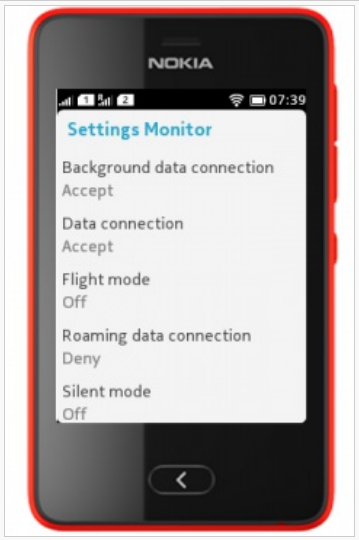
Phone Setting API on the Asha software platform

This article explores the features offered by the Phone Setting API, introduced on the Asha software platform 1.0.

22 Sep 2013

Introduction

The Phone Setting API allows Java apps to retrieve phone setting values, and to be notified when any of those values change.



Device settings retrieved through the Phone Setting API and displayed in a Java app

The Phone Setting API

The Phone Setting API is built upon two objects:

- the Setting class: allows to directly retrieve setting values, and to subscribe to changes
- the SettingListener interface: is responsible for receiving notifications of setting value changes

Available settings

The `Setting` defines the available settings through a series of constants:

- SETTING_BACKGROUND_DATA_CONNECTION: background data connection
- SETTING_DATA_CONNECTION: permission to perform data connections
- SETTING_FLIGHT_MODE: flight mode status
- SETTING_ROAMING_DATA_CONNECTION: permission to perform roaming data connections
- SETTING_SILENT: silent mode status
- SETTING_VIBRATOR: phone vibration status

By using those values, a Java app can both request the setting value, or subscribe to notification changes.

Setting values

Phone settings values can range through a definite set of values, that are similarly defined through a series of constant values in the `Setting` class.

Possible values for settings related to data connections are:

- ACCEPT: network connections are automatically allowed
- ASK: ask before connecting to network
- DENY: network connections are not allowed
- WIFIONLY: value specific for background data connections, allowing connections to be performed only through Wi-Fi networks

Allowed values for vibration, flight mode and silent mode settings are:

- OFF: feature specified by the setting is inactive
- ON: feature specified by the setting is active

A further constant is defined, to identify invalid setting values:

- INVALID: the setting has an invalid value

(C) Copyright Nokia 2013. All rights reserved.

Retrieving setting values

Setting values can be retrieved via the Setting.getSetting static method, by passing one of the constant values described above, corresponding to the desired phone setting.

The following code snippet shows how the values of all the available settings can be retrieved by a Java app:

```
int backgroundDataSettingValue = Setting.getSetting(Setting.SETTING_BACKGROUND_DATA_CONNECTION);
int dataConnectionSettingValue = Setting.getSetting(Setting.SETTING_DATA_CONNECTION);
int flightModeSettingValue = Setting.getSetting(Setting.SETTING_FLIGHT_MODE);
int roamingSettingValue = Setting.getSetting(Setting.SETTING_ROAMING_DATA_CONNECTION);
int silentSettingValue = Setting.getSetting(Setting.SETTING_SILENT);
int vibratorSettingValue = Setting.getSetting(Setting.SETTING_VIBRATOR);
```

Displaying setting values

A utility function can be defined to convert those values from numeric to textual as follows:

```
private String settingDescription(int value)
{
 switch(value)
 {
 case Setting.ACCEPT:
  return "Accept";
 case Setting.ASK:
  return "Ask";
 case Setting.DENY:
  return "Deny";
 case Setting.INVALID:
  return "Invalid";
 case Setting.OFF:
  return "Off";
 case Setting.ON:
  return "On";
 case Setting.WIFIONLY:
  return "Wi-Fi only";
 }
 return "Unknown";
}
```

