من اعداد مصعب غيلان- جمهورية اليمن العربية

3-6- التعامل مع صناديق الحوار Alert في J2ME,
ضمن High Level APIs
سنتعرف في هذا الجزء على كيفية التعامل مع صناديق الرسائل او التنبيه Alert

يرث هذا الصنف عدد من الاعضاء Members من الصنف Screen ومن هذة الطرق

setTicker, setTitle

Alert : عبارة عن شاشة شبيهة الى حد ما ب Message Box تظهر للمستخدم كتنبيه على عملية معينه
عمل هذة الكائنات بسيط جدا ما عليك الا اعطاء هذة الكائنات بعض الخصائص ومن ثم استدعائها لعرضها على الشاشة

هذا الكائن يحتوي على مشيدين Constructor تستطيع اختيار احدهما

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

title : العنوان الذي سيظهر اعلى الصندوق
alertText : النص الذي سيظهر داخل التنبيه ويقرأه المستخدم
alertImage : الايقونه او الصورة الصغيرة التي ستظهر في اعلى التنبيه
ويمكن كتابة null لعدم عرض اي صورة
alertType : نوع هذا التنبيه هل هو
ALARM او CONFIRMATION او ERROR او INFO او WARNING
وكل نوع من هذة الانواع تعطي التنبية شكل معين ليظهر به.
ويتم كتابة النوع على الشكل AlertType.TYPE
حيث نقوم باستبدال الكلمة TYPE باحد الانواع السابقة

****

وبعد ان تعرفنا على بنية المشيد سنذكر الان بعض الطرق التابعة للصنف Alert وكيفية استخدامها

public void addCommand(Command cmd)

اضافة زر الى التنبيه وهذه الطريقة غير مستخدمة عادة لكن قد نضطر لاستخدامها احيانا
والسبب انك عادة لا ترغب باضافة اي زر عدا زر موافق
طبعا الزر موافق سيضاف تلقائيا الى التنبيه عند استخدام Timeout من النوع FOREVER وسنرى في الفقرات اللاحقة كيفية التعامل مع Timeout باستخدام الدالة setTimeout

public void setString(String str)

تقوم هذه الداله بتغيير النص الذي سيعرض في التنبيه

public void setTimeout(int time)

Timeout : هو الوقت الذي سيبقى فيه التنبيه معروضا على الشاشة ويقاس بالمللي ثانية
مثلا لو قمنا بانشاء كائن alert من نوع الصنف تنبيه ثم اردنا عرض هذا التنبيه على الشاشة لمدة 5 ثواني فانا سنقوم بالكتابة كالتالي

alert.setTimeout(5000);

لكن ماذا واذا اردنا ان يستمر العرض الى ان يقوم المستخدم بضغط زر موافق مثلا
هل سنزيد قيمة Timeout بالطبع لا
ولكن لعمل ذلك سنقوم بوضع القيمة الثابته FOREVER والتابعة للصنف Alert
اي سنقوم بكتابة الكود على الشكل

alert.setTimeout(Alert.FOREVER);

كما سنرى ذلك في المثال التالي

public AlertType setType()

يقوم بتغيير الشكل الذي سيظهر به التنبيه هل هو
ALARM او CONFIRMATION او ERROR او INFO او WARNING

والان بعد شرحنا كيفية التعامل مع التنبيه بقي علينا ان نقوم بوضع مثال يوضح لنا كل هذا

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class MyAlert extends MIDlet implements CommandListener
{
private Display display;

Command cmdExit = new Command("Exit", Command.EXIT, 0);
Command cmdAlert = new Command("Alert", Command.SCREEN, 0);
Command cmdAlertError = new Command("Error", Command.SCREEN, 0);
Command cmdAlertSave = new Command("Save", Command.SCREEN, 0);

Form form;

Alert alert;

public MyAlert ()
```

```
{
display =Display.getDisplay(this);
form = new Form("Form Title");
alert = new Alert (" Alert Title ");
}

public void startApp()
{
form.append("This is My Form");
form.addCommand(cmdExit);
form.addCommand(cmdAlert);
form.addCommand(cmdAlertError);
form.addCommand(cmdAlertSave);
form.setCommandListener(this);
display.setCurrent(form);
}

public void pauseApp(){ }
public void destroyApp(boolean unconditional){}

/************* implements CommandListener ***************/
public void commandAction(Command c, Displayable s)
{
if ( s == form )
{
if (c == cmdExit)
{
destroyApp(false);
notifyDestroyed();
}
else if (c == cmdAlert)
{
alert.setString("Alert with command Ignore\n _____\n Alarm Type ");
alert.setType(AlertType.ALARM);
alert.setTimeout(Alert.FOREVER);
display.setCurrent(alert);
}
else if (c == cmdAlertError)
{
alert.setString("Alert with 5 second waitting \n _____\n Error Type");
alert.setType(AlertType.ERROR);
alert.setTimeout(5000);
display.setCurrent(alert);
}
else if (c == cmdAlertSave)
{
alert.setString("Alert with 3 second waitting \n _____\n Confirmation Type");
```

