Bab 11
Topik-topik Tambahan

11.1 Tujuan

Setelah menyelesaikan bab ini, siswa diharapkan mampu:

- mengatur jadwal tugas menggunakan Timers
- meregister koneksi yang datang pada Push Registry

11.2 Timers

Timer dan TimerTasks berfungsi agar Anda bisa melakukan penjadwalan tugas pada suatu waktu. Tugas dapat juga dijadwalkan untuk diulang-ulang sampai interval tertentu.

Anda dapat membuat tugas dengan menurunkan (extending) TimerTask dan mengimplement method run(). Method run() akan dieksekusi berdasarkan jadwal yang ada pada Timer.

```
class CounterTask extends TimerTask {
    int counter = 0;
    public void run() {
        System.out.println("Counter: " + counter++);
    }
}
```

Untuk menjadwalkan sebuah tugas, buat sebuah Timer dan gunakan method schedule() yang ada pada Timer untuk menjadwalkan jalannya tugas. Setiap Timer berjalan pada bagian yang terpisah. Method schedule() memiliki beberapa bentuk. Anda dapat mengatur waktu tugas untuk mulai dengan memberikan delay dalam mililiseconds atau dengan memberikan waktu absolut (java.util.Date). Parameter ketiga pada method schedule() adalah periode pengulangan dari tugas. Jika nilai pengulangan diberikan, tugas akan dieksekusi dalam periode waktu tertentu.

```
    Timer timer = new Timer();
    TimerTask task = new CounterTask();
```

```
    // task akan dimulai dalam 8 detik dan diulangi setiap 1 detik
    timer.schedule(task, 8000, 1000);
```

Anda dapat menghentikan timer dengan menggunakan method close(). Method ini dapat menghentikan timer dan mengabaikan tugas yang dijadwalkan. Perlu Anda catat, bahwa ketika Timer dihentikan, maka tidak dapat diulangi (direstart) kembali.

void	schedule(TimerTask task, Long delay) Melakukan penjadwalan tugas untuk dieksekusi sesudah menentukan delay yang diinginkan (dalam milliseconds)
void	schedule(TimerTask task, Long delay , long period) Melakukan penjadwalan tugas untuk dieksekusi berulang-ulang, dimulai sesudah delay yang ditentukan (dalam milliseconds)
void	schedule(TimerTask task, Date time) Melakukan penjadwalan tugas agar dapat dieksekusi pada waktu yang ditentukan.
void	schedule(TimerTask task, Date time, long period) Melakukan penjadwalan tugas untuk dieksekusi berulang-ulang, dimulai pada waktu yang ditentukan.
void	cancel() Menghentikan timer, mengabaikan tugas yang dijadwalkan.

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

import java.io.*;
import java.util.Timer;
import java.util.TimerTask;
import java.util.Date;

public class TimerMidlet extends MIDlet implements CommandListener{
    private Command exitCommand;
    private Form form;
    private StringItem textField;
    private Display display;
```

```
    public TimerMidlet() {
        exitCommand = new Command("Exit", Command.EXIT, 1);
        textField = new StringItem("Counter", "");

        Timer timer = new Timer();
        TimerTask task = new CounterTask(this);
        timer.schedule(task, 2000, 1000);

        form = new Form("Timer Test");
        form.addCommand(exitCommand);
        form.append(textField);
    }

    public void startApp() {
        display = Display.getDisplay(this);
        form.setCommandListener(this);
        display.setCurrent(form);
    }

    public void pauseApp() {}
    public void destroyApp(boolean unconditional) {
        timer.cancel();
    }

    public void commAndaction(Command c, Displayable d) {
        if (c == exitCommand) {
            destroyApp(true);
            notifyDestroyed();
        }
    }

    public void setText(String text){
        textField.setText(text);
    }
}

class CounterTask extends TimerTask {
```

```
    int counter = 0;
    TimerMidlet midlet;

