

# Lập trình Java Mobile

GV: ThS. Phan Nguyệt Minh  
[minhpn@uit.edu.vn](mailto:minhpn@uit.edu.vn)

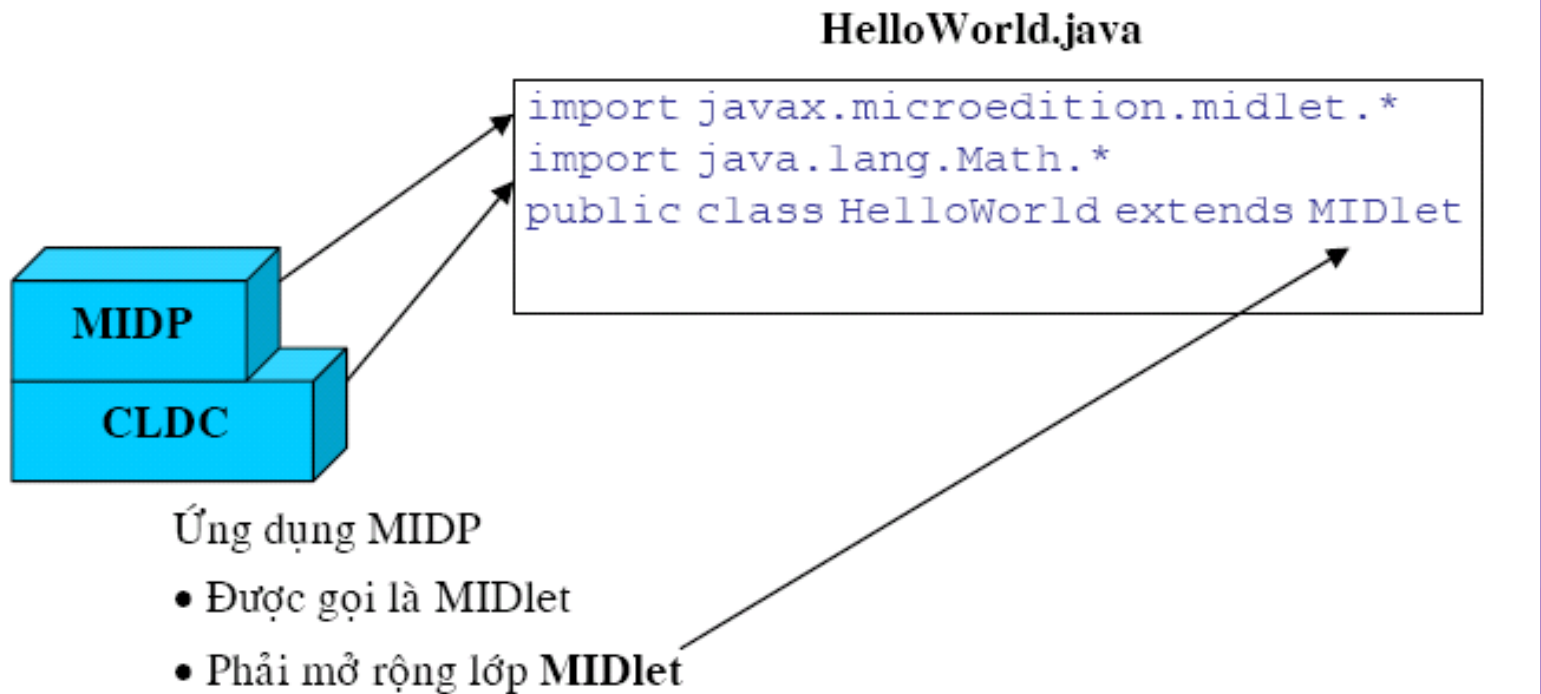
<http://courses.uit.edu.vn>

# Lập trình với J2ME

- ▶ Tổng quan về MIDLet
- ▶ Lập trình giao diện MIDP

# Tổng quan về MIDlet

- ▶ Các ứng dụng J2ME được gọi là MIDlet (Mobile Information Device applet)