```
alert.setType(AlertType.CONFIRMATION);
alert.setTimeout(3000);
display.setCurrent(alert);
}
}

} // end commandAction
} // end Class
```

شرح المثال

```
;(Command cmdAlert = new Command("Alert", Command.SCREEN, 0
Command cmdAlertSave = new Command("Save", Command.SCREEN,
;(0
Command cmdAlertError = new Command("Error", Command.SCREEN,
;(0
```

عرفنا ثلاث ازرار وسنستخدمها لاظهار الانواع المختلفة للتنبيه

Form form;

سنستخدمه من اجل اضافة الازرار اليه وعرضه فقط

Alert alert;

التنبيه الذي سنقوم بإظهاره على الشاشة مع تغيير لشكل العرض

public MyAlert () {

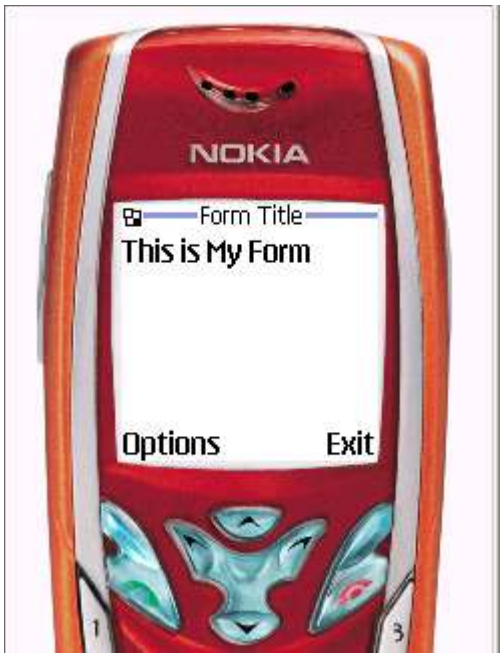
هذه المرة سنضع هذه التعليمات الابتدائية في المشيد

alert = new Alert (" Alert Title ");

هنا قمنا بإنشاء كائن من النوع تنبيه واعطيناه العنوان **Alert Title**
{

form.append("This is My Form");

اضافة نص الى الفورم وسوف ترى هذا النص ظاهرا على الفورم كما بالشكل




```
form.addCommand(cmdExit);
 form.addCommand(cmdAlert);
form.addCommand(cmdAlertError);
 form.addCommand(cmdAlertSave);
```

اضافة الازرار الى الفورم

```
else if (c == cmdAlert)
```

اذا قمنا باختيار هذا الزر

```
alert.setString("Alert with command Ignore\n _____\n Alarm Type ");
```

عرض هذا النص في كائن التنبيه **alert**

