MorseCode application documentation

by Putanu Alexandru E2

Description

This app should take a phrase and at the touch of a button it should make the led on the phone flash the morse code for that phrase. It should also display the code for a certain character when the led is flashing it. The app should also be able to send that code via sms. Also, the app should be able to decrypt the received sms.

Morse Code

The Morse Code is a method of telecommunication used to encode text. It has been invented by Samuel Morse, the inventor of the telegraph. It uses only 2 types of signals, different by duration, the shorter is called a dot (.) and the other one is called a dash (-). Here is a map of all the English alphabet characters and the respective Morse codes (1) for letters and digits.

There are certain rules to be followed if we want a successful and correct Morse communication. These rules are:

- The length of a dot is one unit.
- A dash is three units.
- The space between parts of the same letter is one unit.
- The space between letters is three units.
- The space between words is seven units.

What my application is trying to achieve is communication using Morse code between two people. By installing it on an Android, it can use the flashlight to signal the Morse code of a text typed in by the user; it can send the desired text via a messaging app to a friend; it can decode a given Morse code back into letters and digits.

Usage

In this chapter I am going to present the typical use of the application, together with a pair of screenshots to exemplify.

When opening the app, the user is greeted with the Main menu (2). The user can immediately write a text and message it physically using the device's back flashlight as the transmitter. After typing the text and pressing the "TRANSLATE" (3) button, the phone will start flashing with the first back torch the message using short and long signals, with the unit of 300ms, according to the rules specified in the previous chapter. While the message is displayed, letter by letter, the current displayed character will appear on the screen, together with the Morse code, so the user can verify the correctness of the message transmitted himself.

If he wants to learn the Morse code better, the user can press the "MORSE ALPHABET" (4) button, which prompts him with a picture of letters, digits and their codes.

If the user wants to send this code to a friend, he can press the "SEND VIA SMS" button, which prompts the "share via.." dialog (5) of the android system, which asks the user about the messaging platform he wants to send the message through. By choosing the messaging app, together with a recipient, the end-user will then send a message (6) to the contact with the Morse code of the previously entered text.

After receiving such a message, the recipient can open the app, click on "DECRYPT MESSAGE" (7) button and then paste the content of the message. By pressing decrypt, the english letters text message will appear on the screen (8).

Workflow

In the beginning, I created a "Morse" java class, which would later contain the Morse codes for digits and letters and functions for translating words from Morse to english and backwards. After that I started documenting on the methods of creating time-based events programmatically. The method I chose was based on handlers. Handlers are used to schedule runnables to be executed at some point in the future. Because the display of letters and digits on the screen and the torch had to be synchronous, I chose to create 4 runnables, for the following jobs:

- Displaying a letter and it's morse code on screen.
- Removing the current letter and Morse code from the screen.
- Lighting the torch according to the current character (dot or dash).
- Turn the light off for the torch according to the spaces (space between letters or space between words).

The times for spaces, dots and dashes were standard and always based on the current character, but for the showing/hiding of letters, a timetable would be needed. I solved this by implementing the getTimes() function inside the Morse class. This function receives a String of the input text and returns an array of ints, which describe the time that each character at a certain position should stay on the

screen. This was achieved by summing up the times of the dots + dashes + spaces for each character, be it space, letter or digit.

