Put ICU to Work!

Steven R. Loomis, IBM

Shane Carr, Google
Can’t I just “use Unicode” and be done?
Can’t I just “use Unicode” and be done?

Put ICU to Work!
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts)
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
- More than 143,000 characters
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
- More than 143,000 characters
- Significant update about once a year
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
- More than 143,000 characters
- Significant update about once a year
- 80+ character properties, many multi-valued
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
- More than 143,000 characters
- Significant update about once a year
- 80+ character properties, many multi-valued
- Additional standards and geopolitical requirements
  - CLDR, ISO, TZ, …
Can’t I just “use Unicode” and be done?

- 3,639 pages (core+charts) + Annexes
- More than 143,000 characters
- Significant update about once a year
- 80+ character properties, many multi-valued
- Additional standards and geopolitical requirements
  - CLDR, ISO, TZ, …
- Good vs. fast
Unicode covers the world

- “Unicode provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language.” (unicode.org)
Unicode covers the world 🌍

- “Unicode provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language.”
  (unicode.org)

ICU brings you home 🏡

- Requirements vary widely across languages & countries
- Sorting
- Text searching
- Bidirectional text processing
- Date/time/number/currency formatting
- Codepage conversion
- …many more
I See Unicode

- 1999: *IBM Classes for Unicode* open-sourced as the *International Components for Unicode*
- 2016: ICU joins Unicode as ICU-TC
- 2018: Development now on GitHub and Jira
ICU’s Laundry List

- Breaks: word, line, ...
- Formatting
  - Date & time
  - Durations
  - Messages
  - Numbers & currencies
  - Plurals
- Transforms
  - Normalization
  - Casing
- Transliterations

- Unicode text handling
- Charset conversions (200+)
- Charset detection
- Collation & Searching
- Locales from CLDR (640+)
- Resource Bundles
- Calendar & Time zones
- Unicode Regular Expressions ...
Benefits of ICU

- Mature, widely used, up-to-date set of C/C++ and Java libraries
  - Basis for Java 1.1 internationalization, but goes far beyond Java 1.1
  - Team continues to work on improving and monitoring performance.
- Very portable – identical results on all platforms/programming languages
  - C/C++ (ICU4C): many platforms/compilers
  - Java (ICU4J): Oracle Java SE, IBM JRE, OpenJDK, Android
  - Wrappers: D/C#/PHP/Python/…
- Customizable & Modular
  - Open source (since 1999) – but non-restrictive
  - Contributions from many parties (IBM, Google, Apple, Microsoft, …)
- Sponsored by Unicode
Where do I get ICU?

Main site: http://icu-project.org/

- Downloads, API references, Mailing list, Bug tracking
- Userguide: http://userguide.icu-project.org
  - Moving soon to GitHub Pages
  - User’s guide with examples
Prepackaged ICU

Package Managers (C)

- brew install icu4c
- apt-get install libicu-dev
- dnf install libicu-devel

Maven and friends: (J)

- group: com.ibm.icu
- artifactId: icu4j
ICU Userguide

Formatting and Parsing

Overview

Formatters translate textual representations into computer representations. Textual representations only display the number as text that the user would see. For example, a textual representation of 1234 may contain the text "one thousand two hundred thirty-four" plus a locale-specific indicator of the internal value.

Usage

RuleBasedNumberFormat can be used like other NumberFormats. For example, in Java:

```java
double num = 3718.38;
NumberFormat formatter =
    new RuleBasedNumberFormat(RuleBasedNumberFormat.SPELLOUT);
String result = formatter.format(num);
System.out.println(result);

// output (in en_US locale):
// two thousand seven hundred and eighteen point two eight
```
API Docs

**RuleBasedNumberFormat**

```java
public RuleBasedNumberFormat(String description,
String[][] localizations)
```

Creates a RuleBasedNumberFormat that behaves according to the description passed in.

The formatter uses the default locale.

**Parameters:**

rules A description of the formatter's desired behavior. See the class documentation for a complete explanation of syntax.
error The parse error if an error was encountered.
status The status indicating whether the constructor succeeded.