```
alert.setType(AlertType.ALARM);
```

ضبط نوع هذا التنبيه من النوع **Alarm** كما ترى في هذه الشاشة

alert.setTimeout(Alert.FOREVER);

سيبقى هذا التنبيه معروضا على الشاشة الى ان يقوم المستخدم بكبس الزر **done** او **ignore**

display.setCurrent(alert);

اخيرا بعد ان اعطينا هذه الخصائص للكائن **alert** سنقوم هنا بعرضه على الشاشة

else if (c == cmdAlertError)

اذا قمنا باختيار هذا الزر

alert.setString("Alert with 5 second waitting \n _____\n Error Type");

عرض هذا النص في كائن التنبيه **alert**

alert.setType(AlertType.ERROR);

ضبط نوع هذا التنبيه من النوع **Error** كما ترى في هذه الشاشة

وهذا النوع شبيه بتنبيه **"تم الحفظ"**

alert.setTimeout(3000);

سيبقى هذا التنبيه معروضا على الشاشة لمدة ثلاث ثوان فقط ومن ثم يختفي

display.setCurrent(alert);

اخيرا بعد ان اعطينا هذه الخصائص للكائن **alert** سنقوم هنا بعرضه على الشاشة

6-4- التعامل مع القوائم List في J2ME,
<u>ضمن High Level APIs</u>

سنتعرف في هذا الجزء على كيفية التعامل مع القوائم List

يرث هذا الصنف عدد من الاعضاء Members من الصنف Screen ومن هذة الطرق setTicker, setTitle
كما انه يستخدم الواجهة Choice على الشكل implements Choice

List : عبارة عن شاشة تحتوي على مجموعة خيارات Choices يستطيع المستخدم اختيار اي واحد منها
وهناك ثلاثة اشكال لهذة القائمة
IMPLICIT : قائمة من الازرار يستطيع المستخدم الضغط على اي زر من القائمة ويتم معرفة الزر الذي تم كبسه باستخدام الطريقة getSelectedIndex التابعة للكائن المنشأ من الصنف List
اما SELECT_COMMAND : فيعبر عن الزر اختيار select الذي يأتي مترافقا مع هذا النوع من القوائم

EXCLUSIVE : تمثل قائمة خيارات يستطيع المستخدم اختيار خيار واحد فقط من هذة القائمة وتكون قيمة الخيار الذي تم اختياره true وبقية الخيارات false وهذة القائمة شبيهة بال Option في الفيجوال البيسك ويمكن معرفة قيمة

كل خيار هل تم اختياره ام لا بواسطة الطريقة isSelected حيث نعطيها رقم الخيار في القائمة وتعيد لنا قيمة true او false اي هل تم اختياره ام لا

MULTIPLE : قائمة خيارات من النوع المتعدد وتشبه الى حد ما CheckBox في الفيجوال بيسك ويتم التعامل معها بنفس التعامل مع النوع EXCLUSIVE

لناخذ مثالا انه يوجد لدينا قائمة من النوع الاول IMPLICIT
فاننا سنعرف ان المستخدم قام بالضغط على زر معين كما في هذا المثال

```
public void commandAction (Command c, Displayable d)
{ if (d == myList) { if (List.SELECT_COMMAND == ) {
// do Something } } }
```

يجب ملاحظة ان كل هذة القوائم يتم انشائها بنفس الطريقة ولكن فقط يختلف النوع

هذا الكائن يحتوي على مشيدين Constructor تستطيع اختيار احدهما

```
List(String title, int listType)
List(String title, int listType, String[] stringElements, Image[] imageElements)
```

title : العنوان الذي سيظهر اعلى الصندوق
listType : نوع هذه القائمة وسيترتب على النوع شكل العرض
وستكتب على الشكل List.TYPE حيث ان TYPE ستمثل اما
EXCLUSIVE, IMPLICIT, MULTIPLE

stringElements : مصفوفة نصوص الاختيارات التي ستظهر كعناصر على القائمة
imageElements : مصفوفة الايقونات او الصور التي ستظهر مترافقة مع العناصر
واذا لم نرد استخدام اي ايقونه سنضع القيمة null بدلا عن المصفوفة

****

وبعد ان تعرفنا على بنية المشيد سنذكر الان بعض الطرق التابعة للصنف List وكيفية استخدامها