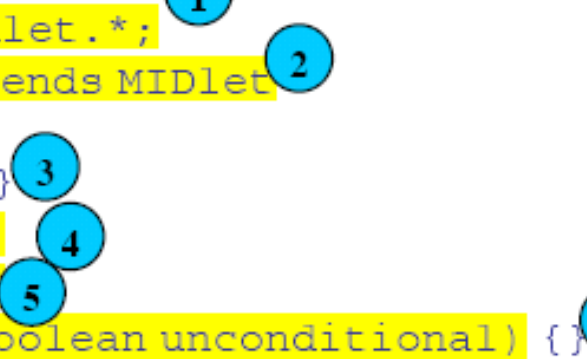
# Tổng quan về MIDlet

- ▶ Thông báo import dùng để truy xuất các lớp của CLDC và MIDP.
- ▶ Lớp chính của ứng dụng được định nghĩa là lớp mở rộng lớp MIDlet của MIDP. Có thể chỉ có một lớp trong ứng dụng mở rộng lớp này. Lớp MIDlet được trình quản lý ứng dụng trên điện thoại di động dùng để khởi động, dừng, và tạm dừng MIDlet (ví dụ, trong trường hợp có cuộc gọi đến).

# Bộ khung MIDlet (MIDlet Skeleton)

- ▶ Một MIDlet là một lớp Java mở rộng (extend) của lớp trừu tượng `javax.microedition.midlet.MIDlet` và thực thi (implement) các phương thức `startApp()`, `pauseApp()`, và `destroyApp()`.

```
import javax.microedition.midlet.*;
public class MIDletExample extends MIDlet
{
    public MIDletExample() {}
    public void startApp() {}
    public void pauseApp() {}
    public void destroyApp(boolean unconditional) {}
}
```



The diagram illustrates the structure of a MIDlet skeleton with numbered callouts:

- 1: Points to the import statement `import javax.microedition.midlet.*;`
- 2: Points to the class declaration `public class MIDletExample extends MIDlet`
- 3: Points to the constructor `public MIDletExample() {}`
- 4: Points to the `startApp()` method `public void startApp() {}`
- 5: Points to the `pauseApp()` method `public void pauseApp() {}`
- 6: Points to the `destroyApp()` method `public void destroyApp(boolean unconditional) {}`

# Bộ khung MIDlet (MIDlet Skeleton)

## 1) Phát biểu import

- ▶ Các phát biểu import được dùng để include các lớp cần thiết từ các thư viện CLDC và MIDP.

## 2) Phần chính của MIDlet

- ▶ MIDlet được định nghĩa như một lớp mở rộng lớp MIDlet.

# Bộ khung MIDlet (MIDlet Skeleton)

## 3) Hàm tạo (Constructor)

- ▶ Hàm tạo chỉ được thực thi một lần khi MIDlet được khởi tạo lần đầu tiên.
- ▶ Hàm tạo sẽ không được gọi lại trừ phi MIDlet thoát và sau đó khởi động lại.

# Bộ khung MIDlet (MIDlet Skeleton)

## 4) startApp()

- ▶ Phương thức startApp() được gọi bởi bộ quản lý ứng dụng khi MIDlet được khởi tạo, và mỗi khi MIDlet trở về từ trạng thái tạm dừng. Nói chung, các biến toàn cục sẽ được khởi tạo lại trừ hàm tạo bởi vì các biến đã được giải phóng trong hàm pauseApp(). Nếu không thì chúng sẽ không được khởi tạo lại bởi ứng dụng.



# Bộ khung MIDlet (MIDlet Skeleton)

## 5) pauseApp()

- ▶ Phương thức pauseApp() được gọi bởi bộ quản lý ứng dụng mỗi khi ứng dụng cần được tạm dừng (ví dụ, trong trường hợp có cuộc gọi hoặc tin nhắn đến).
- ▶ Cách thích hợp để sử dụng pauseApp() là giải phóng tài nguyên và các biến để dành cho các chức năng khác trong điện thoại trong khi MIDlet được tạm dừng.
- ▶ Cần chú ý rằng khi nhận cuộc gọi đến hệ điều hành trên điện thoại di động có thể dừng KVM thay vì dừng MIDlet. Việc này không được đề cập trong MIDP mà đó là do nhà sản xuất quyết định sẽ chọn cách nào.

