

# Message In A Bottle

## v2.0.3

### Quick Guide

02/10/2009

Copyright © 2005-2009 By Ugo Chirico. All rights reserved



## **Disclaimer of Liability**

The content of this manual has been checked for agreement with the software described. Since deviations cannot be precluded entirely, full agreement is not guaranteed. However, the data in this manual are reviewed regularly and any necessary corrections will be included in subsequent versions. Suggestions for improvement are welcomed.

Java (TM) and all Java related trademarks and logos are trademarks of Sun Microsystems, Inc.  
PGP ® is registered by PGP Corporation  
Other company products and service names may be the trademark or service marks of others.

## **Thanks**

Many thanks to Ornella Simeoli for her good suggestions and her kind patience during the nights I spent to develop Message in a Bottle.

---

**Content:**

<i>Abbreviations and Acronyms</i> .....	4
<i>Definitions</i> .....	4
<i>About this manual</i> .....	4
<i>1 Introduction</i> .....	5
<i>2 How it works</i> .....	5
2.1 Encrypting SMS using Public Key Exchange.....	5
2.1 Encrypting SMS using agreed Private Key .....	6
2.1 Secret Phonebook (aka Key-Ring).....	7
2.2 Secret SMS inbox and outbox .....	7
<i>3 Installation</i> .....	7
3.1 Supported Platforms.....	7
3.2 Installation from Mobile Internet Browser.....	7
3.3 Installation via Bluetooth or Infrared port.....	7
3.3 Installation from memory card.....	8
3.4 Installation using Nokia PC Suite .....	8
3.5 First activation.....	8
3.5 SMS sending/receiving authorization.....	9
<i>4 Using the Main Menu</i> .....	9
4.1 Sending the public key .....	9
4.2 Receiving a public key .....	10
4.3 Adding a new contact with established encryption key .....	10
4.4 Writing an SMS.....	10
<i>5 Technical details</i> .....	11

## Abbreviations and Acronyms

<b>MIABO</b>	Message In A Bottle
<b>SMS</b>	Short Message Service
<b>J2ME</b>	Java 2 Micro Edition
<b>MIDP</b>	Mobile Information Device Profile
<b>CLDC</b>	Connected Limited Device Configuration
<b>ECC</b>	Elliptic Curve Cryptography
<b>RSA</b>	Rivest Shamir Hadlemann (Cryptographic Algorithm)
<b>DH</b>	Diffie-Hellman (Key Exchange Algorithm)
<b>CDC</b>	Connected Device Configuration
<b>JVM</b>	Java Virtual Machine
<b>ASN.1</b>	Abstract Syntax Notation One
<b>WMA</b>	Wireless Messaging API
<b>AES</b>	Advanced Encryption Standard

## Definitions

<b>Public Key Exchange</b>	Operation performed by two entities when one entity send its own public key to the other entity and vice versa. At the end of this operation both entities can calculate on their own a common master encryption key.
<b>Master Encryption Key</b>	Encryption used computed after Public Key Exchange used to generate encryption keys used to encrypt SMS.
<b>Elliptic Curve Cryptography</b>	Elliptic Curve Cryptography (ECC) is a public key crypto system
<b>Public Key Cryptography</b>	See <a href="http://en.wikipedia.org/wiki/Public-key_cryptography">http://en.wikipedia.org/wiki/Public-key_cryptography</a>

## About this manual

Message in a Bottle User Manual is designed for experienced Mobile Phone Users having good familiarity with SMS exchange and a minimal knowledge of public key cryptography.

An updated electronic copy of this manual is available at the following URL:

<http://www.ugosweb.com/miabo>

# 1 Introduction

*Message in a Bottle* is an application for Java Enabled Mobile Phone for sending and receiving encrypted or digitally signed SMS messages using Elliptic Curve Cryptography.

*Message in a Bottle* can send and receive Encrypted SMS in order to exchange securely confidential information such as secret messages, passwords, credit card numbers, phone numbers etc. and to avoid from secrets swiping, industrial espionage, password stealing, and, more in general, SMS eavesdropping.

*Message in a Bottle* can also send Signed SMS in order to avoid from SMS tampering and ensuring integrity and non repudiation of messages.