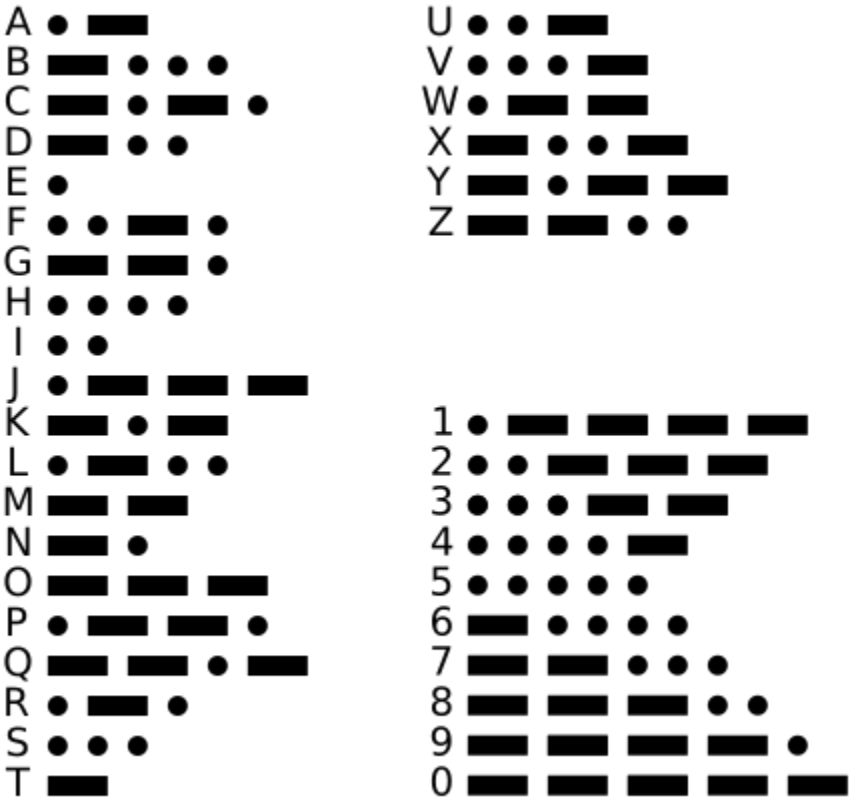
The next step was creating the runnables for each of these tasks. Here (9) is a quick diagram of how every runnable acts. Using the postDelayed(Runnable r, long delayMillis) method of a Handler, we can schedule a runnable r to execute after delayMillis miliseconds. In the code, we can see the first runnable r1 taking care of the displaying of characters. It iterates through each character of the input text and starting from time 0 (the beginning of the translation), it calls showLetter(char letter) and displays the character at the beginning of the signals that it corresponds to.

Runnable r2 is the reverse of r1, it's job is to hide the characters after their signal is finished. Because the space between two letters is one unit, it is executed at time 0 + time of the first character – unit.

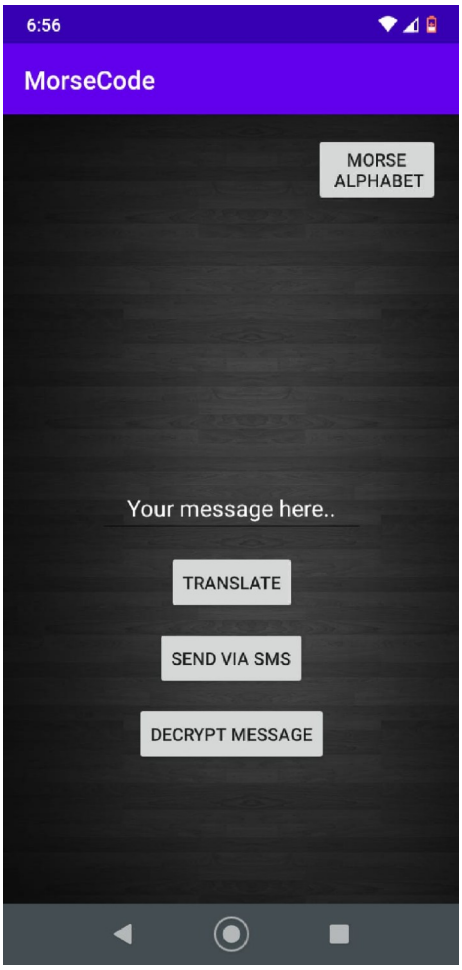
Runnable r3's job is to light the torch. Iterating through the Morse code of the input text, if the character is a dot or dash, it lights the flashlight. After that, it waits to be executed for a time according to the current character.

Runnable r4 is similarly an inverted version of r3. It stops the torch with one unit before the current character is finished displaying.