By using method illustrated above, a simple LCDUI Form could easily display all the settings as follows:

```
StringItem backgroundDataItem = new StringItem("Background data connection", settingDescription(backgroundDataSettingValue));
StringItem dataConnectionItem = new StringItem("Data connection", settingDescription(dataConnectionSettingValue));
StringItem flightModeItem = new StringItem("Flight mode", settingDescription(flightModeSettingValue));
StringItem roamingItem = new StringItem("Roaming data connection", settingDescription(roamingSettingValue));
StringItem silentItem = new StringItem("Silent mode", settingDescription(silentSettingValue));
StringItem vibratorItem = new StringItem("Vibration", settingDescription(vibratorSettingValue));

myList.append(backgroundDataItem);
myList.append(dataConnectionItem);
myList.append(flightModeItem);
myList.append(roamingItem);
myList.append(silentItem);
myList.append(vibratorItem);
```

Subscribing to setting changes

A Java app can subscribe to receive notifications of setting value changes by implementing the SettingListener interface, that defines a single method responsible for receiving such notifications: settingChanged.

This method has two arguments:

1. an *integer value* identifying the setting whose value has changed
2. an *integer value* representing the new setting value

Once implemented, a `SettingListener` must subscribe to setting changes via the Setting.subscribeListener static method. Similarly, once it's not needed anymore, the `SettingListener` must unsubscribe via Setting.unSubscribeListener.

The following sample implementation of the `settingChanged` method updates the values displayed as illustrated in the previous section with the new values, by retrieving the correct `StringItem`, and updating its text:

```
private StringItem settingItem(int setting)
{
 switch(setting)
 {
 case Setting.SETTING_BACKGROUND_DATA_CONNECTION:
  return backgroundDataItem;
 case Setting.SETTING_DATA_CONNECTION:
  return dataConnectionItem;
 case Setting.SETTING_FLIGHT_MODE:
  return flightModeItem;
 case Setting.SETTING_ROAMING_DATA_CONNECTION:
  return roamingItem;
 case Setting.SETTING_SILENT:
  return silentItem;
 case Setting.SETTING_VIBRATOR:
  return vibratorItem;
 }
 return null;
}
public void settingChanged(int setting, int value)
{
 StringItem settingItem = settingItem(setting);
```


(C) Copyright Nokia 2013. All rights reserved.

```
  settingItem.setText(settingDescription(value));
 }
```

Testing different setting values

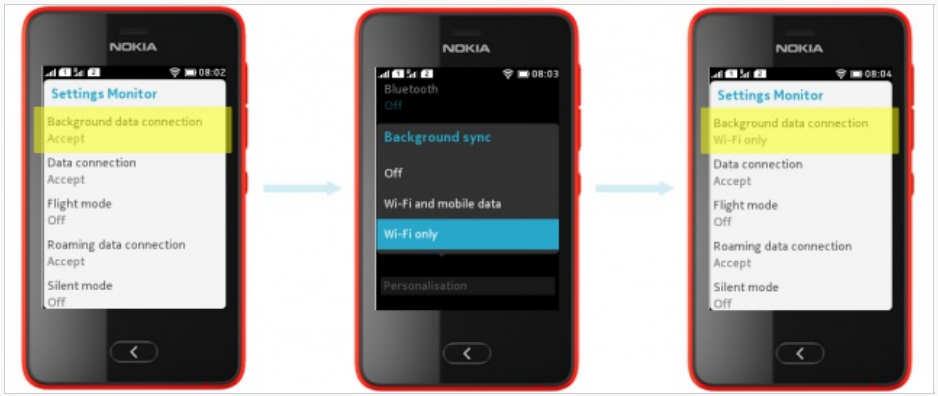
When developing a Java app that relies on any of the settings exposed by the *Phone Setting API*, it is recommended to test how the app behaves with all the possible values of the specific settings.

This section shows how the value of the available settings can be changed on the Asha software platform, and how the corresponding values retrieved and notified by the *Phone Setting API* change accordingly.

