Como usar a API Soft Notification em Java ME

Visão Geral

A Java Runtime 1.3 para dispositivos S60 introduziu um novo recurso de interface gráfica, chamado*Soft Notification API*. É uma extensão da Nokia Nokia UI API 1.2, estando normalmente disponibilizada em dispositivos *S60 5th Edition* (e mais novos). *Soft notifications* informa ao usuário sobre eventos que ocorreram durante a ausência do usuário, ou enquanto o usuário está ocupado realizando alguma tarefa em uma outra aplicação. Eles aparecem no formato de *pop-ups* na tela do dispositivo (modo de visão *idle*). É possível definir o texto que será exibido e ícone que será utilizado para notificar. Diversos formatos de imagens podem ser utilizados para exibição como ícone, tais como png and svg.

Uso

Este artigo introduz a MIDlet EggTimer, que demonstra o uso da *Soft Notification API*. A classe *TimerNotification* contem os mais relevantes partes de código. Além da classe *TimerNotification*, há a classe principal MIDlet chamada EggTimer.java e a classe KeypadCanvas, que é usada para criar a UI que será exibida de tempos em tempos. Os recursos de manipulação de tempo será realizado usando as classes java.util.TimerTask and java.util.Timer.

As notificações podem ser exibidas ao usuário de duas diferentes formas, desagrupado e agrupado. Notificações desagrupadas contem uma parte da informação cada. Este tipo de notificação é usando quanto há apenas uma notificação ativa ou quando as notificações prioritário não suportam agrupamento. Note também, que as notificações são persistentes e reiniciar o dispositivo não deleta as notificações criadas.

Diferentes tipos de itens de informações podem ser combinadas em um grupo de notificações onde os itens são exibidos como umas lista. O usuário pode então selecionar um ou mais dos itens para realizar o devido tratamento. Se o usuário descartar as notificações que estão agrupadas, então todos os itens exibidos serão descartados e consequentemente não serão mais exibidos.

A *Soft Notification API* conssitem de duas classes e uma interface localizadas no pacote com.nokia.mid.ui package:

- Interface SoftNotificationListener
- Classe SoftNotification
- Classe SoftNotificationException

Esta API é um recurso opcional e portanto depende da disponibilidade no dispositivo. Para identificar em tempo de execução se o dispositivo suporta este recurso basta realizar a seguinte chamada dentro de uma MIDlet

- System.getProperty("com.nokia.mid.ui.softnotification")

Se o retorno for *true* então o dispositivo possui suporte a esta API.

O código a seguir mostra como criar notificações e como definir as propriedades delas. Finalmente é exibida ao usuário usando o método **SoftNotification.post()**.

```
private void showTimerNotification() throws SoftNotificationException {
 softNotification = SoftNotification.newInstance();
 softNotification.setListener(this);
 softNotification.setText("Note! Timer time passed: " + time + " minutes.", "");
 softNotification.setSoftkeyLabels("Ok", "Exit");
 softNotification.setImage(readResourceImage("/egg.png"));
 softNotification.post();
}
```

O usuário pode responder a uma notificação usando as teclas do dispositivos (*softkeys*). A *softkey* 1(*Left Soft key*) é usada para ativar a função, por exemplo, abrir uma mensagem recebida. A aplicação que lançou a notificação será então lançada para primeiro plano quando a notificação for aceita pelo usuário. A *softkey* 2 (*RightSoft key*) é usada para descartar a notificação sem tomar qualquer ação adicional. Quando a *softkey* 1 é selecionada, o método **SoftNotificationListener.notificationSelected()** é chamado. Por outro lado, se o usuário selecionar a opção de descartar a notificação. será invocado o método **SoftNotificationListener.notificationDismissed()**.

