Como adicionar contatos usando a JSR 75

Os trechos de código a seguir mostram como adicionar contatos usando a JSR 75.

```
/**
  * PIMWrite Classe
  * Esta demonstração de exemplo gera os nomes e números aleatórios os quais serão
adicionados na lista telefônica.
  * Este código será útil, se nós quizermos adicionar mais contatos para nossa agenda
para testes.
  * Se nós precisamos ver o desempenho das agendas dos telefones de contatos, com
relação a velocidade de escrita e leitura e
  * Este código será útil
  * @ Author Senthil
  *
  */
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;
import java.util.*;

public class PIMWrite extends MIDlet implements CommandListener
{
  private Command PIMWriteCmd,exitCommand,generateName;
  PIMImpl pimImpl;
  private static Random random = new Random();
  static String[] names;

  public static Display display;
  public static Form result;
  private String sp,si,s1;
  private String[] strJSR = new String[12];
  private long l;
  int noOfContacts;

  TextField textfield = new TextField("Enter No of Contacts: ","",3,TextField.NUMERIC);


/*Construtor
*/
  public PIMWrite() {

  display = Display.getDisplay(this);

  PIMWriteCmd = new Command("PIMWrite", Command.SCREEN, 0);


  exitCommand = new Command("Sairt", Command.SCREEN, 0);

  result=new Form("Write Contacts To PIM");
     result.append(textfield);
  System.out.println(" noOfContacts: "+noOfContacts);

  result.addCommand(PIMWriteCmd);
  result.addCommand(exitCommand);
  result.setCommandListener(this);
```


(C) Copyright Nokia 2014. All rights reserved.

```
}

 public void startApp() throws MIDletStateChangeException {
   display.setCurrent(result);

 }

 public void pauseApp() {
 }

 public void destroyApp(boolean unconditional) {
 }

 public void commandAction(Command c, Displayable s)
 {
   if (c == exitCommand)
{
     destroyApp(false);
     notifyDestroyed();
   }

else if(c == PIMWriteCmd)
{
 noOfContacts = Integer.parseInt(textfield.getString());
 createConatactName();
  System.out.println(" noOfContacts: "+noOfContacts);
 //Iniciar a thread do PIM e escreve os contatos
  pimImpl = new PIMImpl();
  pimImpl.start();


   }
 }

public void createConatactName()
{
 names = new String[noOfContacts];
 for (int i = 0; i < noOfContacts; i++){
  names[i] = PIMWrite.generateName()+" "+PIMWrite.generateName();
 System.out.println(" Names :"+ names[i]+ " i "+i);
 }
}


 public static String generateName()
{
 return generateName(4, 7, true);
}


/** Este método gera os nomes aleatórios
          *
          * @param Tamanho mínimo minLength do nome
          * @param Comprimento máximo do nome maxLength
          * @param Capitalizar Captilize
          * @Return
```


(C) Copyright Nokia 2014. All rights reserved.

```
        */
  public static String generateName(int minLength, int maxLength, boolean capitalize)
 {

  int limit = (random.nextInt(maxLength - minLength + 1)) + minLength;
  StringBuffer s = new StringBuffer(limit);
  for (int i = 0; i < limit; i++)
  {
   char c = (char)((random.nextInt(26)) + 'a');
   s.append(c);
  }
  if (capitalize && (s.length() > 0)) s.setCharAt(0,
Character.toUpperCase(s.charAt(0)));
  return s.toString();
 }
}
```

Implementação da classe PIM

```
/**
* Implementação da classe PIM
*/
import javax.microedition.pim.*;
import java.util.*;
import javax.microedition.lcdui.*;
import javax.microedition.lcdui.Command;

public class PIMImpl extends Thread
{
static Alert check;

StringItem si;
//----------------------------------------Agenda---------------------------------

 int noOfRecords;
 public static boolean pimTemp = false;
 Random randPh = new Random();


 public PIMImpl()
 {

 }

    public void run()
    {
  writeData();
    }



public synchronized void writeData()
{
```


(C) Copyright Nokia 2014. All rights reserved.

```
String ph[] = new String[PIMWrite.names.length];
      for (int i=0;i<PIMWrite.names.length ;i++ )
      {

   String r = Long.toString(randPh.nextLong());
   if(r.startsWith("-"))
   {
    r = r.replace('-','9');
   }
   ph[i] =(r.length()>10)?(r.substring(0,10)):(r);



//   System.out.println(" PH :"+ph[i] +"i:"+i);
   }


 int tc = 0;
 PIM pim;
    try{

 ContactList contacts = null;
 ContactList conT = null;
 pim = PIM.getInstance();
  try {

  contacts = (ContactList) pim.openPIMList(PIM.CONTACT_LIST, PIM.READ_WRITE);
  } catch (PIMException e) {
  // Caso ocorra algum erro
  return;
  }
    for(int i=0;i<PIMWrite.names.length;i++)
 {

  Contact contact = contacts.createContact();
  String[] name = new String[contacts.stringArraySize(Contact.NAME)];

//   System.out.println("i = "+i);
  if (contacts.isSupportedField(Contact.TEL))
  {
   contact.addString(Contact.TEL, Contact.ATTR_MOBILE, ph[i]);
  }

  if (contacts.isSupportedArrayElement(Contact.NAME, Contact.NAME_GIVEN))
    name[Contact.NAME_GIVEN] = PIMWrite.names[i];
  contact.addStringArray(Contact.NAME, PIMItem.ATTR_NONE, name);


  try {
    contact.commit();
  } catch (PIMException e) {
    // An error occured
  }
  PIMWrite.result.deleteAll();
  String msg=" Contacts Adding :"+i;
  si = new StringItem("",msg);
```


(C) Copyright Nokia 2014. All rights reserved.

```
   PIMWrite.result.append(si);

  }
     check = new Alert("Added","Added ",null,AlertType.CONFIRMATION);
   check.setTimeout(Alert.FOREVER);
   PIMWrite.display.setCurrent(check);

   try {
     contacts.close();
   } catch (PIMException e) {
   }

  }catch(Exception e){
   String er = e+" ; tc="+tc;
  check = new Alert( "Response",er,null,AlertType.CONFIRMATION );
  check.setTimeout(Alert.FOREVER);

  PIMWrite.display.setCurrent(check);
  }
 // return true;

 }

 }
```


(C) Copyright Nokia 2014. All rights reserved.