**Java I18N**

## **Pendahuluan**

**Sebelumnya harus sudah paham materi terkait** :

* Java Dasar
* Java Object Oriented Programming
* Java Generic
* Java Collection
* Apache Maven
* Java Unit Test

**Agenda**

* Pengenalan Internationalization
* Locale Class
* Resource Bundle
* I18N di Date
* I18N di Number
* I18N di Currency
* Message Format
* Dan lain – lain

## **Pengenalan Internationalization**

* Internationalization atau biasa disingkat I18N, karena ada 18 karakter diantara hurup I dan N
* I18N adalah proses membuat desain aplikasi yang bisa beradaptasi sesuai dengan bahasa dan tempat dari pengguna aplikasinya tanpa harus melakukan perubahan pada kode program aplikasi yang kita buat
* Java sudah mendukung I18N, jadi kitab isa membuat aplikasi kita mendukung banyak bahasa ( misal bahasa Indonesia, Inggris, dan lain – lain)

**Karakteristik Aplikasi yang Mendukung I18N**

* Penambahan dukungan bahasa tidak perlu melakukan perubahan kode program Java
* Tampilan yang berubah text / label, seperti misal status, label di komponen UI, semuanya tidak di hardcode di kode program, melainkan disimpan di tempat diluar kode program, dan diambil secara dinamis sesuai dengan bahasa dan lokasi pengguna aplikasi
* Mengikuti kultur lokasi user, seperti format tanggal, waktu, angka dan mata uang
* Penambahan dukungan bahasa baru bisa dilakukan secara cepat

## **Locale**

* Di Java, bahasa dan negara sudah ada representasinya, namanya adalah class Locale
* Class locale ini bisa digunakan sebagai representasi bahasa dan negara
* Kenapa ada negara dan bahasa? Kenapa tidak negara saja, atau bahasa saja? Karena kadang ada bahasa yang sama untuk negara berbeda, atau bahasa berbeda untuk negara yang sama
* Locale mengikuti standar IETF BPC 47, yang terdiri dari RPF 4647 “Matching of Language Tags” dan RPF 5654 “Tags for Identifying Languages”

**Penulisan Bahasa**

* Penulisan bahasa di Locale menggunakan ISO 639 alpha 2 atau alpha 3
* Ketika sebuah bahasa memiliki kode alpha 2 dan alpha 3, maka yang harus digunakan adalah alpha 2
* <https://www.loc.gov/standards/iso639-2/php/code_list.php>

**Penulisan Negara**

* Penulisan negara di Locale menggunakan ISO 3166 alpha-2
* <https://www.iban.com/country-codes>

## **Resource Bundle**

* ResourceBundle adalah class yang digunakan sebagai bundle yang berisikan data yang bisa berubah sesuai dengan Locale yang di pilih
* ResourceBundle cocok digunakan untuk menyimpan data I18N, dimana semua data akan disimpan dalam properties file

**Properties File**

* Saat kita menggunakan ResourceBundle, kita perlu membuat properties file di dalam resource sebagai tempat untuk ResourceBundle melakukan lookup data
* Misal kita membuat file properties dengan nama **message.properties** di dalam resources

**I18N di ResourceBundle**

* Kode sebelumnya kita hanya menggunakan ResourceBundle sebagai object untuk mengambil data dari properties file, kalo seperti ini ada bedanya dengan class Properties
* Yang membedakan, dalam ResourceBundle, kitab isa melakukan I18N
* Namun untuk melakukan itu, kita perlu menyediakan properties file sesuai dengan Locale nya
* Misal jika kita ingin mendukung bahasa in dan negara ID, maka kita perlu membuat message\_in\_ID.properties
* File nya butuh diawali dengan message karena sebelumnya kita menggunakan nama file message.properties

**Jika Tidak ada Properties untuk Locale**

* Apa yang terjadi jika ada Locale yang tidak tersedia dalam file properties nya?
* Misal kita akan menggunakan Locale en\_US, sedangkan tidak ada file message\_en\_US.properties nya ?
* Jika itu terjadi, secara default ResourceBundle akan menggunakan message.properties

**Amankah Membuat ResorceBundle Terus?**

* Cara membuat ResourceBundle menggunakan static method getBundle()
* Di dalam implementasi getBundle() telah diimplementasikan caching, artinya jika kita mengambil resource yang sama dengan locale yang sama, tidak akan dibuat ulang, melainkan akan menggunakan ResourceBundle yang sudah tersedia
* Hal ini menjadi aman jika kita selalu membuat ResourceBundle terus menerus