**Stable:**

ICU 3.2
API Change Report

Removed from ICU4J 59.1
(no API removed)

Deprecated or Obsoleted in ICU4J 60.1

Package com.ibm.icu.util

Calendar
- (deprecated) protected int computeMillisInDay()
- (deprecated) protected int computeZoneOffset(long, int)

Package com.ibm.icu.lang

UProperty
- (stable) public static final int EMOJI
- (stable) public static final int EMOJI_MODIFIER
- (stable) public static final int EMOJI_MODIFIER_BASE
- (stable) public static final int EMOJI_PRESENTATION

Package com.ibm.icu.text

- (stable) public class BidiTransform
- (stable) public static enum BidiTransform.Mirroring
- (stable) public static enum BidiTransform.Order
- (stable) public static final BidiTransform.Mirroring OFF
- (stable) public static final BidiTransform.Mirroring ON

Added in ICU 60

- Removed from ICU 59
- Deprecated or Obsoleted in ICU 60
- Changed in ICU 60
- Promoted to stable in ICU 60
- Added in ICU 60
- Other existing drafts in ICU 60
- Signature Simplifications

Put ICU to Work!

43rd Internationalization and Unicode Conference • Oct 16th, 2019 — @srl295
Mailing Lists

http://site.icu-project.org/contacts

- icu-support – Technical support and discussion
- icu-design – API proposal discussion
- icu-announce – Low-volume announcements list
Issues (Jira)
Contributing

1. Open an issue in Jira
2. Fork the ICU repo
3. Write and test your code
4. Commit your change to your fork
5. Open a new Pull Request
6. Sign the CLA when prompted
Contributing

1. Open an issue in Jira
2. Fork the ICU repo
3. Write and test your code
4. Commit your change to your fork
5. Open a new Pull Request
6. Sign the CLA when prompted
7. Bask in your newfound fame and fortune!
And now, code
Task at Hand

- Display a list of world regions, with their population figures
Task at Hand

- Display a list of world regions, with their population figures

Example

- 150,000: Ceuta and Melilla
- 38,087,800: Algeria
- 15,439,400: Ecuador
ICU4C First Look

```c
#include <unicode/…>

void func() {
    UErrorCode status = U_ZERO_ERROR
    u_init(&status)
    if ( U_SUCCESS(status) ) { /* … */ }
}
```
ICU4C First Look

```c
#include <unicode/…>

void func() {
    UErrorCode status = U_ZERO_ERROR
    u_init(&status)
    if ( U_SUCCESS(status) ) { /* ... */ }
}
```

- All ICU headers are in the `unicode/` subdirectory
ICU4C First Look

```c
#include <unicode/…>

void func() {
    UErrorCode status = U_ZERO_ERROR;
    u_init(&status);
    if (U_SUCCESS(status)) { /* ... */ }
}
```

**UErrorCode status = U_ZERO_ERROR**

- Error code is a fill-in, but must be initialized
- If in C++, `icu::ErrorCode` is available (example on next slide)
ICU4C First Look

```c
#include <unicode/…>

void func() {
    UErrorCode status = U_ZERO_ERROR
    u_init(&status)
    if ( U_SUCCESS(status) ) { /* ... */ }
}
```

```

u_init(&status)

- Returns successful status if ICU data loaded OK
```
# ICU4C First Look

```c
#include <unicode/…>

void func() {
    UErrorCode status = U_ZERO_ERROR;
    u_init(&status);
    if (U_SUCCESS(status)) {
        /* … */
    }
}
```

if (U_SUCCESS(status))

- TRUE if there was no error
Error codes in C++

No need to initialize! Less prone to error:

```cpp
#include <unicode/…>

int main() {
    icu::ErrorCode status
    u_init(status)
    if (status.isFailure()) {
        return 1
    }
    return 0
}
```
ASSERT_OK()

C++ version:

```c++
#define ASSERT_OK(status) \
if(status.isFailure()) { \
    puts(status.errorName()); \
    return 1; \
} 
```