Secret SMS are stored in a secure separate repository (different from the usual handset's SMS repository),

Secret contacts are stored in a secure separate phonebook (different from the usual handset's phonebook).

Secret phonebook and secret SMS store are protected (encrypted) by a user PIN. Thus, if someone snaffles the handset contacts and SMS cannot be read without the PIN code.

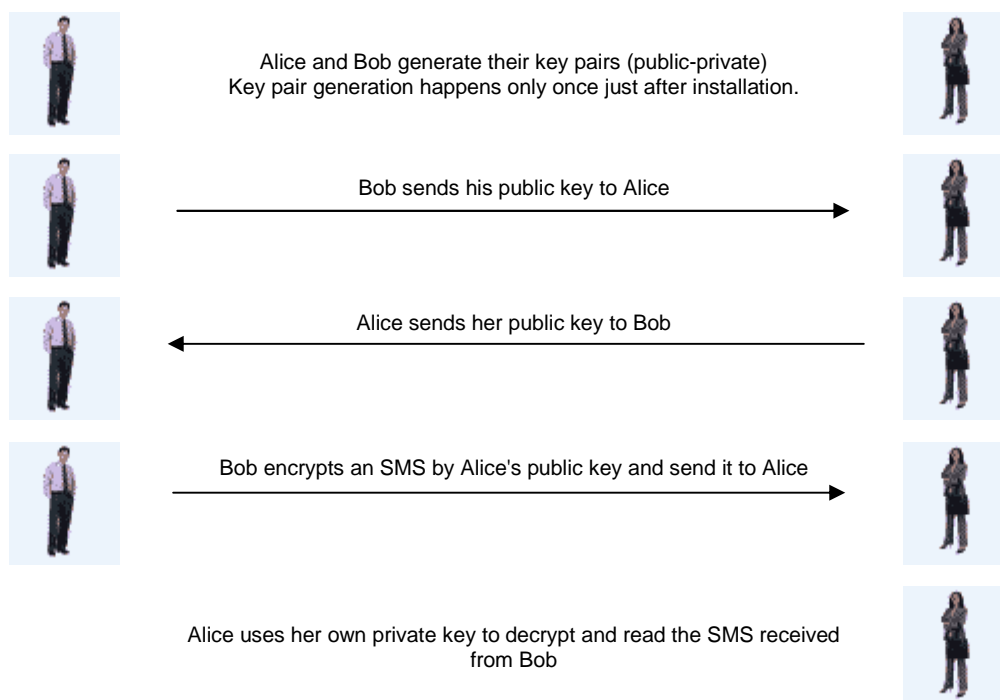
# 2 How it works

*Message in a Bottle* can work in two ways:

- 1) as a Public Key Cryptosystem (similar to PGP) where the two entities exchange their public keys, by which the Encryption Key for encrypting the SMS is generated
- 2) as a Private Key Cryptosystem where both entities establish together (by voice or by another way), the Private Key to use for encrypting the SMS.

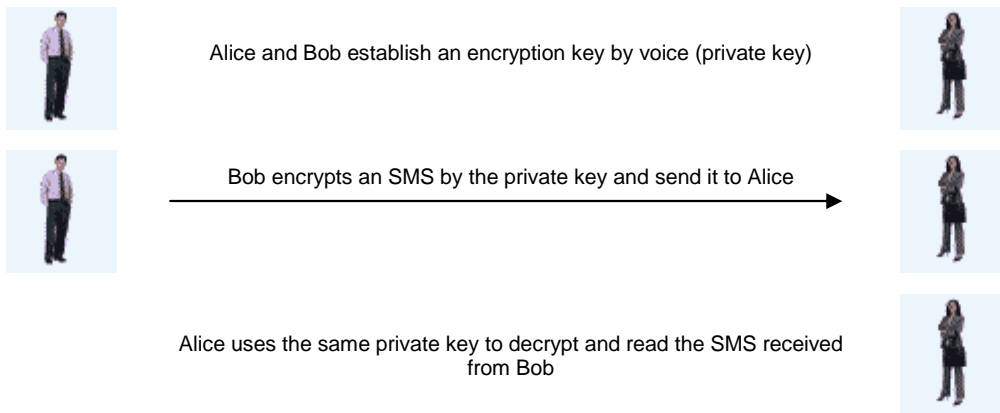
## 2.1 Encrypting SMS using Public Key Exchange

