this right now whenever the user's locale isn't one we have a resource file for, and that we need to think hard about a better approach.

Currency decimal separator, decimal places, and grouping separator

I believe all of these follow the actual currency right now, rather than either the language or country (that is, in the rare cases where the values for currency differ from the locale's normal number-formatting conventions). I'm not proposing to change this.

Currency symbol location

(That is, whether the currency symbol precedes or follows the number, and whether it's set off from the number with a space.)

We have the ability for some currencies to specify this across all locales, but most currencies follow the locale, and for a given locale, values in all currencies (that don't override this) are formatted the same. In our existing locale data files, the positioning of the currency symbol seems to follow the country (at least for English), but of course this doesn't work for locales we don't have resource data for, and there may be exceptions where a given currency position is ungrammatical for some languages.

This also seems broken to me and in need of a fix (or more discussion).

Time formatting

Hour cycle

This is already being chosen based on country, but I think that only works for `DateTimePatternGenerator` and `DateIntervalFormat`, not for the standard patterns. If I remember correctly, we hacked around this in Apple ICU, but the fix didn't make it to OSICU (I could be wrong about this). We should fix this in Apple ICU if it hasn't been fixed already.

Separator character

I think we've gotten at least one bug complaining that we're using the wrong time separator character for some country. Existing locale data (see `da.txt` and `en_DK.txt`, for example) suggests the time separator character should follow the country.

Time zone name

Time zone names are a challenge, because each locale really only has names and abbreviations for time zones that are in use in that locale. If you're in the US, for example, you really want localized names and abbreviations for the US time zones, regardless of what language you speak.

I don't think it makes sense to have names for *all* time zones in *all* languages, especially since those names vary depending on the country too. But I think it might make sense for users of all languages to use the local time zone *abbreviations* for their country, regardless of language. If the country has multiple languages and they have different abbreviations, maybe you use the abbreviations for that country's default language.

We should talk more about this.

Position of day period and time zone

For time formats that include one or both of these elements, it probably makes sense to follow the language, as we're doing now. The problem is that the pattern not only specifies these things, but also the separator character, which probably should follow the country (see above).

Date formatting

Field order

I think we've gotten a few bugs asking that the field order of a date should follow the country, especially in all-numeric dates.

It'd be very difficult to do this with any date format that includes words, because of the punctuation and other elements that are in the date patterns; what you'd have to do is collect completely separate date patterns for each field order for each locale. I don't know if this is worth it (or if it even makes sense for some languages).

All-numeric date formats are an exception, but I don't know if it makes sense to change the field order in a short format if we're not going to do it for longer formats (although some existing locales do this already). And does it make sense to change the field order for short date formats that *aren't* all-numeric, like some of the CJK formats?

Also, if we change the field order for all-numeric date formats, what do we do with the day-of-the-week or era fields, if they're present? How do they interact with everything else?

Apple ICU *is* changing the field order of short dates right now, but I'm not happy with the way we're doing it. I think we need to discuss this.

Separator character

As with time formats, I think a case can be made that the separator character to use in an all-numeric date format should follow the country, but there's no good way to make that happen with the current design. Is this worth talking about?

Date+time formats

The combination formats for date and time probably always need to follow the language, as they do now.

Relative dates

These need to follow the language, except for situations where we're falling back on the standard formats.

Property	Based On			Comments
	Current	Proposed	Agreed	
Numbers				
Default numbering system	Language	(nc)		Actually depends more on script than language per se, and there are regions that override the defaults for their language (e.g., Arabic-speaking countries that use the “latn” numbering system)
Decimal separator	Language	Region		This is one of the issues that first motivated this whole proposal. This probably needs to be user configurable (via a new locale extension?), but it seems like the default should (generally but maybe not always) follow the region (maybe Russian is an exception?).
Grouping separator	Language	Region		Same comments as “decimal separator” above
Grouping intervals	Language	(nc?)		Meant to encompass primary and secondary grouping intervals and “minimum grouping digits”. Seems like this follows language– are there exceptions?
Sign characters	Language	Numbering system		I don’t think this ever varies, except for the inclusion of bidi controls. Should move the bidi controls somewhere else. This should probably just follow numbering system.
Sign position	Language	(nc?)		Hardly ever varies (although bidi presentation makes it hard to tell what’s intended in RTL languages). Should be based on language?
Percent symbol	Language	Numbering system		Only different for “arab” and “arabext” numbering systems.
Percent symbol position	Language	(nc?)		Only varies for Turkish (I think).
Percent symbol spacing	Language	???		Varies a lot. Not sure whether language or region is more important
List separator	Language	Region		Depends on decimal and grouping separators. Should be based on whatever they’re based on.
Infinity and Nan symbols	Language	(nc)		
Exponential symbols	Language	(nc)		I think it makes sense for the “E” to be based on language. Not sure about subscripting exponent (if it even varies).
“Approximately” symbol	Language	???		
Compact decimal formatting	Language	(nc)		Clearly language-dependent.
Spellout/ordinal	Language	(nc)		Clearly language-dependent.