# Bộ khung MIDlet (MIDlet Skeleton)

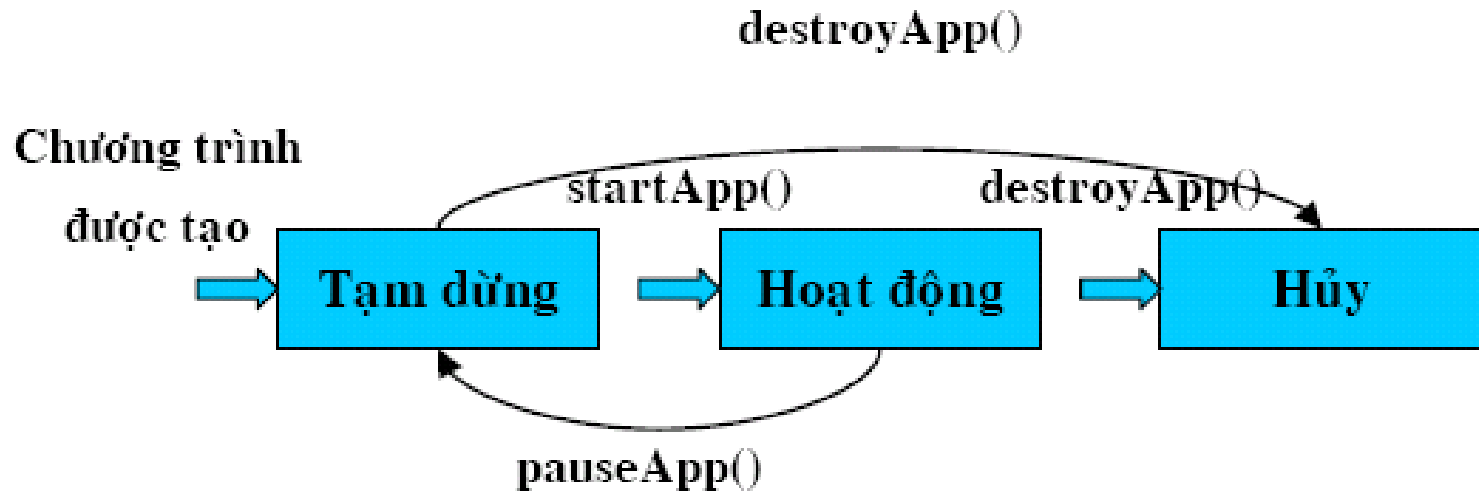
## 6) destroyApp()

- ▶ Phương thức `destroyApp()` được gọi khi thoát MIDlet. (ví dụ khi nhấn nút exit trong ứng dụng).
- ▶ Không thật sự xóa ứng dụng khỏi điện thoại di động.
- ▶ Phương thức `destroyApp()` chỉ nhận một tham số Boolean. Nếu tham số này là `true`, MIDlet được tắt vô điều kiện. Nếu tham số là `false`, MIDlet có thêm tùy chọn từ chối thoát bằng cách gửi ra một ngoại lệ `MIDletStateChangeException`.

# Các trạng thái khác nhau của MIDlet

- ▶ **Tạo (Created)** → Hàm tạo `MIDletExample()` được gọi một lần
- ▶ **Hoạt động (Active)** → Phương thức `startApp()` được gọi khi chương trình bắt đầu hay sau khi tạm dừng
- ▶ **Tạm dừng (Paused)** → Phương thức `pauseApp()` được gọi. Có thể nhận các sự kiện timer.
- ▶ **Hủy (Destroyed)** → Phương thức `destroy()` được gọi.

# Chu kỳ sống của MIDlet



# Tập tin JAR

- ▶ Các lớp đã biên dịch của ứng dụng MIDlet được đóng gói trong một tập tin JAR (Java Archive File). Đây chính là tập tin JAR được download xuống điện thoại di động.
- ▶ Tập tin JAR chứa tất cả các tập tin class từ một hay nhiều MIDlet, cũng như các tài nguyên cần thiết. Hiện tại, MIDP chỉ hỗ trợ định dạng hình .png (Portable Network Graphics).

# Tập tin JAR

- ▶ Tập tin JAR cũng chứa tập tin kê khai (manifest file) mô tả nội dung của MIDlet cho bộ quản lý ứng dụng.
- ▶ Chứa các tập tin dữ liệu mà MIDlet cần.
- ▶ Tập tin JAR là toàn bộ ứng dụng MIDlet. MIDlet có thể load và triệu gọi các phương thức từ bất kỳ lớp nào trong tập tin JAR, trong MIDP, hay CLDC. Nó không thể truy xuất các lớp không phải là bộ phận của tập tin JAR hay vùng dùng chung của thiết bị di động.

