# Porting MIDlets from Nokia Asha to Symbian

Symbian devices, due to being on the higher end of smart phone devices, might allow the installation of MIDlets originally written for Nokia Asha software platform, without any code modifications. This article provides guidelines on the most common porting issues developers should be aware of when porting from Nokia Asha software platform to Symbian devices.

## Orientation

On Symbian devices, high-level UI elements are scaled automatically based on screen resolution and device orientation. If a MIDlet sets the preferred sizes for Form items, these items may need to be scaled manually for different resolutions. The device platform sends a notification (by calling the `Displayable.sizeChanged` method) when the draw-able area changes. Low-level UI elements require you to implement any scaling functionality manually. The `OrientationListener` is an interface, only supported on Series 40 and Nokia Asha platforms. You would therefore need to entirely remove it from your code, in order to be able to compile and run your MIDlet on Symbian. These are typically the lines in your code that you need to remove:

```
import com.nokia.mid.ui.orientation.Orientation;
import com.nokia.mid.ui.orientation.OrientationListener;
…
  protected void startApp() {
    Orientation.addOrientationListener(this);
  }
  ...
  public class YourMIDlet extends MIDlet implements OrientationListener{
    /** Orientation listener callback */
    public void displayOrientationChanged( int newDisplayOrientation ){

    /** Check display orientation */
    switch( newDisplayOrientation ){
      case Orientation.ORIENTATION_PORTRAIT:
      case Orientation.ORIENTATION_PORTRAIT_180:
        /** Change MIDlet UI orientation to portrait */
        Orientation.setAppOrientation(Orientation.ORIENTATION_PORTRAIT);
        break;

      case Orientation.ORIENTATION_LANDSCAPE:
      case Orientation.ORIENTATION_LANDSCAPE_180:
        /** Change MIDlet UI orientation to landscape */
        Orientation.setAppOrientation(Orientation.ORIENTATION_LANDSCAPE);
        break;
    }
  }
}
```

## Hiding the Virtual Keyboard for Canvas based applications

Similarly to Nokia Asha software platform, Symbian devices, depending on the platform version, might display by default an On-Screen Keypad (OSK) for `Canvas` based application. On Series 40 and Asha software platform, the Virtual Keyboard is displayed by default in the Option's Menu. The code for hiding the Virtual Keyboard cannot be used for hiding the On-Screen Keypad on Symbian. You will therefore need to remove the following code from your MIDlet:

```
String keyboardType = System.getProperty("com.nokia.keyboard.type");
if (keyboardType.equals("OnekeyBack") || keyboardType.equals("None")) {
    // Series 40 full touch or Nokia Asha software platform detected
```


(C) Copyright Nokia 2014. All rights reserved.

```
    // Your code here
    com.nokia.mid.ui.VirtualKeyboard.hideOpenKeypadCommand(true);
}
```

By default, the on-screen keypad behaves as follows:

- On S60 5th Edition devices, the on-screen keypad is enabled by default for every `Canvas`-based MIDlet. To disable the on-screen keypad for a MIDlet, use the *Nokia-MIDlet-On-Screen-Keypad* attribute with value `no` as follows:

```
Nokia-MIDlet-On-Screen-Keypad: no
```

- From Symbian^3 onwards, the Java Runtime automatically detects the need for the on-screen keypad when a `Canvas`-based MIDlet is installed, and disables or enables the keypad accordingly for the MIDlet. During installation, the Java Runtime checks if the MIDlet uses any classes that are inherited from `javax.microedition.lcdui.Canvas` and whether those classes implement any of the following methods:
  1. `protected void pointerDragged(int x, int y)`
  2. `protected void pointerPressed(int x, int y)`
  3. `protected void pointerReleased(int x, int y)`

If the MIDlet contains at least one `Canvas`-based class that implements at least one of the above methods, the MIDlet is considered touch-compatible and the on-screen keypad is disabled. If the MIDlet contains `Canvas`-based classes, but does not implement the above methods, or provides empty implementations of them, the MIDlet is considered touch-incompatible and the on-screen keypad is enabled.

If you define a valid value for the `Nokia-MIDlet-On-Screen-Keypad` JAD attribute, the Java Runtime skips automatic detection disables the on-screen keypad as per the attribute value. It is recommended to use this attribute to enforce the hiding of the OSK to older Symbian platforms.

