Chương 1 : Giới thiệu về Swing

**Swing là gì?**

Nếu truy cập vào trang chủ Java (<http://java.sun.com/>), ta sẽ nhận thấy Swing là một tập hợp các thành phần đồ họa có thể tùy chỉnh mà giao diện và hành vi xử lý (look and feel – L&F) có thể được quyết định tại thời điểm chạy chương trình. Tuy nhiên, trong thực tế Swing cung cấp nhiều hơn thế . Swing là bộ công cụ GUI (GUI toolkit) thế hệ kế tiếp mà Sun Microsystems tạo ra nhằm giúp lập trình viên phát triển các ứng dụng doanh nghiệp bằng ngôn ngữ Java. Phát triển phần mềm doanh nghiệp, lập trình viên có thể sử dụng Swing để tạo ra nhựng ứng dụng Java quy mô lớn có chứa một mảng rộng những thành phần mạnh mẽ. Ngoài ra, bạn có thể dễ dàng mở rộng hoặc sửa đổi những thành phần này để kiểm soát giao diện và hành vi của chúng.

Swing không phải là một từ viết tắt. Swing đại diện cho sự lực chọn hợp tác của các nhà thiết kế khi dự án được bắt tay thực hiện vài cuối năm 1996. Trên thực tế, Swing kà một phần của tập hợp các sản phẩm Java có tên gọi là Java Foundation Class (JFC), JFC kết hợp nhiều tính năng của Netscape Foundation Internet Class (IFC) cũng như các khía cạnh thiết kế từ bộ phận Taligent của IBM và Lighthouse Design. Swing được phát triển tích cực từ giai đoạn beta của bản JDK (Bộ công vụ dành cho người phát triển ứng dụng Java – Java Development Kit) 1.1, khoảng mùa xuân năm 1997. Các API Swing được đưa vào bản beta vào nửa cuối năm 1997 và được phát hành tháng 3/1998. Tại thời điểm phát hành, các thư viện Swing 1.0 chứa gần 250 lớp (class) và 80 giao diện (interface). Kể từ đó thư viện Swing vẫn tiếp tục tăng trưởng.

Mặc dù Swing được phát triển riêng biệt với lõi của JDK, nhưng để chạy được thì bản JDK tối thiểu là 1.1.5. Swing xây dựng dựa trên mô hình sự kiện (event model) của chuỗi phiên bản JDK 1.1; ta không thể sử dụng các thư viện Swing với phiên bản JDK 1.0.2 cũ hơn. Ngoài ra, ta phải có một trình duyệt kích hoạt Java 1.1 để hỗ trợ các Swing applet. Bản phát hành Java 2 SDK 1.4 bao gồm nhiều lớp Swing cập nhật cùng với một vài tính năng mới. Swing được tích hợp đầy đủ vào cả công cụ của nhà phát triển và môi trường chạy của tất cả các phiên bản Java 2 (SDK 1.2 và phiên bản cao hơn), bao gồm cả Java Plug-In

**Tính năng bổ sung của Swing**

Một số tính năng phân biệt Swing với các thành phần AWT cũ hơn:

* Swing có nhiều thành phần mới, như bảng (table), cây (tree), thanh trượt (slider), spinner, thanh tiến trình (progress bar), internal frame và thành phần văn bản (text).
* Thành phần Swing hỗ trợ khả năng thay thế thành phần inset bằng một số lượng tùy ý các đường bao lồng nhau.
* Thành phần Swing có thể chứa các tooltip. Tooltip là hộp văn bản xuất hiện trong giây lát khi con trỏ chuột nằm bên trong vùng hoạt động của thành phần. Tooltip có thể được sử dụng để cung cấp thêm thông tin về thành phần chưa biết.
* Ta có thể tùy ý ràng buộc sự kiện bàn phím với thành phần, định nghĩa cách chúng phản ứng với các tổ hợp phím khác nhau trong những điều kiện nhất định
* Có thêm phần hỗ trợ gỡ lỗi cho việc render các thành phần Swing lightweight của riêng bạn