# Tập tin manifest và tập tin JAD

- ▶ Tập tin kê khai (manifest.mf) và tập tin JAD (Java Application Descriptor) mô tả các đặc điểm của MIDlet.
- ▶ Sự khác biệt của hai tập tin này là tập tin kê khai là một phần của tập tin JAR còn tập tin JAD không thuộc tập tin JAR.
- ▶ Ưu điểm của tập tin JAD là các đặc điểm của MIDlet có thể được xác định trước khi download tập tin JAR. Nói chung, cần ít thời gian để download một tập tin văn bản nhỏ hơn là download một tập tin JAR.

# Tập tin manifest và tập tin JAD

- ▶ Như vậy, nếu người dùng muốn download một ứng dụng không được thiết bị di động hỗ trợ (ví dụ, MIDP 2.0), thì quá trình download sẽ bị hủy bỏ thay vì phải đợi download hết toàn bộ tập tin JAR.
- ▶ Tập tin JAD chứa cùng thông tin như tập tin manifest. Nhưng nó nằm ngoài tập tin JAR.



# Tập tin manifest và tập tin JAD

## Ví dụ một tập tin manifest.mf:

MIDlet-Name: CardGames

MIDlet-Version: 1.0.0

MIDlet-Vendor: Sony Ericsson

MIDlet-Description: Set of Card Games

MIDlet-Info-URL: <http://www.semc.com/games>

MIDlet-Jar-URL: <http://www.semc.com/j2me/games>

MIDlet-Jar-Size: 1063

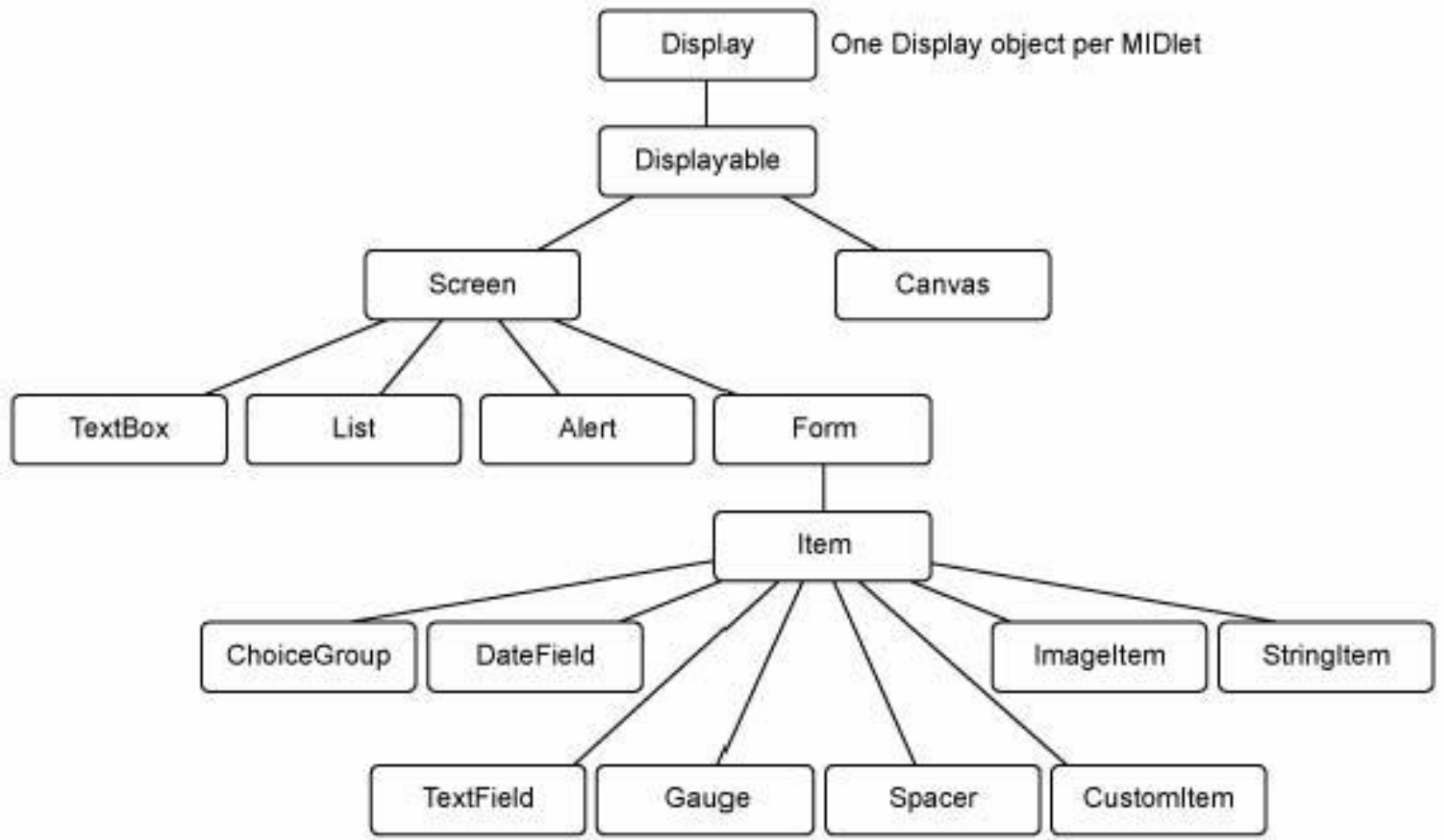
MicroEdition-Profile: MIDP-1.0

MicroEdition-Configuration: CLDC-1.0

MIDlet-1: Solitaire, /Sol.png, com.semc.Solitaire

MIDlet-2: BlackJack, /Blkjk.png, com.semc.BlackJack

# Giao diện người dùng trong MIDP



# Display

- ▶ Mỗi MIDlet có một tham chiếu đến một đối tượng Display.
- ▶ Display là đối tượng có nhiệm vụ quản lý việc hiển thị của màn hình.
- ▶ Quyết định danh sách các thành phần cần xuất hiện trên màn hình cũng như thời điểm phù hợp để hiển thị.

# Displayable

- ▶ Mặc dù mỗi MIDlet có thể có rất nhiều đối tượng Displayable.
- ▶ Một đối tượng Display có thể hiển thị bao nhiêu đối tượng Displayable tùy ý.