## Porting CategoryBar MIDlets to Symbian

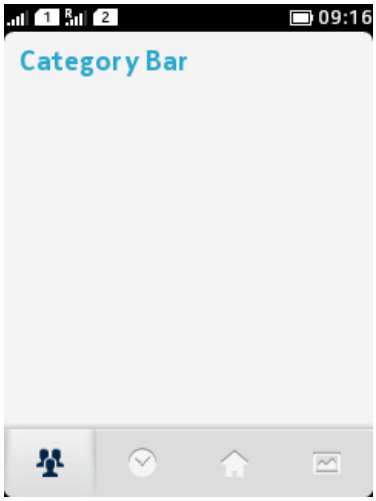
The `CategoryBar` component is only available on Series 40 full touch and Asha software platform devices. When porting to Symbian, you need to modify your UI in order to make it Symbian compatible. This typically involves initiating your different views from Commands. As an example, let us assume that we have the following 4-element `CategoryBar` MIDlet:

```
public class CategoryBarSimple
 extends MIDlet
 implements ElementListener {

 Form mainForm;
 CategoryBar bar;
 Image[] unselectedIcons;
 Image[] selectedIcons;
 Command okCmd = new Command("OK", Command.OK, 1);
 Command cancelCmd = new Command("CANCEL", Command.CANCEL, 2);

 protected void destroyApp(boolean arg0) throws MIDletStateChangeException {
  // TODO Auto-generated method stub
 }

 protected void pauseApp() {
  // TODO Auto-generated method stub
 }

 protected void startApp() throws MIDletStateChangeException {
  mainForm = new Form("Category Bar");
```


(C) Copyright Nokia 2014. All rights reserved.

```
  Display.getDisplay(this).setCurrent(mainForm);
  try {

   unselectedIcons = new Image[6];

   InputStream ios1 = this.getClass().getResourceAsStream("qgn_catbar_groups.png");
   InputStream ios2 = this.getClass().getResourceAsStream("qgn_catbar_history.png");
   InputStream ios3 = this.getClass().getResourceAsStream("qgn_catbar_home.png");
   InputStream ios4 = this.getClass().getResourceAsStream("qgn_catbar_images.png");

   unselectedIcons[0] = Image.createImage(ios1);
   unselectedIcons[1] = Image.createImage(ios2);
   unselectedIcons[2] = Image.createImage(ios3);
   unselectedIcons[3] = Image.createImage(ios4);

  } catch (IOException e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
  }

  String[] labels = { "Groups", "History", "Home", "Images"};
  bar = new CategoryBar(unselectedIcons, null, labels);

  bar.setElementListener(this);
  bar.setVisibility(true);

 }

 public void notifyElementSelected(CategoryBar bar, int selectedIndex) {
  //To be implemented
 }
}
```

This is how the MIDlet looks on Asha software platform, provided that we have added the appropriate .png files into the resource folder:



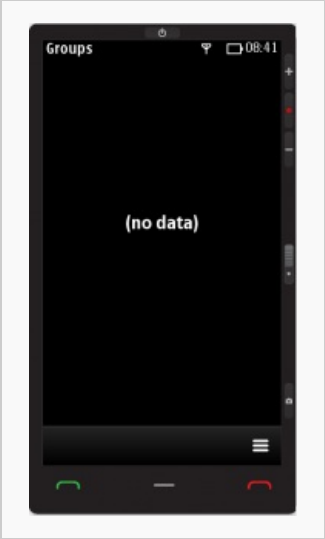

The four categories here are **Group**, **History**, **Home** and **Images**. Given that there is no identical horizontal, parallel view component on Symbian, we need to include the contents of each view in a separate Form instance and then allow the user to switch among the views with `Command`s as follows:

```
public class CategoryBarPortedSymbian
    extends MIDlet
    implements CommandListener {
```