**Gói và lớp của Swing**

javax.accessibility

Chứa các lớp và giao diện có thể được sử dụng để cho phép công nghệ hỗ trợ tương tác với thành phần Swing. Công nghệ hỗ trợ bao gồm một loạt các mục, từ trình đọc văn bản bằng âm thanh tới màn hình phóng đại. Mặc dù về mặt bản chất thì không phải là 1 phần của Swing, song các lớp hỗ trợ truy cập được sử dụng rộng rãi trên các thành phần của Swing.

javax.swing

Chứa các thành phần Swing cốt lõi, bao gồm hầu hết giao diện mô hình và lớp hỗ trợ

javax.swing.border

Chứa các định nghĩa về lớp border trừu tượng cũng như tám border (đường bao) được định nghĩa trước. Đường bao không phải là thành phần; mà là những phần tử đồ họa đặc biệt mà Swing coi như các thuộc tính và đặt xung quanh cá thành phần ở vùng không gian đệm bên ngoài thành phần (inset). Nếu muốn tạo ra đường bao riêng, bạn có thể tạo lớp con của một đường bao hiện có trong gói này hoặc lập trình tạo một lớp mới từ đầu.

javax.swing.colorchooser

Bao gồm hỗ trợ cho thành phần JColorChooser.

javax.swing.event

Định nghĩa một số listener và sự kiện mới mà các thành phần Swing dùng để giao tiếp thông tin không đồng bộ giữa các lớp. Để tạo ra sự kiện riêng, bạn có thể tạo lớp con của các sự kiện khác nhau trong gói này hoặc viết lớp sự kiện của riêng bạn

javax.swing.filechooser

Có hỗ trợ cho thành phần JfileChooser

javax.swing.plaf

Định nghĩa những ohần tử đặc trưng tạo nên L&F pluggable cho mỗi thành phần. Những gói con của gói này được sử dụng để render các L&F riêng lẻ cho mỗi thành phần trên cơ sở đa nền tảng (platform-by-platform). (Phần thực thi cụ thể của L&F trong Window và Motif nằm trong các gói con của com.sun.java.swing.plaf, và L&F của MacOS nằm trong gói com.apple.mrj.swing).

javax.swing.tanle

Cung cấp mô hình (model) và hiển thị (view) thành phần của bảng (table), cho phep2 bạn bố trí và xếp đặt thông tin dưới dạng lưới giống như một bảng tính (spreadsheet). Bằng việc sử dụng các lớp ở cấp thấp hơn, bạn có thể sửa đổi việc lựa chọn và hiển thị bảng, cũng như cách hiển thị thông tin ở mỗi dòng, mỗi ô của bảng.

javax.swing.text

Cung cấp các lớp và giao diện cốt lõi liên quan đến văn bản, hỗ trợ một thiết kế chung dạng tài liệu/hiển thị (document/view).

javax.swing.text.html

Được sử dụng đặc biệt trong việc đọc và định dạng văn bản HTML thông qua một trình chỉnh sửa phụ trợ (editor kit).

javax.swing.text.html.parser

Chứa các lớp hỗ trợ phân tích mã HTML.

javax.swing.text.rtf

Được sử dụng đặc biệt trong việc đọc và định dạng văn bản Rich Text Formal (RTF) thông qua một trình chỉnh sửa phụ trợ.

javax.swing.tree

Định nghĩa các mô hình và hiển thị cho thành phần cây phân cấp, thành phần này có thể biểu diễn một cấu trúc file hoặc chuỗi các thuộc tính.