- ▶ Đối tượng Displayable là có thể nhìn thấy được một cách trực quan trên màn hình.
- ▶ MIDP có chứa 2 lớp con của Displayable là Screen và Canvas

# Screen

- ▶ Một đối tượng Screen không phải là một đối tượng hiện ra trên thiết bị, mà lớp Screen này sẽ được thừa kế bởi các thành phần hiển thị ở mức cao, chính các thành phần này sẽ được hiển thị ra trên màn hình.

# Form, Item

- ▶ Một Form đơn giản là một khung chứa các thành phần, mà mỗi thành phần được thừa kế từ lớp Item.
  - ▶ DateField
  - ▶ Gauge
  - ▶ StringItem
  - ▶ TextField
  - ▶ ChoiceGroup
  - ▶ Image và ImageItem

# DateField

- ▶ Thành phần DateField cung cấp một phương tiện trực quan để thao tác đối tượng Date được định nghĩa trong `java.util.Date`.



# DateField

- ▶ `DateField(String label, int mode)` `DateField(String label, int mode, TimeZone timeZone)`
- ▶ Các mode tương ứng của lớp `DateField` gồm:
  - ▶ `DateField.DATE_TIME`: cho phép thay đổi ngày giờ
  - ▶ `DateField.TIME`: chỉ cho phép thay đổi giờ
  - ▶ `DateField.DATE`: chỉ cho phép thay đổi ngày



# DateField

► Ví dụ:

```
private DateField dfAlarm;
```

```
// Tạo đối tượng DateField cho thay đổi cả ngày  
và giờ
```

```
dfAlarm = new DateField("Set Alarm Time",  
                        DateField.DATE_TIME);
```

```
dfAlarm.setDate(new Date());
```

# Gauge

- ▶ Kiểu giao diện thường dùng để mô tả mức độ hoàn thành một công việc.
- ▶ Có 2 loại Gauge là loại tương tác và loại không tương tác.

```
Gauge(String label, boolean interactive, int  
maxValue, int initialValue)
```

```
private Gauge gaVolume;  
// Điều chỉnh âm lượng  
gaVolume = new Gauge("Sound Level", true,  
100, 4);
```



# String Item

- ▶ Được dùng để hiển thị một nhãn hay chuỗi văn bản.
- ▶ Người dùng không thể thay đổi nhãn hay chuỗi văn bản khi chương trình đang chạy.