Plain C version:

```c
#define ASSERT_OK(status) \
if(U_FAILURE(status)) { \
    puts(u_errorName(status)); \
    return 1; \
} 
```
**ASSERT_OK()**

C++ version:

```cpp
#define ASSERT_OK(status) \
  if(status.isFailure()) { \ 
    puts(status.errorName()); \ 
    return 1; \ 
  }
```

Plain C version:

```c
#define ASSERT_OK(status) \ 
  if(U_FAILURE(status)) { \ 
    puts(u_errorName(status)); \ 
    return 1; \ 
  }
```

- always check for failure
**ASSERT_OK()**

C++ version:

```cpp
#define ASSERT_OK(status) \  
if(status.isFailure()) { \  
    puts(status.errorName())\  
    return 1\  
}  
```

Plain C version:

```c
#define ASSERT_OK(status) \  
if(U_FAILURE(status)) { \  
    puts(u_errorName(status))\  
    return 1\  
}  
```

- always check for failure
- (We will use this macro to keep test code more compact)
#include <unicode/ustdio.h>

int main(int argc, const char *argv[]) {
    u_printf_u(u"This is ICU %s! 😻\n", U_ICU_VERSION)
    return 0;
}
#include <unicode/ustdio.h>

int main(int argc, const char *argv[]) {
    u_printf_u(u"This is ICU %s! 😷 \n", U_ICU_VERSION);
    return 0;
}

This is ICU 64.2! 😷
This is ICU 64.2! 🐱

- but, let’s actually build this
Building s09_test.c

$ brew install icu4c pkg-config
Building s09_test.c

$ brew install icu4c pkg-config

$ git clone https://github.com/unicode-org/icu-demos.git -b iuc43
Building `s09_test.c`

```
$ brew install icu4c pkg-config
$ git clone https://github.com/unicode-org/icu-demos.git -b iuc43

$ cd iucsamples/c/s09_test
$ make check
This is ICU 64.2! 🐱
everything is OK 🌟
```
Building `s09_test.c`

```bash
$ brew install icu4c pkg-config

$ git clone https://github.com/unicode-org/icu-demos.git -b iuc43

$ cd iucsamples/c/s09_test
$ make check
This is ICU 64.2! 😻
everything is OK 🎉
```

under the hood:

- paths detected via pkg-config

```bash
cc -I/usr/local/Cellar/icu4c/64.2/include -o s09_test s09_test.c -L/usr/local/
```
#include <unicode/errorcode.h>
#include <unicode/locale.h>
#include <unicode/ustdio.h>
#include <unicode/ustream.h>
#include <iostream>

int main() {
    icu::ErrorCode status
    icu::Locale locale("und_001")
    icu::UnicodeString world
    locale.getDisplayCountry(world)
    ASSERT_OK(status)

    std::cout << "Hello, " << world << "!" << std::endl
    return 0
}
Hello, World!
Hello, Mundo!
icuhelloworld.cpp

$ LC_ALL=mt ./s13a_hello

Hello, Dinja!

$ LC_ALL=zh ./s13a_hello

Hello, 世界!
What if we want to change “Hello”?
String Concatenation 😲

- Order is different for different languages, can't just concatenate strings.
String Concatenation 😳

- Order is different for different languages, can't just concatenate strings.

*My Aunt’s pen* is on the table.
String Concatenation 😻

- Order is different for different languages, can't just concatenate strings.

My Aunt's pen is on the table.

whom + "'s " + what + " is on the " + where
String Concatenation 😵

- Order is different for different languages, can't just concatenate strings.

My **Aunt's pen** is on the table.

```plaintext
whom + "'s " + what + " is on the " + where
```

La **pluma de mi tía** está sobre la mesa.
Pattern Syntax
Pattern Syntax

en: {whom}’s {what} is on the {where}.
Pattern Syntax

en: {whom}’s {what} is on the {where}.

es: {what} de {whom} está sobre la {where}.
Pattern Syntax

en: {whom}’s {what} is on the {where}.

es: {what} de {whom} está sobre la {where}.