## **I18N di Date dan Time**

* I18N di tipe data String mungkin sederhana, bagaimana jika tipe data Date atau Waktu?
* Di java kita tahu bahwa tipe data Date dan Time direpresentasikan dalam class Data\
* Bagaimana misal jika kita ingin menampilkan tanggal dalam bahasa Indonesia, bahasa inggris, atau yang lainnya?
* Di java, I18N di Date sudah disediakan menggunakan DateFormatter

**DateFormat Class**

* DateFormat merupakan class yang bisa kita gunakan untuk melakukan parsing dari String ke Date, atau format dari Date ke String
* DateFormat mendukung I18N jika misal kita ingin melakukan format dan parse, sehingga bisa menerima hari dengan data Sunday, Monday, atau Minggu, Senin
* DateFormat adalah abstract class, salah satu implementasinya adalah class SimpleDateFormat
* https://docs.oracle.com/javase/8/docs/api/java/text/SimpleDateFormat.html

## **I18N di Number**

* Apakah perlu I18N di number?
* Yup perlu, karena kadang beda bahasa beda saat penulisan number nya
* Contoh, di Indonesia menggunakan .(titik) sebagai pemisah ribuan, dan , (koma) untuk pemisah bilangan pecahan, sedangkan di Amerika, itu terbalik

**NumberFormat Class**

* Untuk melakukan I18N di Java, kita bisa menggunakan class NumberFormat
* NumberFormat adalah abstract class, untuk membuat object NumberFormat, kitab isa menggunakan static method getInstance() milik class NumberFormat

## **I18N di Currency**

* Java mendukung I18N untuk mat uang atau currency
* Ada 2 class ynag bis akita gunakan untuk melakukan I18N di Java, yaitu class Currency dan class NumberFormat

**Currency Class**

* Currency class adalah implementasi standard dari ISO 4217 currency codes
* <https://www.iso.org/iso-4217-currency-codes.html>
* Dengan menggunakan class ini, kitab isa mengetahui mata uang. Kode mata uang sampai simbol yang digunakan
* Untuk membuat object Currency, kita wajib menggunakan Locale yang memiliki negara, karena memang mata uang tergantung dari negaranya, bukan bahasanya
* Constructor Currency class adalah private, jadi untuk membuat Currency, kita harus menggunakan static method getInstance()

**Currency di NumberFormat**

* Currency class hanya digunakan untuk mendapatkan data nama mata uang, satuan, dan symbol, bagaimana jika kita ingin melakukan format number dalm bentuk mata uang?
* Jika dalam kasus ini, lebih baik menggunakan NumberFormat
* NumberFormat juga tidak hanya bisa digunakan untuk format dan parsing number, tapi juga number dalam bentuk currency
* Untuk membuat NumberFormat yang mendukung currency, kitab isa gunakan static method getCurrencyInstance();

## **Message Format**

* Ketika melakukan I18N, kadang kita membutuhkan data yang bentuknya dinamis, sesuai dengan parameter, misal kita butuh String Hi {nama}, Selamat Datang di {aplikasi}
* Jika kita menggunakan Rsource Bundle, hal tersebut tidak bisa dilakukan secara otomatis, kita perlu melakukannya secara manual, dengan cara menggabungkan beberapa String
* Untungnya di Java terdapat class bernama MessageFormat
* MessageFormat class ini digunakan untuk melakukan substitusi data parameter, namun data string awalnya perlu ditambahkan penanda parameter

**Parameter di MessageFormat**

* Penggunakan parameter di MessageFormat, menggunakan kurung kurawal buka, diikuti index (dimulai dari 0) dan diakhiri dengan kurung kurawal tutup, misal :
* Hi {0}, Selamat Datang di {1}
* Jika kita menggunakan nomor index yang sama, artinya data akan menggunakan parameter yang sama, misal :
* Hi {0}, Anda bisa mencari data Anda dengan mengetik “{0}” di pencarian

**Integrasi Dengan ResourceBundle**

* Sebenarnya sekilas kita perhatikan jika MessageFormat itu tidak ada hubungannya dengan I18N
* Namun, kita bisa mengkombinasikan dengan ResourceBundle, dan menyimpan pattern nya di dalam ResourceBundle properties file
* Dengan demikian, MessageFormat seakan-akan memiliki kemampuan I18N