    public CounterTask(TimerMidlet midlet){
        this.midlet = midlet;
    }

    public void run() {
        counter++;
        midlet.setText("" + counter);
        System.out.println("Counter: " + counter);
    }
}
```

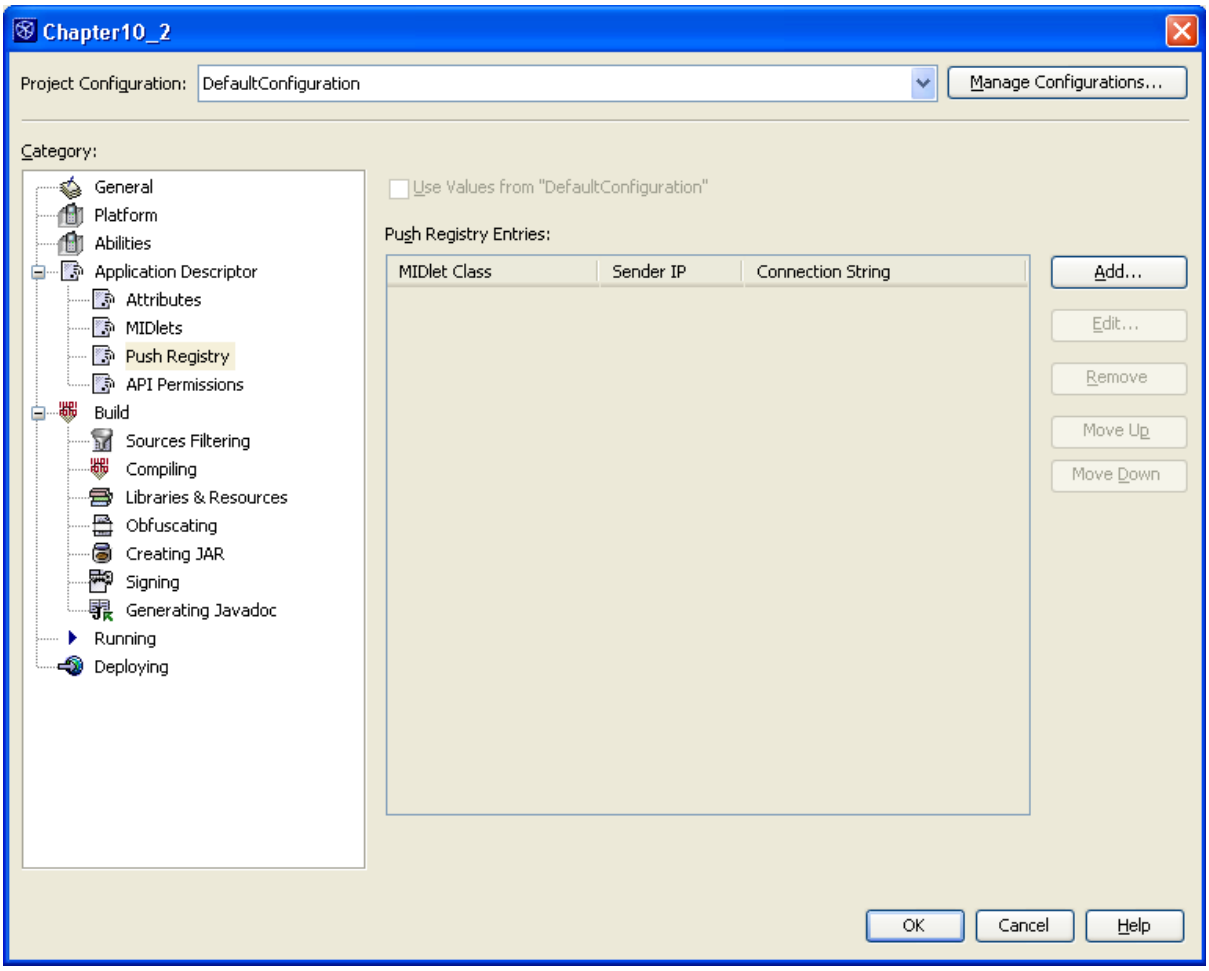
11.3 Push Functionality

Push Registry berfungsi agar MIDlet bisa meregister koneksi yang masuk dengan Application Management Software (AMS). Jika program tidak berjalan, AMS akan mendengarkan koneksi pada alamat yang telah diregister oleh aplikasi. Hampir semua tipe koneksi didukung, termasuk ServerSocket dan MessageConnection.