formatting				
Currency				
Default currency	Region	(nc)		Already based on region
Long currency name	Language	(nc)		Clearly language-dependent
Currency symbol	Language	Region		Currently based on language, with regions overriding. Maybe should be based first on the currency itself, with regions overriding, and then (if needed) languages overriding.
Currency decimal places	Currency	(nc)		
Currency rounding	Currency	(nc)		
Currency decimal and grouping separators	Language	Region		Currently based on language, with regions overriding. Maybe should be based first on the currency itself, with regions overriding, and then (if needed) languages overriding. I think we have a couple of overridden currency grouping separators (I don't remember where– Switzerland?) and one overridden currency decimal separator (the Cape Verdean escudo). For the escudo, it seems more appropriate to just call "$" the currency symbol and say it goes in the middle of the number (so there are three currency-symbol positions: before, after, and decimal).
Currency symbol position	Language	???		Right now language first, then region, wth some currencies overriding. Does it make sense to invert this order?
Currency symbol spacing	Language	???		Currently the language specifies this with the region overriding. If the language and region don't put a space between the currency symbol and the number, there's an algorithm whereby a space is added *anyway* for certain localized currency symbols. Does this still make sense?
Units				
Unit names and abbreviations	Language	(nc)		Included for completeness (in case I got something wrong). No changes proposed.
Unit position and spacing	Language	(nc)		
Default units for usage	Region	(nc)		
Dates				
Default calendar	Region	(nc)		There's a resource for this in the "locales" bundles too, but I think the one in supplementalData/calendarPreferenceData is the one we're actually using.
Field order	Language	Region		For short (i.e., all-numeric) dates, should probably

				be user configurable via a new locale subtag, with the default being based on region. For longer dates, we either need to continue to base this on language or add resources to allow support of multiple field orders per language. Is it okay for the field order to vary depending on the length of the date pattern?
Non-numeric formatting patterns	Language	(nc)		Thinking here about punctuation and static words. This has to be based on language (but if we allow for different field orders, we need more patterns).
Separator character in numeric dates	Language	???		Maybe this should also be user configurable via a new locale subtag (or at least by API). Other than that, I can see arguments for following either language or region, but not sure it should be tied as intimately to field order as it is right now.
Position of era with numeric date	Language	???		This is specified by the same pattern that specifies field order and separator character. Does it make sense to break out somehow? Should era position (or presence) be based on calendar first and only then on either language or region?
Position of day-of-week with numeric date	Language	???		This is specified by the same pattern that specified field order and separator character. Does it make sense to break out somehow?
Names and abbreviations for months, days, quarters, and eras	Language	(nc)		
Combination date+time formats	Language	(nc)		
Relative date formats	Language	(nc)		Unless we're falling back on an absolute date format
Date interval formats	Language	???		Some of the items above, such as field order, should also affect date intervals, but it probably makes sense for the stuff specific to date intervals to continue to be based on language. (It *would* be nice to have the dash character and the spaces around it be independently configurable without having to change all the interval formats…)
Times				
Hour cycle	Region	(nc)		Standard "canned" time formats still based on language, but I think a change is in the works.
Day period names/abbreviations	Language	(nc)		
Separator character	Language	???		This is : almost everywhere. Do the locales that override this do so based on language or region? Should this be independently configurable?
Day period position	Language	???		Is this okay the way it is?

Fractional second separator?	Language	(nc?)		Does this ever vary, or is it always "."? Is this important enough to worry about?
Time zone position	Language	(nc?)		This is in the pattern with everything else. Are there any reasons to break it out?
Time zone names	Language	(nc)		
Time zone abbreviations	Language	???		This is complicated. The abbreviations are based on the long names, which are language-dependent, but the set of time zones you care about depends on your location, and many time zones have names and abbreviations that are clearly based on the user's location (is it "Japan standard time" or "Korea standard time"?). We also have abbreviations which are used to mean more than one time zone, with the correct time zone for the abbreviation being based on location. It might be fine for a user in some region to use that region's time zone abbreviations even if they're based on a language other than the user's actual language. This needs some serious disentangling.