Or, avoid sentences entirely

“Location: table, Object: pen, Owner: Aunt”
```cpp
const int kArgCount = 1;
Formattable arguments[kArgCount] = { world }
UnicodeString argnames[kArgCount] = {"world"}
FieldPosition fpos = 0;
```
```cpp
const int kArgCount = 1;
Formattable arguments[kArgCount] = { world }
UnicodeString argnames[kArgCount] = {"world"}
FieldPosition fpos = 0

MessageFormat msg_en("Hello, {world}",
   Locale("en"), status)
UnicodeString result_en
msg_en.format(argnames, arguments, kArgCount, result_en, status)
ASSERT_OK(status)
std::cout << "en: " << result_en << std::endl
```

en: Hello, World
const int kArgCount = 1;
Formattable arguments[kArgCount] = { world }
UnicodeString argnames[kArgCount] = {"world"}
FieldPosition fpos = 0

MessageFormat msg_es("¡Hola, {world}!",
   Locale("es"), status)
UnicodeString result_es
msg_es.format(argnames, arguments, kArgCount, result_es, status)
ASSERT_OK(status)
std::cout << "es: " << result_es << std::endl

es: ¡Hola, Mundo!
Java
Java(ICU4J)
ICU4J: Hello, Maven

<dependency>
  <groupId>com.ibm.icu</groupId>
  <artifactId>icu4j</artifactId>
  <version>64.2</version>
</dependency>
Hello.java

Locale locale = Locale.getDefault();
String world = LocaleDisplayNames
  .getInstance(ULocale.forLocale(locale))
  .regionDisplayName("001")
System.out.println("Hello, " + world + "☃")

Hello, World☃
Locale locale = Locale.forLanguageTag("es");
String world = LocaleDisplayNames.getInstance(ULocale.forLocale(locale)).regionDisplayName("001");
System.out.println("Hello, "+ world + "☃");

Hello, Mundo ☃️
Hello.java (español)

Locale locale = Locale.forLanguageTag("es");
String world = LocaleDisplayNames.getInstance(ULocale.forLocale(locale)).regionDisplayName("001");
System.out.println("Hello, " + world + "☃");

Hello, Mundo ☃

- use java.util.Locale
Hello.java (español)

```java
Locale locale = Locale.forLanguageTag("es");
String world = LocaleDisplayNames
    .getInstance(ULocale.forLocale(locale))
    .regionDisplayName("001");
System.out.println("Hello, " + world + "☃");
```

Hello, Mundo ☃

- use java.util.Locale
- ...except for some ICU4J APIs that still use ICU’s ULocale
BadMessage.properties

population=The territory of {territory} has {population} persons.
BadMessage.java

```java
final Locale locale = Locale.getDefault();
ResourceBundle rb = ResourceBundle.getBundle(BadMessage.class.getName());
String popmsg = rb.getString("population");
System.out.println("Message: " + popmsg);

for(final PopulationData.TerritoryEntry entry :
    PopulationData.getTerritoryEntries(locale)) {
    MessageFormat m = new MessageFormat(popmsg, locale);
    Map msgArgs = new HashMap<String, Object>();
    msgArgs.put("territory", entry.territoryName());
    msgArgs.put("population", entry.population());
    System.out.println(m.format(msgArgs));
}
```
BadMessage.java

```java
final Locale locale = Locale.getDefault();
ResourceBundle rb = ResourceBundle.getBundle(BadMessage.class.getName());
String popmsg = rb.getString("population");
System.out.println("Message: " + popmsg);

for(final PopulationData.TerritoryEntry entry : PopulationData.getTerritoryEntries(locale)) {
    MessageFormat m = new MessageFormat(popmsg, locale);
    Map msgArgs = new HashMap<String, Object>();
    msgArgs.put("territory", entry_territoryName());
    msgArgs.put("population", entry.population());
    System.out.println(m.format(msgArgs));
}
```

Message: The territory of {territory} has {population} persons.
The territory of Afghanistan has 34,124,800 persons.
The territory of Albania has 3,047,990 persons.
The territory of Algeria has 40,969,400 persons.