Alice and Bob want to talk to each other with encrypted/signed SMS. Each of them generates his/her own key pair (public part – private part) and sends his/her public key to the other (this process is called *Public Key Exchange*). Then Bob prepares an SMS encrypts by Alice's Public Key and sends it to Alice. Finally Alice uses her private key to decrypt the SMS:



## 2.1 Encrypting SMS using agreed Private Key

Alice and Bob want to talk to each other with encrypted SMS. They establish together a Private Key (for example by voice). Then Bob prepares an SMS encrypts by such a Private Key and sends it to Alice. Finally Alice uses the same Private Key to decrypt the SMS:



## **2.1 Secret Phonebook (aka Key-Ring)**

*Message in a Bottle* stores contacts in a secret phonebook protected by the user PIN. Each contact has a name, a phone number and a master encryption key. This is why it is also called Key-Ring.

Such an encryption key is used as a master key to generate the encryption keys used to encrypt SMS.

The master key:

- 1) is computed when a public key exchange is done

or

- 2) is entered by the user when it is established in another way instead of public key exchange (for example by voice or another mean).

The "send public key" function (see below) allows sending the public key to others in order to start public key exchange.

## **2.2 Secret SMS inbox and outbox**

*Message in a Bottle* stores incoming, sent and draft SMS in a secret store protected by the user PIN.

## **3 Installation**

The installation of *Message in a Bottle* is really simple and is similar to the installation of a Java game. Because each handset has its own installation procedure for installing java applications the following instructions are only a sort of guide lines to drive you during installation.

For a comprehensive guide please refer to your handset's user guide.

### **3.1 Supported Platforms**

*Message in a Bottle* can be installed on any Java enabled mobile phone compliant with:

- Java MIDP2.0
- Wireless Messaging API (WMA 1.0 / JSR120 or later)

### **3.2 Installation from Mobile Internet Browser**

- 1) Point the internet browser of your mobile phone to:

Standard Edition:

<http://www.ugosweb.com/miabose>

Business Edition:

<http://www.ugosweb.com/miabobe>

Government Edition:

<http://www.ugosweb.com/miaboge>

- 2) Follow the instructions on your mobile phone to complete the installation. Some handsets, during installation, could advise you that the application isn't secure. To complete the installation ignore such a message (such a message is because *Message in a Bottle* isn't digitally signed by a digital certificate known by the handset).

- 3) After the installation is completed follow the instructions reported in the paragraph "First Activation" (see below)

### **3.3 Installation via Bluetooth or Infrared port**

- 1) Download the JAR file from the following link:

<http://www.ugosweb.com/miabo/download.aspx>

- 2) send the downloaded file to you mobile phone via infrared or bluetooth connection
- 3) open on your mobile phone the received file
- 5) follow the instructions on your mobile phone to complete the installation. Some handsets, during installation, could advise you that the application isn't secure. This is because Message in a Bottle isn't digitally signed by a digital certificate known by the handset. In this case, to complete the installation ignore such a message.
- 6) after the installation in completed follow the instructions in the paragraph "First Activation" (see below)

### **3.3 Installation from memory card**

- 1) Download the JAR file from the following link:  
<http://www.ugosweb.com/miabo/download.aspx>
- 2) Copy the downloaded file on your memory card
- 3) Insert the memory card in the mobile phone
- 4) Open on your mobile phone the file just copied
- 5) Follow the instructions on your mobile phone to complete the installation. Some handsets, during installation, could advise you that the application isn't secure. This is because Message in a Bottle isn't digitally signed by a digital certificate known by the handset. In this case, to complete the installation ignore such a message.
- 6) After the installation in completed follow the instructions in the paragraph "First Activation" (see below)

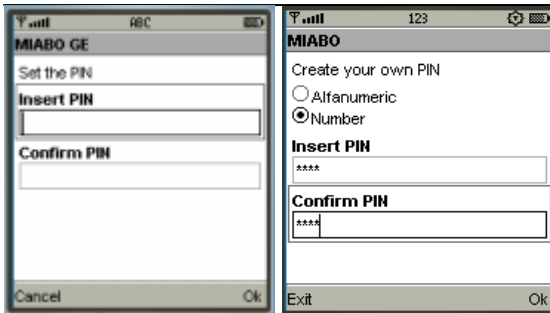
### **3.4 Installation using Nokia PC Suite**