Abaixo é apresentado um trecho de código que implementa esses dois métodos. Em ambos os casos o alarme é parado. No

(C) Copyright Nokia 2014. All rights reserved.

entanto, se a tecla *softkey* 1 for selecionada, a MIDlet EggTimer será automaticamente trazida para primeiro plano de execução. Caso seja selecionada a *softkey* 2 a MIDlet será encerrada.

```
public void notificationSelected(SoftNotification notification) {
 try {
  midlet.player.stop();
  midlet.player.close();
  midlet.player = null;
  time = 0;
 } catch (MediaException me) {
  midlet.showError("MediaException", me.getMessage());
 }
}

public void notificationDismissed(SoftNotification notification) {
 try {
  midlet.player.stop();
  midlet.player.close();
  midlet.player = null;
  time = 0;
 } catch (MediaException me) {
  midlet.showError("MediaException", me.getMessage());
 }
 midlet.notifyDestroyed();
}
```

Identificando as teclas pressionadas usando *scancodes*

Além de usar o *touch scree* é possível definir o tempo de alarme na MIDlet EggTimer usando as teclas numéricas. Por exemplo, em um dispositivo N97 as teclas numéricas estão localizadas na linha superior. Normalmente, as teclas de eventos para números pode ser capturasas usando as teclas juntamente com a tecla FN pressionada. Por exemplo, pressionando FN + Q seria equivalente a seleção da tecla *1*. Uma outra opção é usar *scan codes*, como mostrado abaixo:

```
protected void keyPressed(int keyCode) {
 int selected = 0;  // the selected number key
 keyScanCode = 0; // the scan code of the selected key
 try {
  keyScanCode = Integer.parseInt(System.getProperty("com.nokia.key.scancode"));
 }
 catch (NumberFormatException nfe) {
  midlet.showError("NumberFormatException", nfe.getMessage());
 }
 if (keyScanCode == 81) selected = 1;
 else if (keyScanCode == 87) selected = 2;
 else if (keyScanCode == 69) selected = 3;
 else if (keyScanCode == 82) selected = 4;
 else if (keyScanCode == 84) selected = 5;
 else if (keyScanCode == 89) selected = 6;
 else if (keyScanCode == 85) selected = 7;
 else if (keyScanCode == 73) selected = 8;
 else if (keyScanCode == 79) selected = 9;
 else if (keyScanCode == 80) selected = 0;
}
```


(C) Copyright Nokia 2014. All rights reserved.

Lendo dados de imagens para um array de bytes

Como mostrado no código abaixo, é possível definir a imagem que será utilizada apra notificação. Imagens sofreram escala de forma a se adequarem ao tamanho do *pop-up* das noticidações. Para evitar esse problema, é recomendável que as imagens escolhidas tenha um tanho pequeno (de preferência menor que 300x300 px). Para definir a imagem, usa-se o método '*setImage(byte[])*, onde o parametro passado é um array de bytes da imagem. se a imagem for incluída no jar da aplicação, então a imagem poderá ser carregada usando o método abaixo:

```
private byte[] readResourceImage(String resource) {
 byte[] byteData = null; // resource data
 try {
  InputStream is = getClass().getResourceAsStream(resource);
  if(is == null){ }
  else {
   DataInputStream dataStream = new DataInputStream(is);
   int byteRead = 0;
   ByteArrayOutputStream baos = new ByteArrayOutputStream();
   while (byteRead != -1){
    byteRead = dataStream.read();
    baos.write(byteRead);
   }
   byteData = baos.toByteArray();
   baos.close();
   dataStream.close();
  }
 }
 catch (IOException ioe) {
  midlet.showError("IOException", ioe.getMessage());
 }
 return byteData;
}
```

Usando as classes *Timer* e *TimerTask* para alertar

A definição do tempo para alerta é dado em minutos. Este tempo é usado no método **TimerNotification.startTimer()**, onde instâncias da classe *Timer* e *EggTimerTask* são criadas. O método *schedule* é usado para agendar a tarefa a ser executada após o período definido.

```
protected void startTimer() {
 timer = new Timer();
 task = new EggTimerTask();
 long millisecs = time*60*1000;
 timer.schedule(task, millisecs);
}
```