- ok so far
BadMessage.java

```java
final Locale locale = Locale.getDefault();
ResourceBundle rb = ResourceBundle.getBundle(BadMessage.class.getName());
String popmsg = rb.getString("population");
System.out.println("Message: " + popmsg);

for(final PopulationData.TerritoryEntry entry : PopulationData.getTerritoryEntries(locale)) {
    MessageFormat m = new MessageFormat(popmsg, locale);
    Map msgArgs = new HashMap<String,Object>();
    msgArgs.put("territory", entry.territoryName());
    msgArgs.put("population", entry.population());
    System.out.println(m.format(msgArgs));
}
```

The territory of Bouvet Island has 1 persons.
The territory of Unknown Region has 0 persons.

- Not so OK!
BadMessage.java

```java
final Locale locale = Locale.getDefault();
ResourceBundle rb = ResourceBundle.getBundle(BadMessage.class.getName());
String popmsg = rb.getString("population");
System.out.println("Message: " + popmsg);

for(final PopulationData.TerritoryEntry entry :
   PopulationData.getTerritoryEntries(locale)) {
   MessageFormat m = new MessageFormat(popmsg, locale);
   Map msgArgs = new HashMap<String, Object>();
   msgArgs.put("territory", entry.territoryName());
   msgArgs.put("population", entry.population());
   System.out.println(m.format(msgArgs));
}
```

The territory of Bouvet Island has 1 persons.
The territory of Unknown Region has 0 persons.

- Not so OK!
CLDR Plurals
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn,
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn, 1 ci,
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn, 1 ci, 2 gi,
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn, 1 ci, 2 gi, 3 ci,
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn, 1 ci, 2 gi, 3 ci, 4 ci
CLDR Plurals

- English: 0 dogs, 1 dog, 2 dogs, 3 dogs, 4 dogs
- Welsh: 0 cŵn, 1 ci, 2 gi, 3 ci, 4 ci
## CLDR Plurals

<table>
<thead>
<tr>
<th>Language</th>
<th>Plural Markers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Azerbaijani</td>
<td>BX</td>
</tr>
<tr>
<td>Bambara</td>
<td>BX</td>
</tr>
<tr>
<td>Burmese</td>
<td>BX</td>
</tr>
<tr>
<td>Chinese</td>
<td>BX</td>
</tr>
<tr>
<td>Dzongkha</td>
<td>BX</td>
</tr>
<tr>
<td>Georgian</td>
<td>BX</td>
</tr>
<tr>
<td>Hungarian</td>
<td>BX</td>
</tr>
<tr>
<td>Igbo</td>
<td>BX</td>
</tr>
<tr>
<td>Indonesian</td>
<td>BX</td>
</tr>
<tr>
<td>Japanese</td>
<td>BX</td>
</tr>
<tr>
<td>Javanese</td>
<td>BX</td>
</tr>
<tr>
<td>Kabuverdianu</td>
<td>BX</td>
</tr>
<tr>
<td>Kannada</td>
<td>BX</td>
</tr>
<tr>
<td>Khmer</td>
<td>BX</td>
</tr>
<tr>
<td>Korean</td>
<td>BX</td>
</tr>
<tr>
<td>Koyraboro Henri</td>
<td>BX</td>
</tr>
<tr>
<td>Lao</td>
<td>BX</td>
</tr>
<tr>
<td>Tongan</td>
<td>BX</td>
</tr>
<tr>
<td>Turkish</td>
<td>BX</td>
</tr>
<tr>
<td>Vietnamese</td>
<td>BX</td>
</tr>
<tr>
<td>Wolof</td>
<td>BX</td>
</tr>
<tr>
<td>Yoruba</td>
<td>BX</td>
</tr>
</tbody>
</table>

---

Put ICU to Work!