Anda dapat meregister koneksi yang masuk dengan Push Registry menggunakan dua cara: cara statis dengan menggunakan file application descriptor (JAD) atau dinamis selama proses runtime menggunakan PushRegistry API.

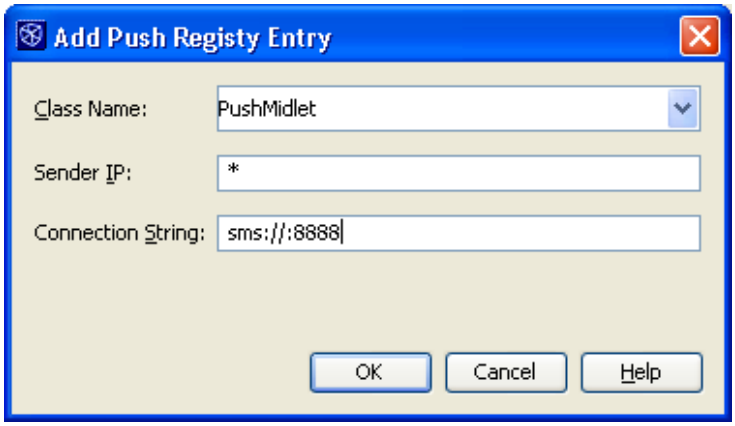
Pada bab ini kita akan meregister secara statis push application kita pada application descriptor (JAD) kita. NetBeans Mobility Pack membantu kita untuk memodifikasi Application Descriptor dengan mudah termasuk pada Push Registry.

Klik kanan pada Project name, selanjutnya klik Properties untuk membuka Properties Page pada project.

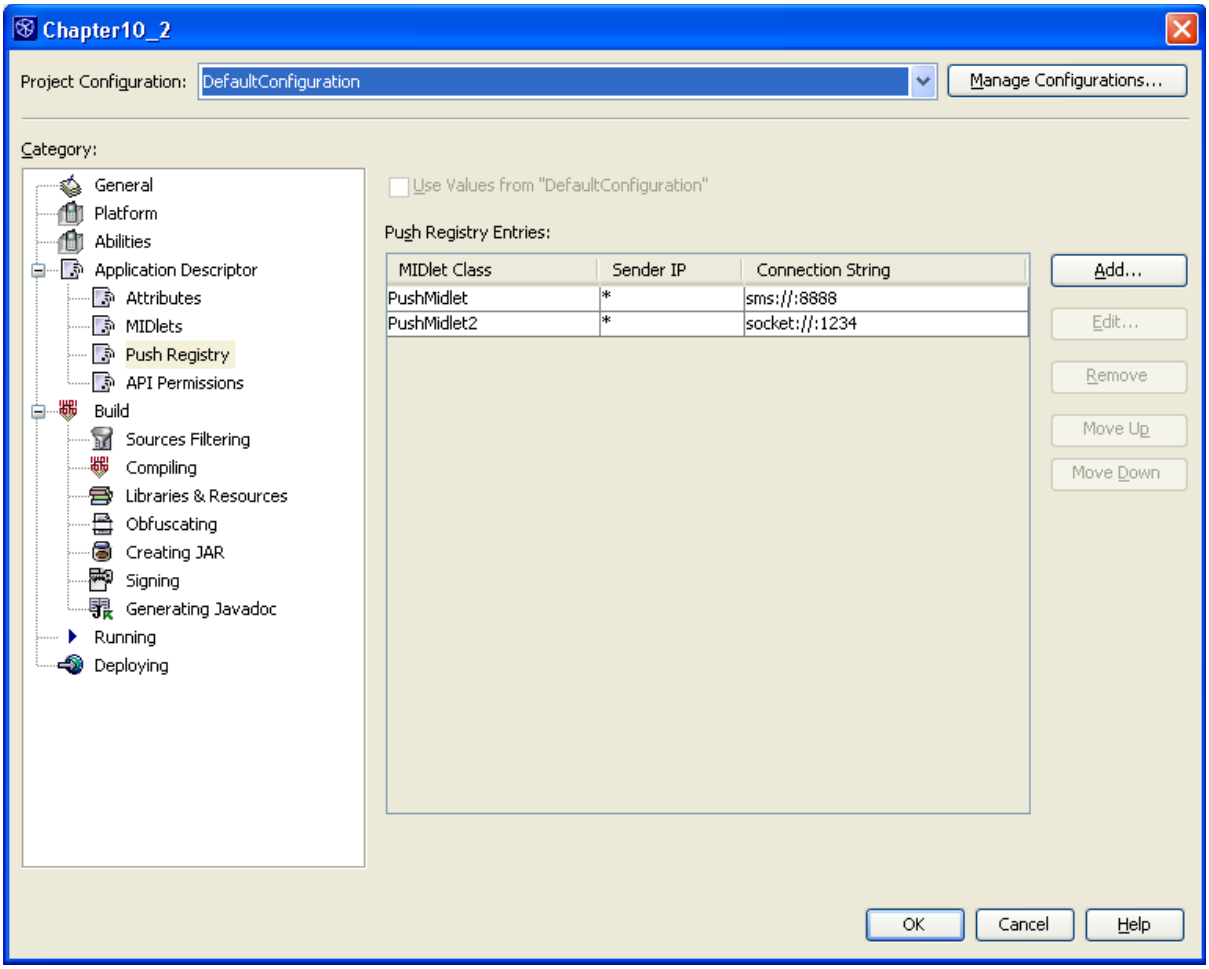


Pilih bagian Push Registry:

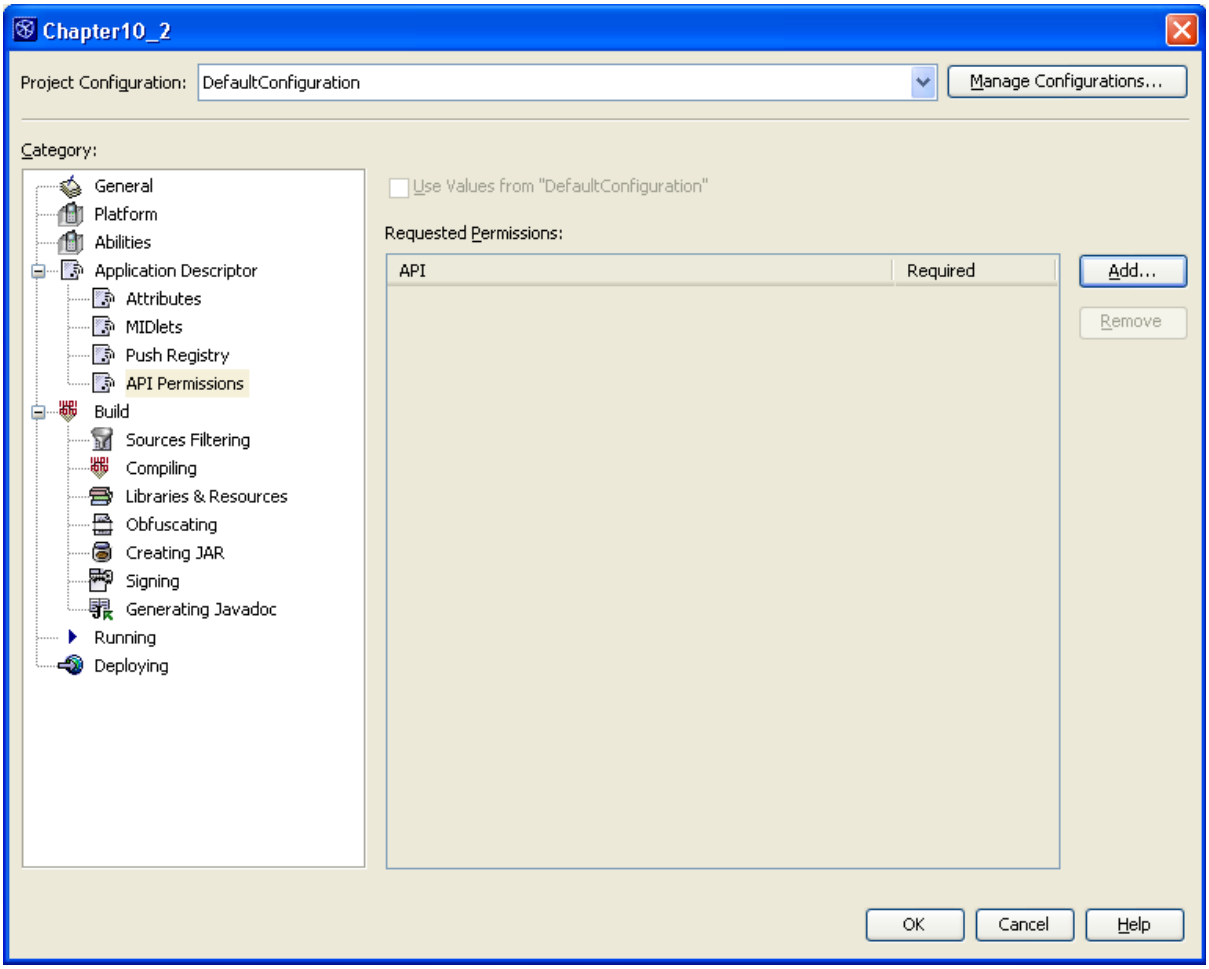
Klik "Add" untuk meregister koneksi yang masuk:



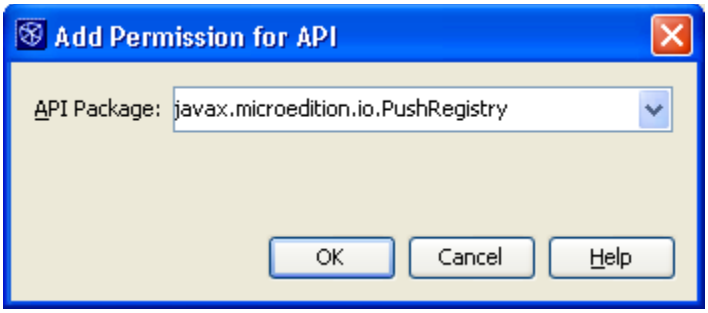
Ulangi proses sebelumnya hingga semua koneksi yang masuk sudah teregister. Dalam kasus ini, kita melakukan koneksi sms pada port 8888 dan koneksi socket (socket connection) pada port 1234:



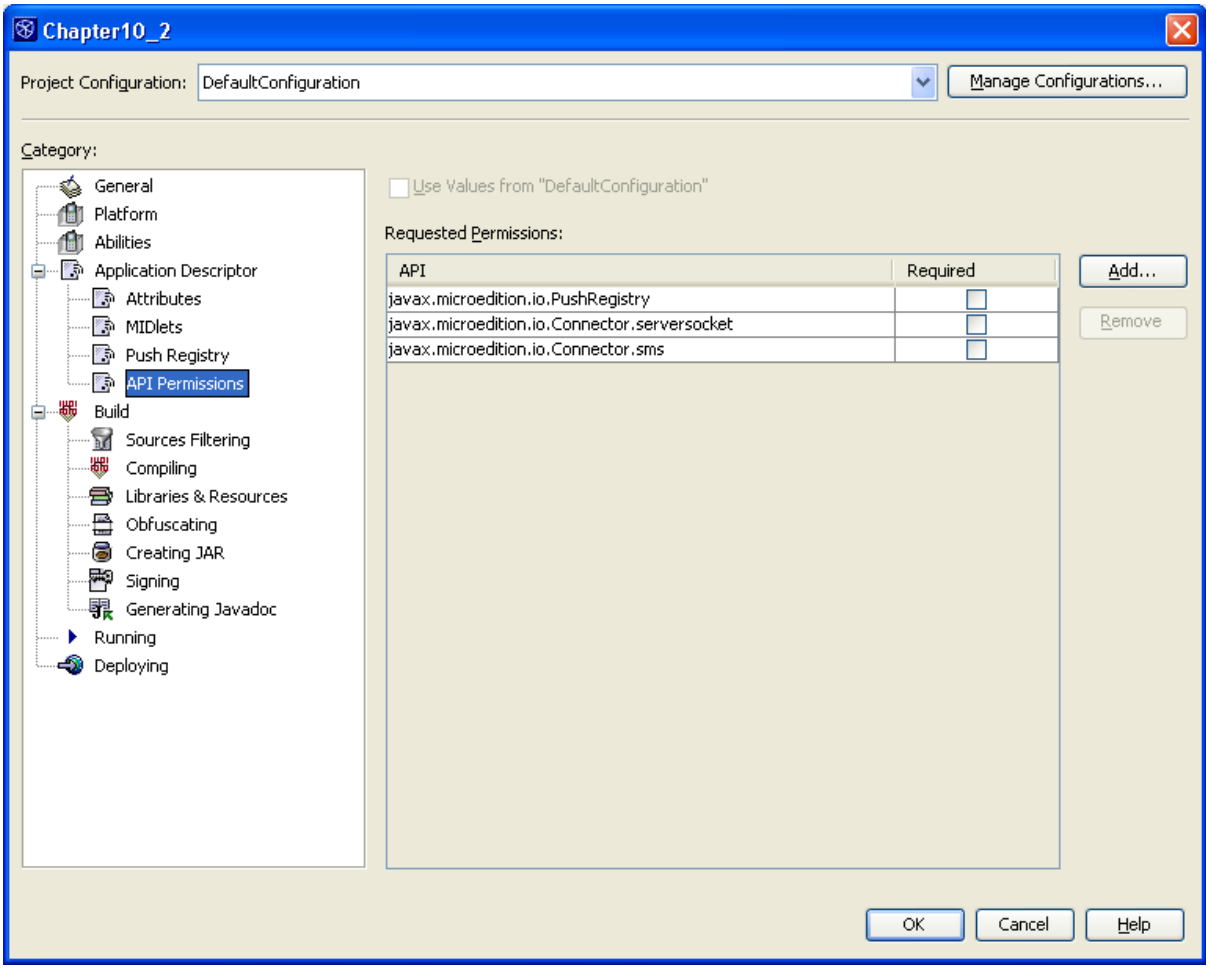
Pilih bagian "API Permissions":



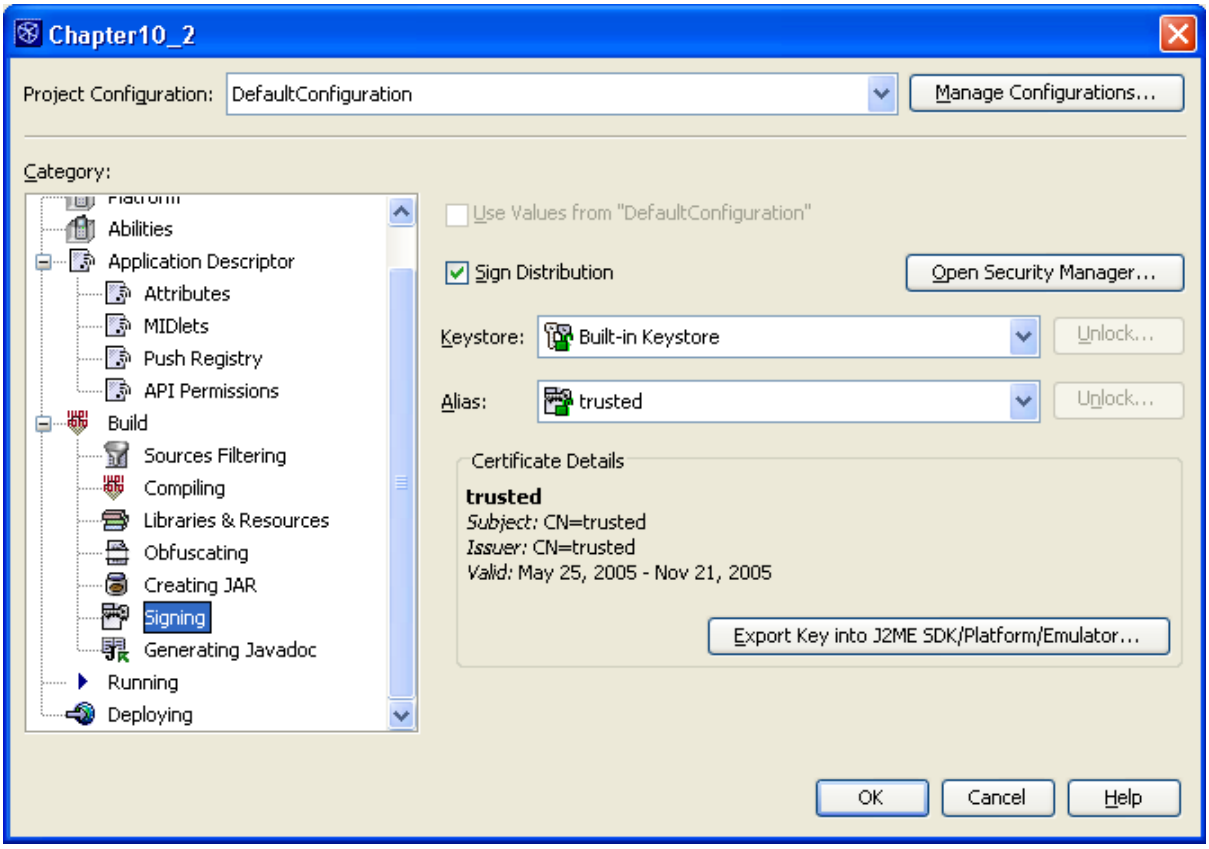
Pilih "Add" untuk menambah ijin (*permission*) untuk aplikasi MIDlet. Kita harus menambahkan API javax.microedition.io.PushRegistry untuk menginstall aplikasi. Kita juga harus menambahkan semua API yang digunakan oleh aplikasi:



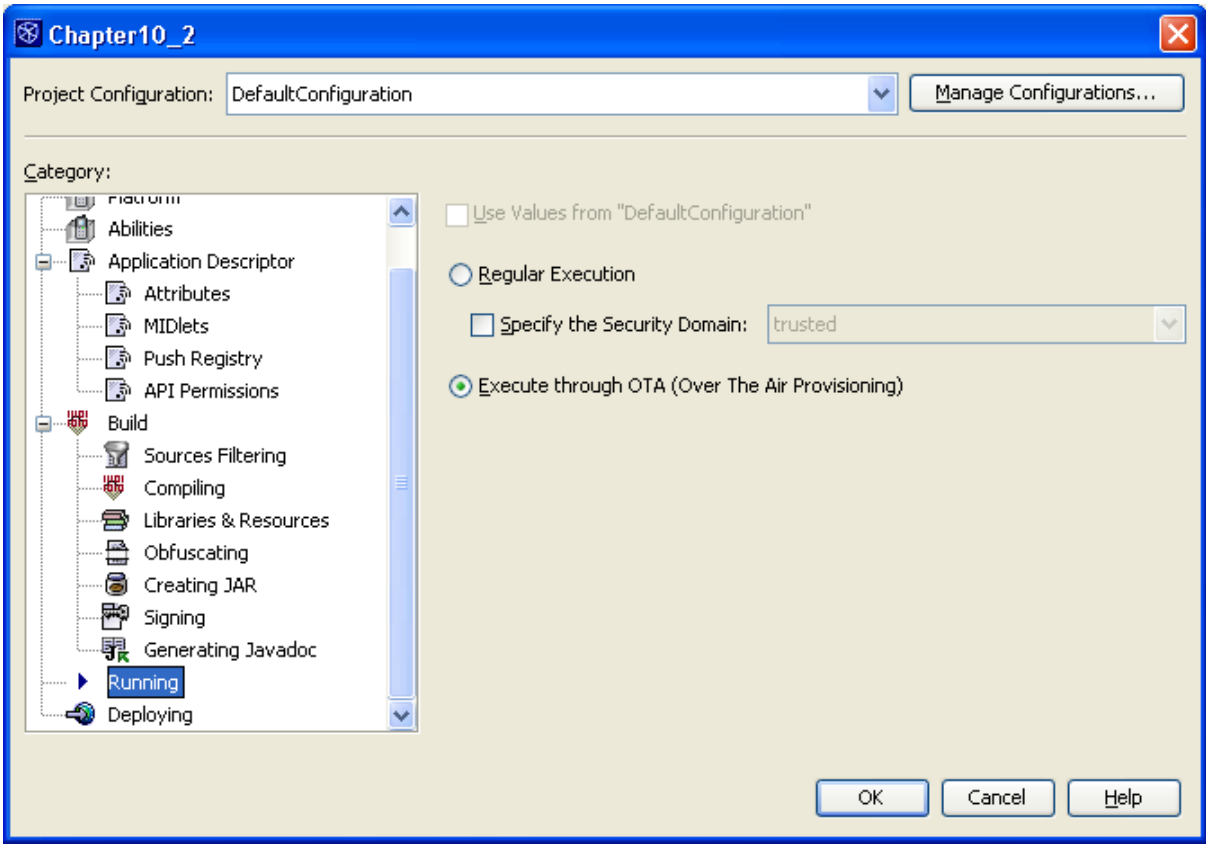
Hilangkan tAnda pada bagian required untuk semua API:



Pilih bagian "Signing" dan beri tAnda "Sign Distribution" untuk mendaftarkan aplikasi MIDlet:

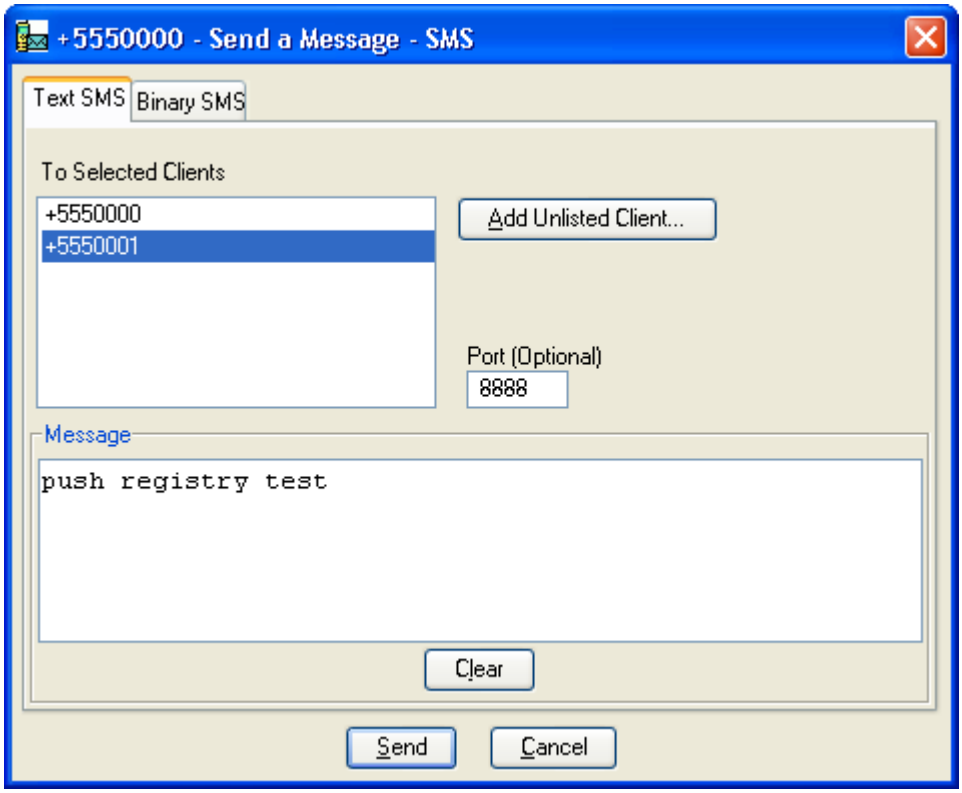


Pilih bagian "Running" dan pilih "Execute through OTA (Over the Air Provisioning)". Hal ini merupakan proses instalasi dan eksekusi aplikasi pada device.

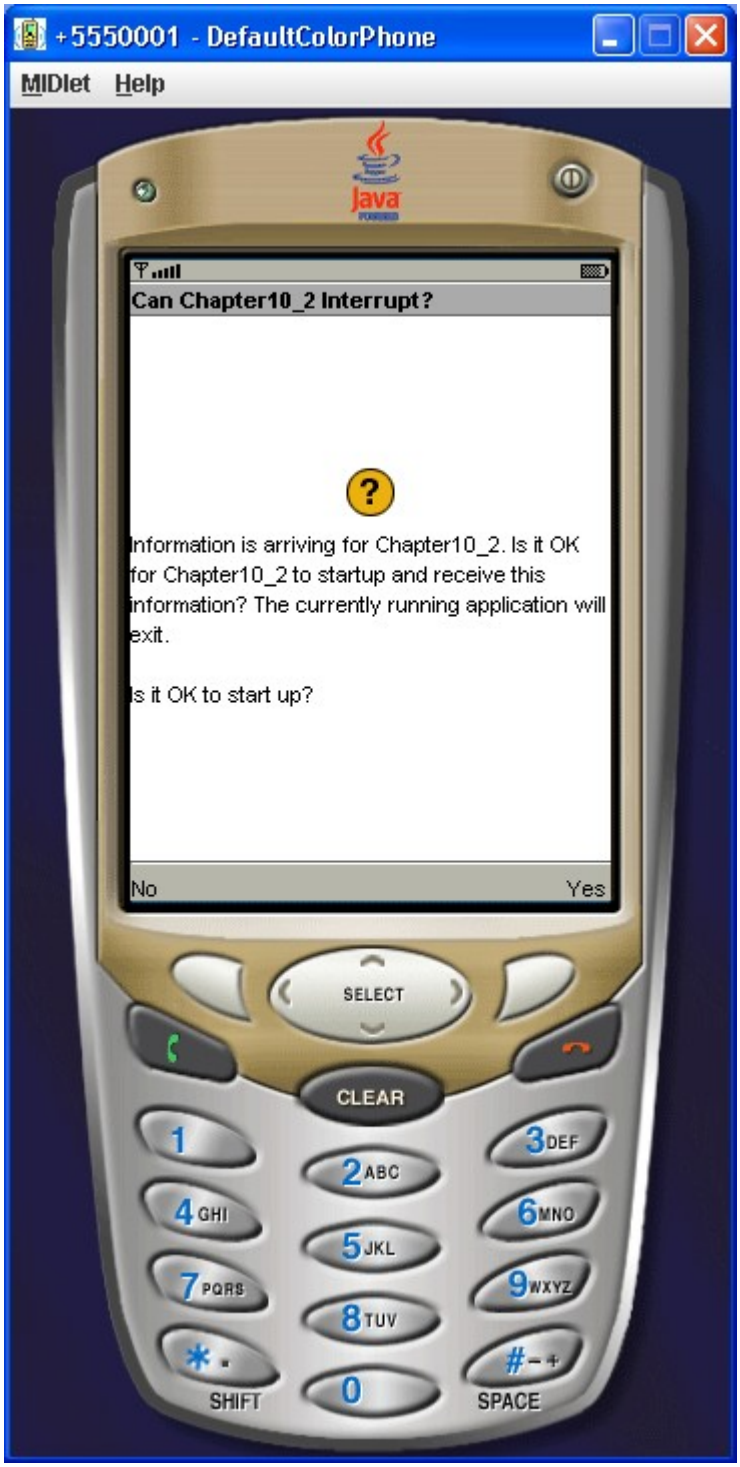


Langkah selanjutnya adalah menjalankan aplikasi MIDlet. Pastikan *build* berjalan dan tidak ada error ketika melakukan instalasi device (via OTA provisioning).

Untuk menjalankan aplikasi MIDlet, gunakan WMA console (Tools -> Java Platform Manager -> J2ME Wireless Toolkit 2.2 -> Open Utilities -> WMA: Open Console -> Send