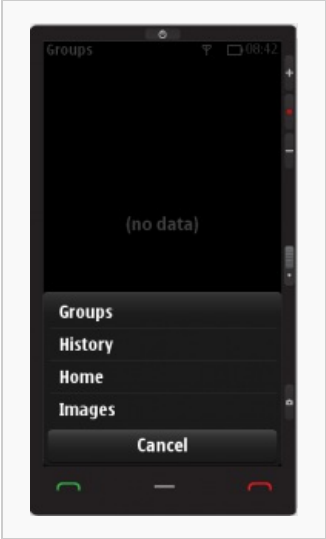

(C) Copyright Nokia 2014. All rights reserved.

```
    Command Groups = new Command("Groups", Command.OK, 1);
    Command History = new Command("History", Command.OK, 2);
    Command Home = new Command("Home", Command.OK, 3);
    Command Images = new Command("Images", Command.OK, 4);

    Form groupForm;
    Form historyForm;
    Form homeForm;
    Form imagesForm;
    Display display;

    protected void destroyApp(boolean unconditional)
            throws MIDletStateChangeException {
        // TODO Auto-generated method stub
    }

    protected void pauseApp() {
        // TODO Auto-generated method stub
    }

    protected void startApp() throws MIDletStateChangeException {
        Form groupForm = new Form("Groups");  // default view
        Form historyForm = new Form("History");
        Form homeForm = new Form("Home");
        Form imagesForm = new Form("Images");

        groupForm.addCommand(Groups);
        groupForm.addCommand(History);
        groupForm.addCommand(Home);
        groupForm.addCommand(Images);


        display = Display.getDisplay(this);
        display.setCurrent(groupForm);
    }

    public void commandAction(Command c, Displayable d) {
        //handle here the change of views
    }
}
```


(C) Copyright Nokia 2014. All rights reserved.




The Group (default) view




Switching among views is performed with Commands

## Persistent Storage and retrieval of Record Store entries on Symbian

There is a significant difference between the Asha software platform and Symbian when it comes to retrieving persistent storage entries. When a MIDlet stores data to the Persistent Storage (RecordStore), there is an index or `RecordStoreId` associated with the entry. On both Series 40 and Asha software platforms, retrieving the entries of a `RecordStore` with the `nextRecordId()` method is performed with the reversed order in which the entries were added (i.e. last one added retrieved first). On Symbian devices however, the entry order is actually followed upon retrieval (i.e. last one added retrieved last). If the retrieval order matters, we need to always ensure that the `RecordStore` is traversed in the same order in both Nokia Asha platform and Symbian. This can be done as follows:

```
RecordStore rs = RecordStore.openRecordStore("test",
true);
RecordEnumeration en;
en = rs.enumerateRecords(null, null, false);
en.reset();
int index = en.nextRecordId();
//Series 40
if(index > 1) {
  for(int i = 1; i <=index; i++) {
    rs.getRecord(i);
  }
}
//Symbian
else {
  while (en.hasNextElement()) {
    int RecId = en.nextRecordId();
    rs.getRecord(RecId);
  }
}
```

## Porting Location aware MIDlets to Symbian

The Nokia Asha platform, like Symbian, supports JSR-179 (the Location API). On Symbian devices, the retrieval of the Location Provider is done based on defining Criteria that among others include a desired horizontal and vertical accuracy or the request to also retrieve speed, besides coordinates. Criteria-based location retrieval is also supported on Asha software platform provided that an external Bluetooth GPS module is used. For applications that do not require precise and on the fly location updates, the current (approximate) location is performed by utilizing Cell ID information in Online or WLAN modes. Symbian devices do not


(C) Copyright Nokia 2014. All rights reserved.

support location retrieval based on Cell ID, so when porting Cell ID enabled MIDlets to Symbian, you need to modify the way you retrieve the location provider.

This is a sample snippet on how Cell ID based location retrieval is performed on Nokia Asha platform:

```
//Specify the retrieval method to Online/Cell-ID
int[] methods = {(Location.MTA_ASSISTED | Location.MTE_CELLID | Location.MTE_SHORTRANGE
| Location.MTY_NETWORKBASED)};
// Retrieve the location provider
LocationProvider provider = LocationUtil.getLocationProvider(methods, null);
Location location = provider.getLocation(50000);
QualifiedCoordinates coordinates = location.getQualifiedCoordinates();
```

The closest equivalent for Symbian, is an Assistant GPS (A-GPS) location retrieval, where the device will use the data connection (Internet) to contact an assistance server, by allowing costs to occur, i.e. by calling the method `setCostAllowed(true)`. This is a sample snippet on how the `QualifiedCoordinates` instance can be retrieved on Symbian:

```
public class LocationMIDlet
  extends MIDlet
  implements LocationListener {
  …
  Criteria criteria = new Criteria();
  criteria.setSpeedAndCourseRequired(false)
  criteria.setCostAllowed(true)
  criteria.setPreferredPowerConsumption(Criteria.POWER_USAGE_LOW)
  LocationProvider lp = LocationProvider.getInstance(criteria);
  lp.setLocationListener(this, -1, -1, -1);
  …
  public void locationUpdated(LocationProvider provider, Location location) {
    if(location.isValid()) {
      QualifiedCoordinates c = location.getQualifiedCoordinates();
    }
  }
}
```


(C) Copyright Nokia 2014. All rights reserved.