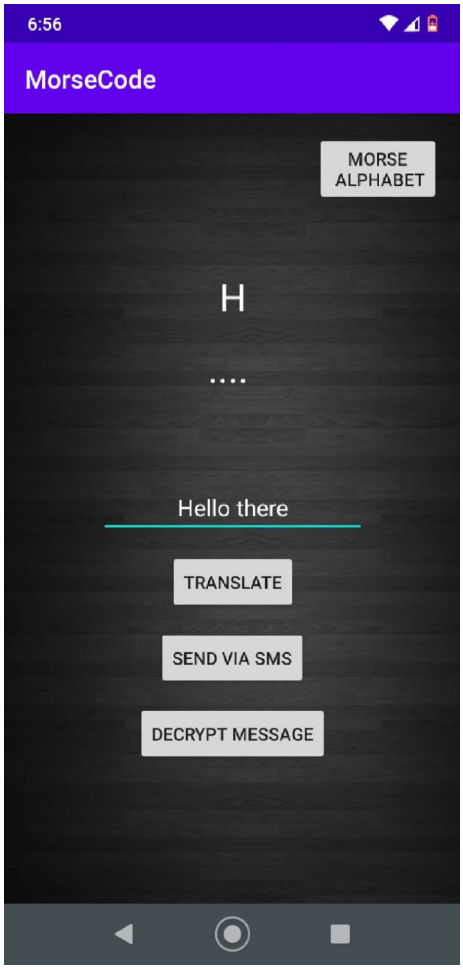
Having this part achieved, I then added buttons for the sms intent and for the decryption activity. The decryption is fairly simple, it just uses the Morse class to decode the incoming pasted message, but it first checks for the "Hey, put this text into MorseCode app:" intro. In the end of the development phase, I created support for digits.


A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
1
2
3
4
5
6
7
8
9
0

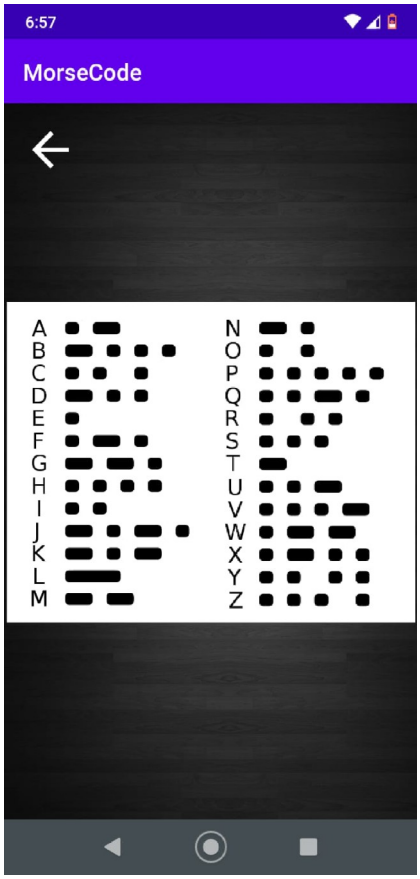
(1)


6:56
MorseCode
MORSE ALPHABET
Your message here..
TRANSLATE
SEND VIA SMS
DECRYPT MESSAGE

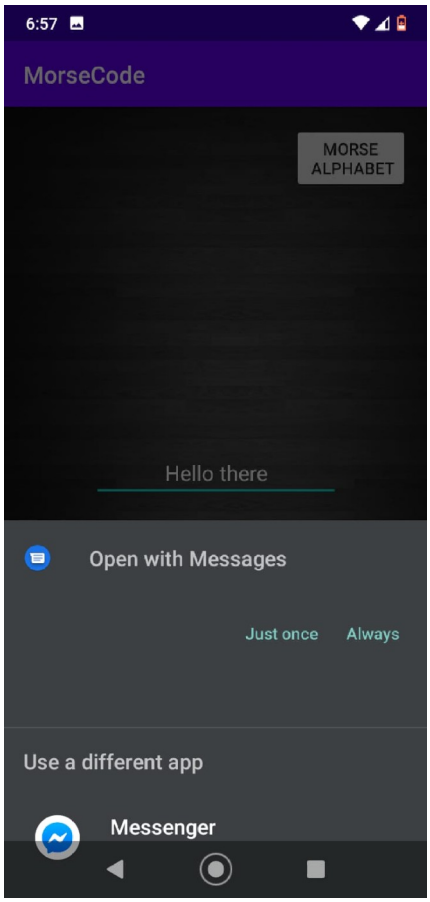
(2)


6:56
MorseCode
MORSE ALPHABET
H
....
Hello there
TRANSLATE
SEND VIA SMS
DECRYPT MESSAGE