SMS...). Pilih nomor device, tentukan nomor port yang sudah ada pada PushRegistry, masukkan pesan dan klik "Send":



AMS akan menerima koneksi yang datang dan menanyakan konfirmasi selanjutnya kepada user:



Ini adalah aplikasi MIDlet, dijalankan melalui Push Registry (melalui pesan SMS masuk):

Ini adalah aplikasi kita yang dijalankan melalui Push Registry (socket pada port 1234). Untuk menjalankan aplikasi MIDlet dengan cara ini, layar console dan ketik "telnet localhost 1234".

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

import java.io.*;
import java.util.Timer;
import java.util.TimerTask;
import javax.microedition.io.*;

public class PushMidlet extends MIDlet implements CommandListener{
    private Command exitCommand;
    private Form form;
    private StringItem textField;
    private Display display;

    private String[] connections;

    public PushMidlet() {
        exitCommand = new Command("Exit", Command.EXIT, 1);
        textField = new StringItem("Status", "");

        form = new Form("Push via sms message");
        form.addCommand(exitCommand);
        form.append(textField);
    }

    public void startApp() {

        connections = PushRegistry.listConnections(true);

        if (connections != null && connections.length > 0){
            textField.setText(
                "Launched via Push Registry: " + connections[0]);
        }


        display = Display.getDisplay(this);
        form.setCommandListener(this);
```

```
        display.setCurrent(form);
    }

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

    public void commAndaction(Command c, Displayable d) {
        if (c == exitCommand) {
            notifyDestroyed();
        }
    }

    public void setText(String text){
        textField.setText(text);
    }
}
```

11.4 Latihan

11.4.1 Time Midlet

Buatlah sebuah aplikasi MIDlet yang menampilkan tanggal dan waktu hari ini dan terupdate setiap detik. Gunakan Timer untuk melakukan update dan StringItem untuk menampilkan tanggal dan waktu.