- 1) Download the JAR file from the following link:  
<http://www.ugosweb.com/miabo/download.aspx>
- 2) Right click on the downloaded file and select "Install by Nokia Application Installer"
- 3) Follow the instructions on the PC and on your mobile phone to complete the installation. Some handsets, during installation, could advise you that the application isn't secure. This is because Message in a Bottle isn't digitally signed by a digital certificate known by the handset. In this case, to complete the installation ignore such a message.
- 6) After the installation in completed follow the instructions reported in the paragraph "First Activation" (see below)

### **3.5 First activation**

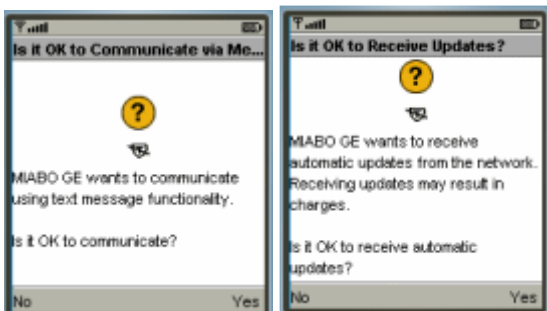
Once installed, at the first start, Message in a Bottle asks to type the PIN code (and to retype the PIN as confirmation) by which the secret phonebook and secret SMS store will be encrypted. From now on every time you start Message in a Bottle a correct PIN must be entered to gain access to the secret phonebook and SMS store.

For Standard and Business Edition (at left) a numeric PIN must be entered, for Government Edition (at right) a numeric or alphanumeric PIN can be entered



### 3.5 SMS sending/receiving authorization

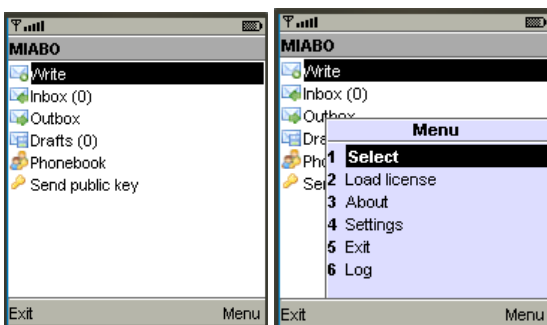
Every time *Message in a Bottle* tries to send/receive an SMS, the mobile device asks the user for the authorization with something similar to the following pictures:



To complete the operation authorization must be given by clicking on “yes”.

## 4 Using the Main Menu

The main menu is shown in the following pictures:



The main operations are:

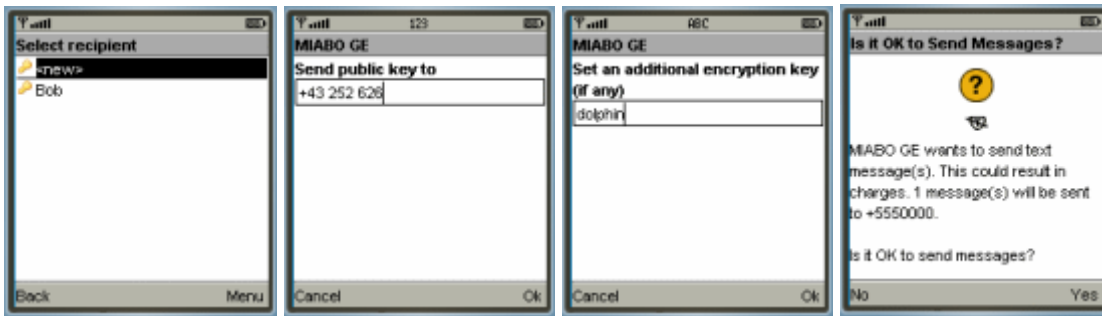
- *Write* to write a new SMS.
- *Inbox* shows the list of received SMS (*in parenthesis the number of unread SMS*).
- *Outbox* shows the list of sent SMS.
- *Draft* shows the list of the drafts (*in parenthesis the number of drafts*)
- *Phonebook* shows the phonebook
- *Send Public Key* sends the public key to a recipient