---

43rd Internationalization and Unicode Conference • Oct 16th, 2019 — @sr1295
GoodMessage.properties

population={population, plural,
  one{The territory of {territory} has # person}
  other{The territory of {territory} has # persons}}
GoodMessage.properties

```properties
population={population, plural,
    one{The territory of {territory} has # person}
    other{The territory of {territory} has # persons}}
```

- no code change
GoodMessage.properties

```properties
population={population, plural,
    one{The territory of {territory} has # person}
    other{The territory of {territory} has # persons}}
```

- no code change

The territory of United States has 323,996,000 persons
The territory of Unknown Region has 0 persons
The territory of Uruguay has 3,351,020 persons
The territory of Botswana has 2,209,210 persons
The territory of Bouvet Island has 1 person
The territory of Brazil has 205,824,000 persons
Units and Currencies

The room measures
{0, plural, one{1 meter} other{# meters}}
wide.

The room measures 0 meters wide.
The room measures 1 meter wide.
The room measures 0 meters wide.

But with ICU message strings, ICU can handle measurement units without having to enumerate all the plural forms yourself!

Use the "number" type instead of "plural" type and pass a number skeleton:

The room measures
{0, number, ::measure-unit/length-meter unit-width-full-name}
wide.

Also works for currencies.

Sample code: s88_units.cpp
Compact Notation

Luis Fonsi - Despacito ft. Daddy Yankee

5.4B views ⬆️ 29M ⬇️ 3.4M

Programmatically:

```
std::cout
  << icu::number::NumberFormatter::with()
     .notation(icu::number::Notation::compactShort())
     .locale("en-us")
     .formatDouble(quantity, status)
     .toString(status)
  << std::endl
```

Via Message String:

{0, number, ::compact-short}

Sample code: s99_compact.cpp
Break Iteration

- Unicode standards + tailoring
- UAX#14 line breaking
- UAX#29 sentence, grapheme cluster, word
Break Iteration Sample

Sample: s23_brki.cpp
Collators (Text Sorting)

- binary comparison inadequate
- order varies by language (Danish ‘aa…’ follows ‘z…’)
- need multiple-level collation

Uses:
- comparing
- sorting
- searching

Options:
- case sensitive?
- ignore punctuation?
- UPPERCASE first?
- which variant collator?
- which locale?
- custom tailorings?
- time vs. memory tradeoff?
CollateMessage.java

Collator col = Collator.getInstance(locale);
for(final PopulationData.TerritoryEntry entry :
    PopulationData.getTerritoryEntries(locale,
        new TreeSet<>((o1, o2)
            -> col.compare(o1.territoryName(), o2.territoryName())))) {
    ...
}
CollateMessage.java

Collator col = Collator.getInstance(locale);
for(final PopulationData.TerritoryEntry entry :
    PopulationData.getTerritoryEntries(locale, 
        new TreeSet<>((o1, o2) 
            -> col.compare(o1.territoryName(), o2.territoryName())))) {
    ...

- No Lambda function needed if Set<String>
Multilingual

Russian

The territory of Мхмзq.wur {-om has 26 200 persons in it.
The territory of Мхпмзия has 3 011 410 persons in it.

Japanese

アイスランドには、315,281人います。
アイルランドには、4,775,980人います。

Spanish

En la región de “Afganistán” hay 31.108.100 personas.
En la región de “Albania” hay 3.011.410 personas.
En la región de “Angola” hay 18.565.300 personas.
API Stability

- Internal: Used by ICU implementation or Technology Preview.
API Stability

- **Internal**: Used by ICU implementation or Technology Preview.
- **Draft**: New API, reviewed and approved by ICU project team. The API might be still changed.
API Stability

- **Internal**: Used by ICU implementation or Technology Preview.
- **Draft**: New API, reviewed and approved by ICU project team. The API might be still changed.
- **Stable**: For public use, the API signature won’t be changed in future releases.
API Stability

- **Internal**: Used by ICU implementation or Technology Preview.
- **Draft**: New API, reviewed and approved by ICU project team. The API might be still changed.
- **Stable**: For public use, the API signature won’t be changed in future releases.
- **Deprecated**: Previously Stable, but no longer recommended. The API might be removed after a few releases.
API Stability

- **Internal**: Used by ICU implementation or Technology Preview.
- **Draft**: New API, reviewed and approved by ICU project team. The API might be still changed.
- **Stable**: For public use, the API signature won’t be changed in future releases.
- **Deprecation**: Previously Stable, but no longer recommended. The API might be removed after a few releases.