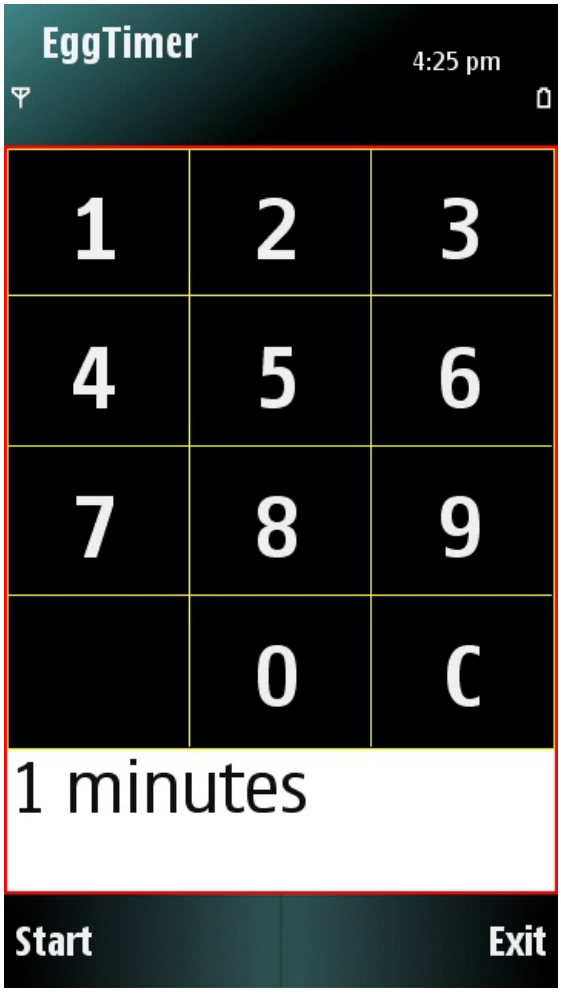
Finalizado o período de delay o método **EggTimerTask.run()** será invocado. A notificação de tempo é então exibida na tela e um som é tocado repetidamente, caso a tela do dispositivo não estiver exibindo nenhum aplicação. Caso contrário apenas um único toque é executado.

```
public final void run() {
 try {
  showTimerNotification();
 } catch (SoftNotificationException sne) {
```


(C) Copyright Nokia 2014. All rights reserved.

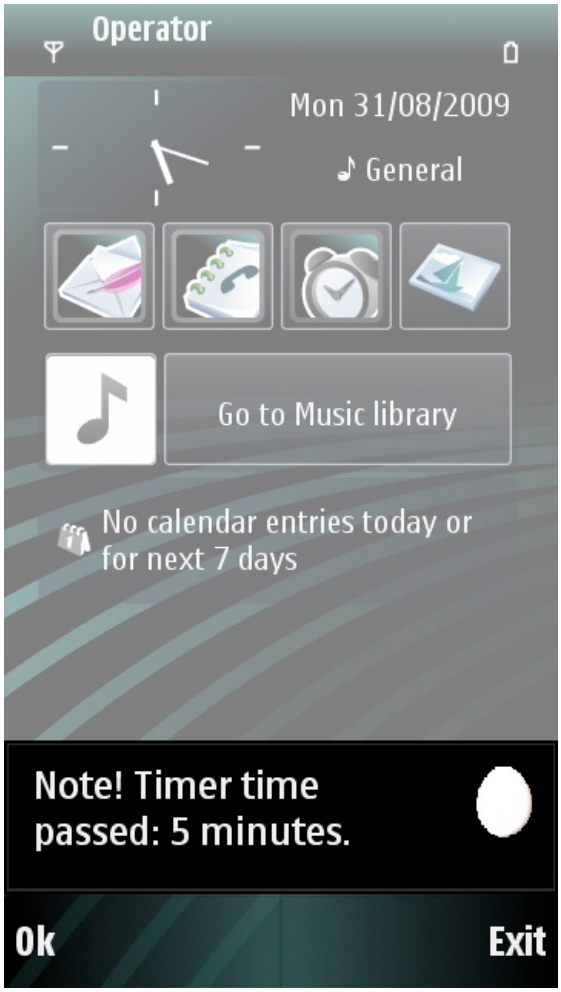
```
    midlet.showError("SoftNotificationException", sne.getMessage());
  }
  playTimerTone();
 }
```

A imagem abaixo mostra a MIDlet EggTimer sendo executado na SDK do Nokia N97.



A imagem abaixo mostra a notificação exibida pela MIDlet EggTimer.

(C) Copyright Nokia 2014. All rights reserved.



Aplicação exemplo

- EggTimer.zip contendo o código fonte, e os arquivos EggTimer.jad and EggTimer.jar.

Veja também

- SoftNotification class javadoc in Nokia Developer Java Developer's Library
- Nokia UI API's UI extensions

(C) Copyright Nokia 2014. All rights reserved.