```
public void addCommand(Command cmd)
```

يرث الصنف List هذة الطريقة من الصنف Displayable

public int append(String stringPart, Image imagePart)

تقوم هذة الطريقة بإضافة عنصر ما الى القائمة وتعيد رقم العنصر في القائمة
stringPart : نص العنصر الذي نقوم باضافته
imagePart : الايقونة التي ستظهر مترافقة مع العنصر المضاف
واذا اردنا عدم اظهار اي ايقونة سنستخدم القيمة null

public int getSelectedIndex()

تعيد هذه الدالة رقم العنصر الذي تم اختياره حاليا

public boolean isSelected(int elementNum)

تعيد هذه الدالة قيمة true اذا كان هذا العنصر قد تم اختياره او false اذا لم يتم اختياره
elementNum : تعبر عن رقم العنصر
والان بعد شرحنا كيفية التعامل مع القائمة List بقي علينا ان نقوم بوضع مثال يوضح لنا كل هذا

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class MyList extends MIDlet implements CommandListener
{
private Display display;

Command cmdExit = new Command("Exit", Command.EXIT, 0);
Command cmdBack = new Command("Back", Command.BACK, 0);
Command cmdInfo = new Command("Info", Command.SCREEN, 0);

List mainList;
List subList;

String[] choices;

Ticker ticker;

public MyList ()
{
display =Display.getDisplay(this);
mainList = new List("Main List",List.IMPLICIT);
ticker = new Ticker (" www.arabteam2000.com ");
choices = new String[3];
choices[0] = "choice1";
choices[1] = "choice2";
```

```
choices[2] = "choice3";
}

public void startApp()
{
mainList.append("EXCLUSIVE",null);
mainList.append("MULTIPLE",null);
mainList.addCommand(cmdExit);
mainList.setTicker(ticker);
mainList.setCommandListener(this);
display.setCurrent(mainList);
}

private void whatSelected( List list )
{
String selected = new String();
for ( int i = 0; i < list.size(); i++ )
if ( list.isSelected( i ) )
selected = selected + " , " + list.getString( i );
Alert alert = new Alert("choices");
if ( selected.length() > 0 )
alert.setString(" you selected : " + selected);
else
alert.setString("you didn`t select");
display.setCurrent(alert);
}

public void pauseApp(){ }
public void destroyApp(boolean unconditional){}

/************* implements CommandListener ***************/
public void commandAction(Command c, Displayable s)
{
if ( s == mainList )
{
if (c == cmdExit)
{
destroyApp(false);
notifyDestroyed();
}
else if ( c == List.SELECT_COMMAND )
if ( mainList.getSelectedIndex() == 0 )
{
subList = new List("EXCLUSIVE List", List.EXCLUSIVE, choices, null);
subList.addCommand(cmdBack);
subList.addCommand(cmdInfo);
subList.setCommandListener(this);
```

```
display.setCurrent(subList);
}
else if (mainList.getSelectedIndex() == 1)
{
subList = new List("MULTIPLE List",List.MULTIPLE, choices, null);
subList.addCommand(cmdBack);
subList.addCommand(cmdInfo);
subList.setCommandListener(this);
display.setCurrent(subList);
}
}
else if ( c == cmdInfo )
whatSelected( (List) s );
else if ( c == cmdBack )
{
mainList.setCommandListener(this);
display.setCurrent(mainList);
}
} // end commandAction
} // end Class
```

شرح فقط الاشياء التي لم تشرح سابقا

List mainList;

mainList من نوع List وهي القائمة الرئيسية التي ستظهر عند تنفيذ البرنامج

List subList;

القائمة الثانية التي هي الاختيارات choices

String[] choices;

عبارة عن Feld يتكون من اكثر من عنصر

mainList = new List("Main List",List.IMPLICIT);

قمنا بانشاء ال Main List وشكلها IMPLICIT

choices = new String[3]; choices[0] = "choice1"; choices[1] = "choice2"; choices[2] = "choice3";

اضافة عناصر ال Feld choices

```
private void whatSelected( List list )
{
String selected = new String();
for ( int i = 0; i < list.size(); i++ )
if ( list.isSelected( i ) )
```

```
selected = selected + " , " + list.getString( i );
Alert alert = new Alert("choices");
if ( selected.length() > 0 )
alert.setString(" you selected : " + selected);
else
alert.setString("you didn`t select");
display.setCurrent(alert);
}
```

هذا ال Method لمعرفة ما تم اختياره من List

```
list.size();
```

لمعرفة عدد عناصر ال List

```
if ( mainList.getSelectedIndex() == 0 )
{
subList = new List("EXCLUSIVE List", List.EXCLUSIVE, choices, null);
subList.addCommand(cmdBack);
subList.addCommand(cmdInfo);
subList.setCommandListener(this);
display.setCurrent(subList);
}
```

اذا كان الاختيار الاول من ال mainList قمنا بانشاء subList وعناصرها choices وشكلها واضفنا الازرار Info and Back

```
if ( mainList.getSelectedIndex() == 0 )
{
subList = new List("EXCLUSIVE List", List.EXCLUSIVE, choices, null);
subList.addCommand(cmdBack);
subList.addCommand(cmdInfo);
subList.setCommandListener(this);
display.setCurrent(subList);
}
```

اذا كان الاختيار الثاني من ال mainList قمنا بانشاء subList وعناصرها choices وشكلها واضفنا الازرار Info and Back

```
if ( c == cmdInfo )
whatSelected( (List) s );
```

اذا اختير الز Info سيعود الى الاسلوب whatSelected((List) s) لمعرفة الذي تم اختياره وكتابة ذلك عن طريق Alert

```
if ( c == cmdBack )
                                                                {
mainList.setCommandListener(this);
;(display.setCurrent(mainList
                                                                {
```
اذا اختير الزر Back سيرجع ال mainList عن طريق

```
mainList.setCommandListener(this);
display.setCurrent(mainList);
```

البرنامج بعد التنفيذ