javax.swing.undo

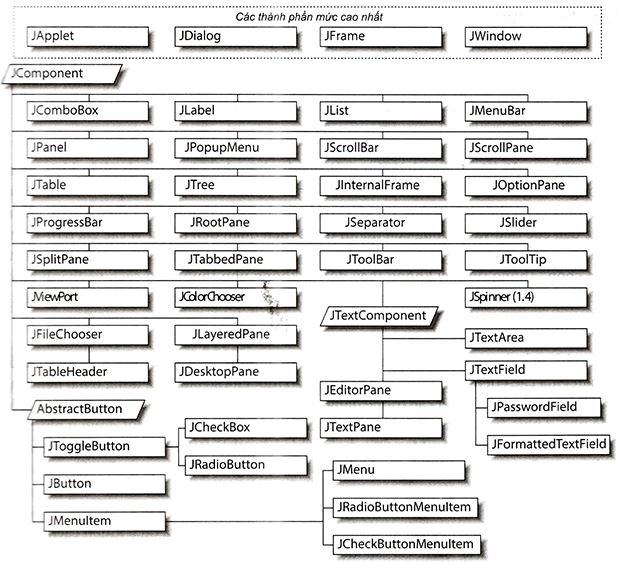
Chứa các chức năng cần thiết để cài đặt các chứa năng undo (hoàn tác).

Cho tới nay thì gói được sử dụng một cách rộng rãi nhất là gói javax.swing. Trên thực tế, hầu hết các thành phần trong Swing – kể cả một vài lớp tiện ích (utility class) – đều được đặt trong gói này. (Ngoại lệ duy nhất là trường bao (border) và lớp hỗ trợ cho cây, bảng và thành phần văn bản). Do về sau các thành phần cung cấp khả năng mở rộng tốt hơn và thường có nhiều lớp lớn hơn để làm việc, tương tác nên các lớp này được chia thành những gói riêng biệt

**Phân cấp lớp**

Thoạt nhìn, cây phân lớp Swing rất giống AWT. Mỗ thành phần Swing và một thành phần tưng đương trong AWT có cùng tên, ngoại trừ ký tự đầu tiên trong Swing có thêm chữ J viết hoa. Trong hầu hết các trường hợp, nếu thay thế các thành phần AWT thì thành phần Swing có thể được sử dụng như một sự thay thế dạng drop-in (không cần điểu chỉnh). Tuy nhiên, giữ thành phần Swing và thành phần của AWT có những điểm khác biệt cơ bản.

Ta có thể để ý thấy rằng số lượng frame và pane trong Swing tăng lên. Ví dụ với các internal frame (frame bên trong). Swing hỗ trợ đặt một frame vào một frame khác – điề này thường được ám chỉ như một giao diện đa tài liệu (multiple document interface – MDI) trong Microsoft. Bạn có thể tùy ý chỉ định internal frame này vào các tầng xếp theo chiều thẳng đứng; các tầng này sẽ quyết địnhinternal frame nào sẽ xuất hiện ở trên. Trên thực tế, kể cả frame đơn giản nhất – Jframe – cũng hỗ trợ tầng bằng cách cung cấp layered pane (pane xếp tầng), trên đó bạn có thể chỉ định vị cho các phần tử khác nhau khi phát triển ứng dụng.

**Lớp JComponent**

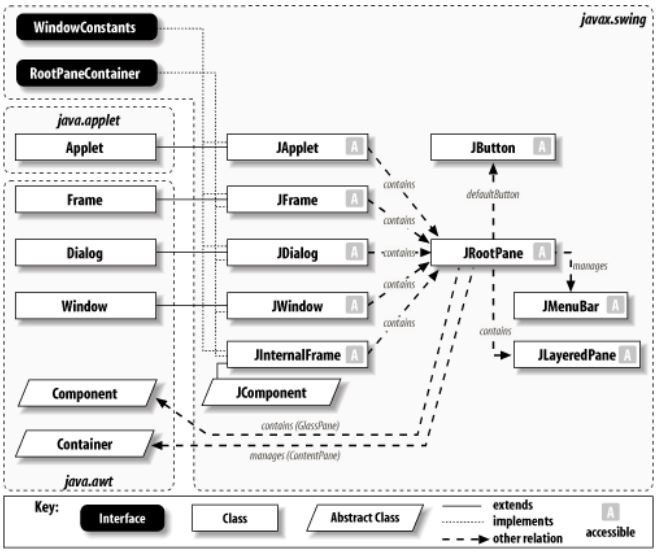
JComponent là lớp trừu tượng mà hầu như mọi thành phần Swing đều kế thừa: JComponent cung cấp nhiều chức năng cơ bản phổ biến cho toàn bộ thư viện thành phần Swing (Swing component library). Cũng như java.awt.JComponent đóng vai trò là framework cho hầu hết thành phần AWT, lớp java.swing.JComponent cũng đóng vai trò tương tự thành phần Swing.