- The data connection setting can be easily changed by using the sliding top menu available on the Nokia Asha 501

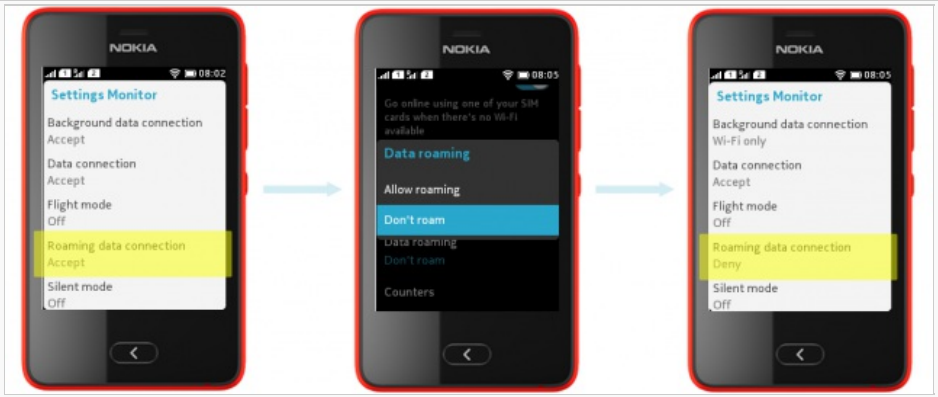
Changing the data connection setting

- The background data connection setting can be changed by opening *Settings -> Background sync*



Changing the background data connection setting

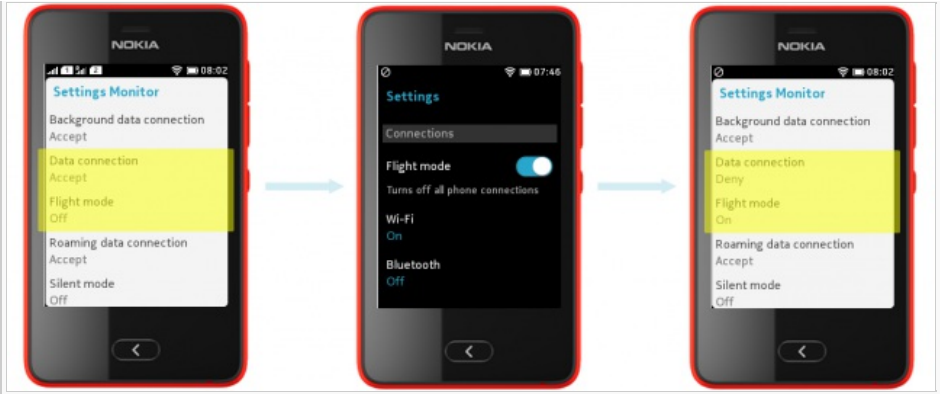
- The roaming data connection setting can be changed by opening *Settings -> Dual SIM -> Data roaming*



Changing the roaming data setting

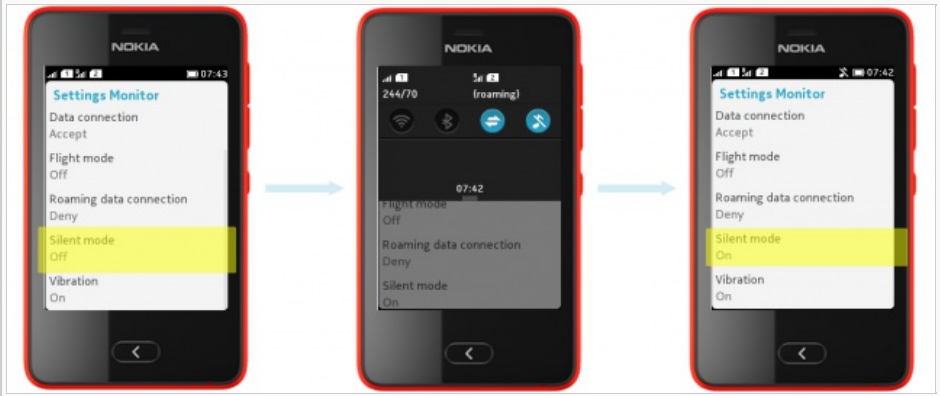
- Flight mode can be turned on and off by using the *Settings -> Flight mode* option. Note that turning on flight mode also changes the *Data connection* settings, setting its value to *Deny*

(C) Copyright Nokia 2013. All rights reserved.



Turning on flight mode

- Silent mode status can be switched through the sliding top menu, as shown by the picture below



Turning on silent mode

- Vibration can be turned on and off by opening the *Sounds and vibra* screen within the device *Settings*

Turning on vibration

Summary

This article illustrates the features offered by the *Phone Setting API*, explaining how a Java app can easily access device settings on the Asha software platform.

More details about the API is available in its JavaDocs pages.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.