[Bài đọc] Localization

Trong thời đại toàn cầu hóa, các ứng dụng trở nên có nhu cầu được triển khai ở nhiều quốc gia và khu vực khác nhau. Có hai thuật ngữ cần được nhắc tới ở đây. Đầu tiên là Internationalization - quốc tế hóa, thường được viết tắt là i18n vì được bắt đầu bằng chữ i, sau đó tới 18 chữ cái và kết thúc bằng ký tự n. I18n là một kỹ thuật để phát triển ứng dụng hỗ trợ nhiều định dạng ngôn ngữ và dữ liệu mà không cần phải viết lại logic lập trình.

Thuật ngữ thứ hai là Localization - bản địa hóa, tương tự như trên, được gọi tắt là l10n, là kỹ thuật thích ứng với một ứng dụng quốc tế hóa để hỗ trợ một địa phương cụ thể.

Java có thiết kế i18n, sử dụng bảng mã Unicode cho các ký tự và chuỗi. Do đó tạo ứng dụng quốc tế trong Java là rất dễ dàng. Cách triển khai quốc tế hóa phụ thuộc vào lượng dữ liệu cần trình bày bằng các ngôn ngữ khác nhau. Từ đó có hai cách tiếp cận:

1. Nếu một lượng lớn dữ liệu (cần trình bày bằng các ngôn ngữ khác nhau) là dữ liệu tĩnh, hãy tạo cho mỗi một ngôn ngữ một văn bản. Cách tiếp cận này đơn giản, và không được nhắc tới ở đây.
2. Nếu phần “tĩnh” là ít, hãy tách riêng các phần văn bản như nhãn của element, các thông báo lỗi, … thành tệp dữ liệu. Mỗi tệp chứa bản dịch của tất cả các phần tử, cho một ngôn ngữ. Sau đó ứng dụng sẽ truy xuất động từng phần tử một. Lợi thế rất rõ ràng, mỗi phần tử văn bản có thể được chỉnh sửa dễ dàng mà không cần biên dịch lại ứng dụng. Đây là kỹ thuật sẽ được thảo luận trong bài đọc này.

Bài đọc này bắt đầu bằng việc giải thích một địa phương (locale) là gì. Tiếp theo là kỹ thuật quốc tế hóa các ứng dụng, tiếp theo là một ví dụ bằng Spring MVC.

**Locale**

Lớp java.util.Locale đại diện cho một local. Có ba phần chính của một locale: ngôn ngữ, quốc gia, và biến thể. Ngôn ngữ chắc chắn là phần quan trọng nhất, tuy nhiên đôi khi ngôn ngữ không đủ để phân biệt được một địa phương. Ví dụ: ngôn ngữ Anh được nói ở cả Hoa Kỳ lẫn Anh, ngôn ngữ Tiếng Trung được nói cả ở Trung Quốc lẫn Đài Loan, nhưng nó không giống nhau ở mỗi nước. Vì vậy việc xác định cả ngôn ngữ lẫn quốc gia là cần thiết.

Biến thể là mã đi theo nhà cung cấp hay trình duyệt cụ thể. Ví dụ WIN, MAC, POSIX. Trong trường hợp locale có hai biến thể, thông tin biến thể được đặt theo quy ước cách nhau bởi dấu gạch dưới, và đặt biến thể quan trọng hơn lên trước, ví dụ Traditionla\_WIN.

Để xây dựng một đối tượng Locale, sử dụng một trong số các constructor của lớp Locale (hoặc xây dựng builder theo pattern buider):

public Locale(String language);

public Locale(String language, String country, String variant);

public Locale(String language, String country);

language là mã theo chuẩn ISO về mã ngôn ngữ. Bảng sau đây hiển thị mã của một số ngôn ngữ phổ biến.

| **Code** | **Language** |
| --- | --- |
| de | German |
| el | Greek |
| en | English |
| es | Spanish |
| fr | French |
| hi | Hindi |
| it | Italian |
| ja | Japanese |
| nl | Dutch |
| pt | Portuguese |
| ru | Russian |
| zh | Chines |

country là mã theo chuẩn ISO về mã quốc gia, là mã gồm 2 chữ cái viết hoa, được chỉ định trong [ISO 3166](http://userpage.chemie.fu-berlin.de/diverse/doc/ISO_3166.html). Bảng sau liệt kê một số mã quốc gia trong ISO 3166.

| **Country** | **Code** |
| --- | --- |
| Australia | AU |
| Brazil | BR |
| Canada | CA |
| China | CN |
| Egypt | EG |
| France | FR |
| Germany | DE |
| India | IN |
| Mexico | MX |
| Switzerland | CH |
| Taiwan | TW |
| United Kingdom | GB |
| United States | US |

Ví dụ, để tạo một đối tượng Locale đại diện cho ngôn ngữ tiếng Anh vùng Canada:

Locale locale = new Locale("en", "CA");

Bên cạnh đó, lớp Locale cung cấp các hằng số trả về đối tượng locale cho các quốc gia và ngôn ngữ cụ thể mà có thể sử dụng như trong ví dụ dưới đây:

Locale locale = Locale.CANADA\_FRENCH;

Lớp locale cũng có phương thức trả về đối tượng locale mặc định cho máy tính đang thực thi:

Locale locale = Locale.getDefault();

Last modified: Wednesday, 27 March 2019, 3:51 PM