Bởi vì JComponent kế thừa lớp Container nên nhiều thành phần Swing có thể đóng vai trò như là container của các thành phần AWT và Swing khác. Các thành phần này có thể được thêm thông qua phương thức add() truyền thống của Container. Thêm nữa JComponemt có thể được định vị với bất kỳ trình quản lý bố cục nào của Java ở bên trong container. Các thuật ngữ sau này vẫn giữ nguyên ý nghĩa : Con là thành phần được thêm vào một container (container này là cha của thành phần đó). Tương tự, bất kỳ thành phần nào đó ở mức cao hơn trong cây được gọi là tổ tiên, trong khi bất kỳ yhành phần nào thấp hơn được gọi là con cháu.

**Root Pane**

Hầu hết các container khác trong Swing (JFrame, JApplet, JWindow, JDialog và thậm chí là JInternalFrame) đều chứa thể hiện của một lớp khác (JRootPane) như thành phần duy nhất của chúng và thực thi giao diện chung là RootPaneContainer.

Ta hãy dành thời gian để tìm hiểu tổng thể về các lớp giao diện tạo nên container gốc (root container) của Swing. Hình … cho thấy JApplet, JFrame, JDialog và JWindow không kế thừa lớp JComponent như các thành phần Swing khác, mà kế thừa từ những thành phần AWT tương ứng, đóng vai trò như cửa sổ giao diện người dùng ở mức cao nhất. Điề này ngụ ý rằng những thành phần này (không giống như thành phần lightweight Swing) có các đối tượng AWT tương ứng của nền tảng.

Chú ý rằng nhưng Swing container này (cũng như là JInternal Frame) thực thi một giao diện chung là RootPaneContainer. Giao diện này cung cấp truy cập tới các thuộc tính của JRootPane. Thêm nữa là mỗi container trong năm container này đều sử dụng JRootPane làm container (thành phần chứa) “thực sự” cho các thành phần con mà container này quản lý.

*Hình . Sơ đồ lớp của container “gốc” trong Swing*

**Lớp JFrame**

Hầu hết các Swing container của ứng dụng Java đều là lớp JFrame. JFrame cung cấp một cửa sổ ở mức cao nhất xó một tiêu đề, đường bao và các yếu tố phụ thuộc nền tảng khác (ví dụ như các nút bấm thu nhỏ, phóng to và đóng). Bởi vì lớp này sử dụng JRootPane như là một lớp con duy nhất nên làm việc với một JFrame khá là khác biệt so với khi làm việc với một Frame của AWT. Một JFrame rỗng được mình họa như sau.

*Hình . Các thể hiện JFrame rỗng trên nền tảng Unix , Mac và Window*

**Lớp JPanel**

JPanel là lớp kế thừa JComponent (mà JComponent kế thừa java.awt.Container) được sử dụng để nhóm các thành phần khác lại với nhau. JPanel có hều hết các phần thực thi từ lớp cha. Thông thường, chúng ta sẽ khởi tạo đối tượng JPanel, thiết lập trình quản lý bố cục (có thể được thiết lập trong phương thức khởi tạo và mặc định là một FlowLayout) và thêm các thành phần vào JPanel thông qua phương thức add() được kế thừa từ Container.

**Nhãn**