More details:

- [userguide.icu-project.org/design](http://userguide.icu-project.org/design)
API Stability in docs

```cpp
icu::BreakIterator::BreakIterator ( const BreakIterator & other )

Internal:
Do not use.

This API is for internal use only.

Definition at line 632 of file brkiter.h.
src source smart pointer

Returns
*this

Draft:
This API may be changed in the future versions and was introduced in ICU 56
Definition at line 255 of file localpointer.h.

Parameters
status The error code, set if a problem occurs while creating the set.

Stable:
ICU 51
```
Binary Stability

Source code compatible

- Consumer program should be compiled successfully without changes.
- Rare exceptions, documented in readme.

Serialization compatible (ICU4J)

- Newer ICU version should be able to deserialize object data serialized by older ICU version.
- (see docs for limited exceptions)
ICU Data File

- aka, icudt64l.dat
- 20MB+ of data to support ICU's features
ICU Data File

- aka, icudt64l.dat
- 20MB+ of data to support ICU's features

![Diagram showing major languages and regions, regional variants, and locales]
Packaging
Packaging “It’s too big”
Packaging “It’s too big”

ICU 64 Data Build Tool

- Slice data by feature and locale
- Read the Docs, and attend Shane's session tomorrow

Other Customization

- Repackage ICU data http://userguide.icu-project.org/icudata
- Repackage ICU4C Code http://userguide.icu-project.org/packaging

Example: #define UCONFIG_NO_LEGACY_CONVERSION

  (Note: removes code but not data)
Data Changes

ICU 4.4
isLetter(‘ा’) : false (undefined)
Polish Date Format: "08-04-2014"

ICU 53
isLetter(‘ा’) : true
Polish Date Format: "8 kwi 2014"

Unicode 4.1 5.0 5.1 5.2 6.0 6.2 6.3

CLDR 1.6 1.7 1.8 1.9 2.0 21 22 23 24 25 26
Data Stability

Unicode stability
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
- policy http://www.unicode.org/policies/stability_policy.html
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
- policy http://www.unicode.org/policies/stability_policy.html
- Unicode is still growing.
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
- policy http://www.unicode.org/policies/stability_policy.html
- Unicode is still growing.

Locale data
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
- policy http://www.unicode.org/policies/stability_policy.html
- Unicode is still growing.

Locale data

- cultural data can be updated based on community voting
Data Stability

Unicode stability

- character type, upper/lower case, normalization, text direction, sorting order...
- policy http://www.unicode.org/policies/stability_policy.html
- Unicode is still growing.

Locale data

- cultural data can be updated based on community voting
- cultural format results are not suited for serializing data, application protocols and storage
Stability Problems
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
- DO send and store non-localized format
  - Binary: 0x12345678
  - “Neutral” - ISO 8601 - “2019-04-08”
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
- DO send and store non-localized format
  - Binary: 0x12345678
  - “Neutral” - ISO 8601 - “2019-04-08”
- REMEMBER ® may not be a letter
Stability Problems

- “08-04-2019” may not parse as “8 kwì 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
- DO send and store non-localized format
  - Binary: 0x12345678
  - “Neutral” - ISO 8601 - “2019-04-08”
- REMEMBER ⚠ may not be a letter (isLetter()) in one Unicode version, but may later be defined.
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
- DO send and store non-localized format
  - Binary: 0x12345678
  - “Neutral” - ISO 8601 - “2019-04-08”
- REMEMBER ￼ may not be a letter (isLetter()) in one Unicode version, but may later be defined. Could cause difficulties if used to validate account names, …
Stability Problems