`StringItem(String label, String text)`



# TextField

- ▶ Là đối tượng nhập văn bản
- ▶ Có thể chỉ định một nhãn, số ký tự tối đa được phép nhập, và loại dữ liệu được phép nhập.
- ▶ Cho phép nhập vào mật khẩu với các ký tự nhập vào sẽ được che bởi ký tự mặt nạ



# TextField

`TextField TextField(String label, String text, int maxSize, int constraints)`

constraints là thành phần xác định loại dữ liệu nào được phép nhập vào TextField.

- ▶ ANY: cho phép nhập bất kỳ ký tự nào
- ▶ EMAILADDR: chỉ cho phép nhập vào các địa chỉ email hợp lệ
- ▶ NUMERIC: chỉ cho phép nhập số
- ▶ PHONENUMBER: Chỉ cho phép nhập số điện thoại
- ▶ URL: Chỉ cho phép nhập các ký tự hợp lệ bên trong URL
- ▶ PASSWORD: che tất cả các ký tự nhập vào

# ChoiceGroup

- ▶ Cho phép người dùng chọn từ một danh sách đầu vào đã được định nghĩa trước. ChoiceGroup có 2 loại:

- multi-selection (cho phép chọn nhiều mục): nhóm này có liên quan đến các checkbox
- exclusive-selection (chỉ được chọn một mục): nhóm này liên quan đến nhóm các radio button



# Image, ImageItem

- ▶ Image được dùng để tạo ra một đối tượng hình ảnh và giữ thông tin như là chiều cao và chiều rộng, và dù ảnh có biến đổi hay không.

•Lớp ImageItem mô tả một tấm ảnh sẽ được hiển thị như thế nào, ví dụ tấm ảnh sẽ được đặt ở trung tâm, hay đặt về phía bên trái, hay bên trên của màn hình.



# Image, ImageItem

```
Form fmMain = new Form("Images");
```

```
... // Create an image
```

```
Image img = Image.createImage("/house.png");
```

```
// Append to a form
```

```
fmMain.append(new ImageItem(null, img,  
    ImageItem.LAYOUT_CENTER, null));
```

- ▶ Chú ý: PNG là loại ảnh duy nhất được hỗ trợ bởi bất kỳ thiết bị MIDP nào



# List

- ▶ Một List chứa một dãy các lựa chọn



# TextBox

- ▶ TextBox được dùng để cho phép nhập nhiều dòng.

`TextBox(String title, String text, int maxSize, int constraints)`

# Alert

- ▶ Một Alert đơn giản là một hộp thoại rất nhỏ. Có 2 loại Alert:
  - ▶ Modal: là loại hộp thoại thông báo được trình bày cho đến khi người dùng ấn nút đồng ý
  - ▶ Non-modal: là loại hộp thoại thông báo chỉ được trình bày trong một số giây nhất định

Alert(String title)

Alert(String title, String alertText, Image alertImage, AlertType alertType)



# Ticker

- ▶ Thành phần Ticker được dùng để thể hiện một đoạn chuỗi chạy theo chiều ngang.
- ▶ Tham số duy nhất của thành phần Ticker là đoạn văn bản được trình bày.
- ▶ Tốc độ và chiều cuốn được xác định bởi việc cài đặt trên thiết bị nào.

Ticker(String str);



# Canvas

- ▶ Cung cấp một khung vẽ cho phép tạo ra giao diện tùy biến người dùng.
- ▶ Được dùng để xử lý sự kiện, vẽ ảnh và chuỗi lên thiết bị hiển thị.

# Graphics

- Graphics cung cấp các công cụ thật sự để vẽ như vẽ đường thẳng, hình chữ nhật, cung, ...



# Tài liệu tham khảo

- ▶ Kim Topley, *J2ME in a Nutshell*, O'Reilly, 2002
- ▶ Roger Riggs, *Programming Wireless Devices with the Java™ 2 Platform Micro Edition Second Edition*, Addison Wesley, 2003
- ▶ <http://java.sun.com>
- ▶ <http://www.javavietnam.org>

# Q/A