In the context menù (at right) there are:

- *Load License* to load a license key
- *About* shows information about the program
- *Settings* allows to change application settings
- *Exit* exits the application and returns to the phone’s menu
- *Log* shows log and error events

### 4.1 Sending the public key

Select Send Public Key from the main menu. The following are the steps to send own public key:



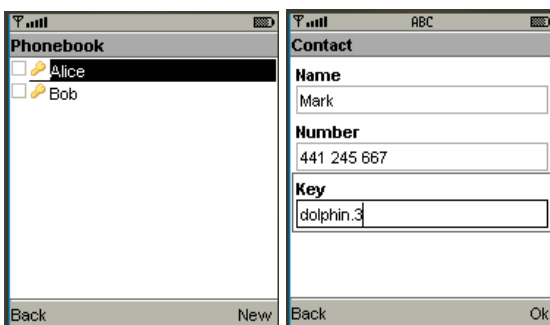
#### 4.2 Receiving a public key

Message in a Bottle notify you of an incoming public key:



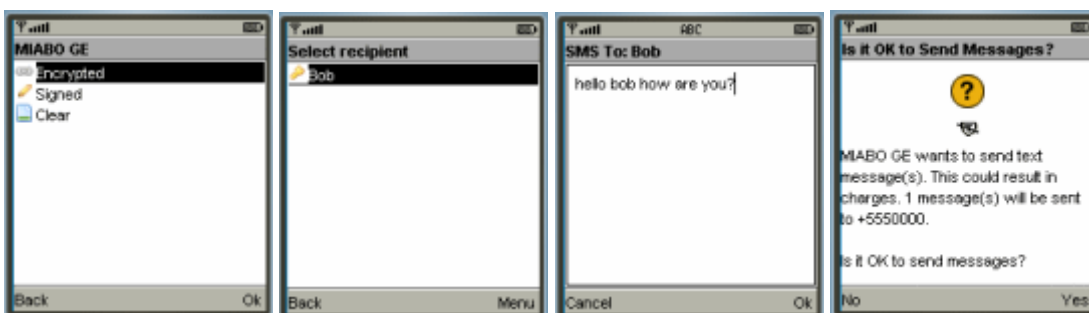
#### 4.3 Adding a new contact with established encryption key

To add a contact with established encryption key (without public key exchange) select Phonebook from the main menu. The following are the steps to add a new contact with established encryption key:



#### 4.4 Writing an SMS

Select *Write* from the main menu. The following are the steps to create and send an encrypted SMS:



## 5 Technical details

*Message in a Bottle* uses a public key cryptographic engine based upon Elliptic Curve Cryptography (ECC). Such an algorithm is essential for encrypted/signed SMSes exchange because the size of the key (about 192 bit) agrees with the size of an SMS (that is 140 byte for binary SMS and 160 for text SMS). To have the same security a 1024 bit RSA key is comparable with 160 bit ECC key.

<i>Time to break in MIPS years</i>	<i>RSA/DSA key size</i>	<i>ECC key size</i>	<i>RSA/ECC key size ratio</i>
$10^4$	512	106	5 : 1
$10^8$	768	132	6 : 1
$10^{11}$	1,024	160	7 : 1
$10^{20}$	2,048	210	10 : 1
$10^{38}$	21,000	600	35 : 1

**Picture 1 Elliptic Curve Cryptography vs. RSA**

*Message in a Bottle* uses Elliptic Curve Cryptography over Prime Finite Field (Galois Field) GF(p) with  $||p|| = 192$  <sup>(1)</sup>:

- SMS encryption uses the algorithm Elliptic Curve Secret Value Derivation Primitive, Diffie-Hellman version (ECSVDP-DH) as reported in IEEE-1363 §7.2.1 and AES256 for symmetric encryption
- SMS digital signature uses the algorithm Elliptic Curve Signature Primitive, DSA version (ECSP-DSA) as described in IEEE-1363 §7.2.7
- SMS digital signature verification uses the algorithm Elliptic Curve Verification Primitive, DSA version (ECVP-DSA) as described in IEEE-1363 §7.2.8
- The SMS repository (sent and received SMS) is encrypted by the symmetric cryptographic algorithm AES256