- “08-04-2019” may not parse as “8 kwi 2019”
- DON’T send localized data across the network between programs (other side may parse/format differently)
- DON’T store localized data on disk (later app version may parse/format differently)
- DO send and store non-localized format
  - Binary: 0x12345678
  - “Neutral” - ISO 8601 - “2019-04-08”
- REMEMBER ἀ may not be a letter (isLetter()) in one Unicode version, but may later be defined. Could cause difficulties if used to validate account names, ...
- DO Think carefully about where Unicode properties are used.
ICU4J vs JDK (0/2)

- ICU has functionality beyond JDK - See userguide.
ICU4J vs JDK (0/2)

- ICU has functionality beyond JDK - See userguide.
- Where there is overlap, in some cases JDK may be used instead of ICU.
ICU4J vs JDK (0/2)

- ICU has functionality beyond JDK - See userguide.
- Where there is overlap, in some cases JDK may be used instead of ICU.
  Example: Locale instead of ICU’s ULocale
## ICU4J vs JDK (1/2)

<table>
<thead>
<tr>
<th>JDK class</th>
<th>ICU class</th>
<th>ICU Benefits</th>
<th>Suggestion</th>
</tr>
</thead>
</table>
| java.lang.Character | com.ibm.icu.lang.UCharacter | - Latest Unicode standard  
                        | - More character properties support             | JDK OK: ICU as-needed |
| java.math.BigDecimal | com.ibm.icu.math.BigDecimal | - For backward compatibility only                 | ICU not recommended in new code                 |
                        | - Dictionary based word break (Thai, Lao, Chinese/Japanese) | JDK OK: ICU as-needed |
                        | - Faster comparison                             | ICU recommended                                |
### ICU4J vs JDK (2/2)

<table>
<thead>
<tr>
<th>JDK class</th>
<th>ICU class</th>
<th>ICU Benefits</th>
<th>Suggestion</th>
</tr>
</thead>
<tbody>
<tr>
<td>java.text.SimpleDateFormat</td>
<td>com.ibm.icu.text.SimpleDateFormat</td>
<td>• Abstract (skeleton) pattern (e.g. year-month only format)</td>
<td>JDK OK, ICU as-needed</td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.SimpleDateFormat</td>
<td>• Patterns for additional calendar types</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.SimpleDateFormat</td>
<td>• More field format types (e.g. narrow weekday, standalone month)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.SimpleDateFormat</td>
<td>• Capitalization control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.SimpleDateFormat</td>
<td>• Slower service object creation, format &amp; parse than JDK</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.MessageFormat</td>
<td>• Gender formatting (social applications)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.MessageFormat</td>
<td>• Named arguments (&quot;{filename}&quot; vs &quot;{4}&quot;)</td>
<td></td>
</tr>
<tr>
<td>java.text.DecimalFormat</td>
<td>com.ibm.icu.text.NumberFormat + RuleBasedNumberFormat</td>
<td>• More styles (e.g. scientific, currency spell out)</td>
<td>JDK OK, ICU as-needed</td>
</tr>
<tr>
<td>java.text.DecimalFormat</td>
<td>com.ibm.icu.text.DecimalFormat + RuleBasedNumberFormat</td>
<td>• Parse currency</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.DecimalFormat + RuleBasedNumberFormat</td>
<td>• Algorithmic numbering systems</td>
<td></td>
</tr>
<tr>
<td></td>
<td>com.ibm.icu.text.DecimalFormat + RuleBasedNumberFormat</td>
<td>• Slower service object creation, format &amp; parse than JDK</td>
<td></td>
</tr>
</tbody>
</table>
Action for You: Join our mailing lists!

http://site.icu-project.org/contacts
Action for You: Join our mailing lists!

http://site.icu-project.org/contact

Sample Code: bit.ly/iuc43-icu-samples

Presenter: Steven Loomis

- Social: @srl295
- Web site: git.io/srl295
- Email: srloomis@us.ibm.com

Presenter: Shane Carr

- Social: @sffc or @_sffc
- Web site: https://sffc.xyz
- Email: sffc@google.com / shane@unicode.org

Have a nice day!