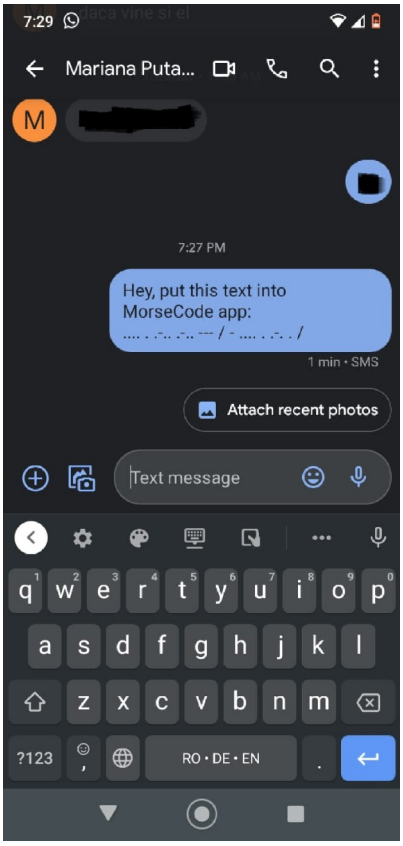
(3)


6:57
MorseCode
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

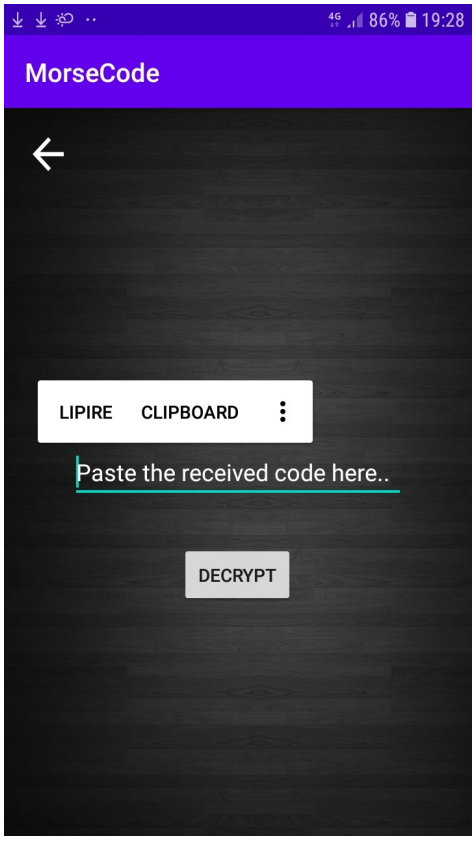
(4)


6:57
MorseCode
MORSE ALPHABET
Hello there
Open with Messages
Just once
Always
Use a different app
Messenger

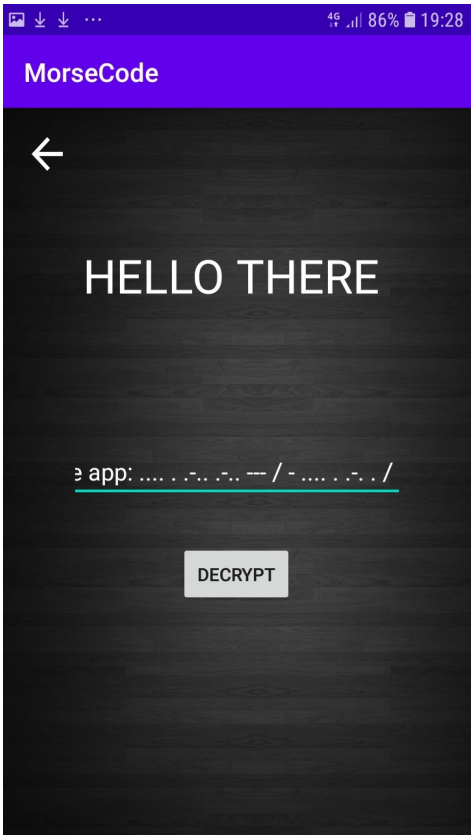
(5)


7:29
Mariana Puta...
7:27 PM
Hey, put this text into MorseCode app:
.... . .-.. .-.. --- / - .... . .-. . /
1 min • SMS
Attach recent photos
Text message
RO • DE • EN

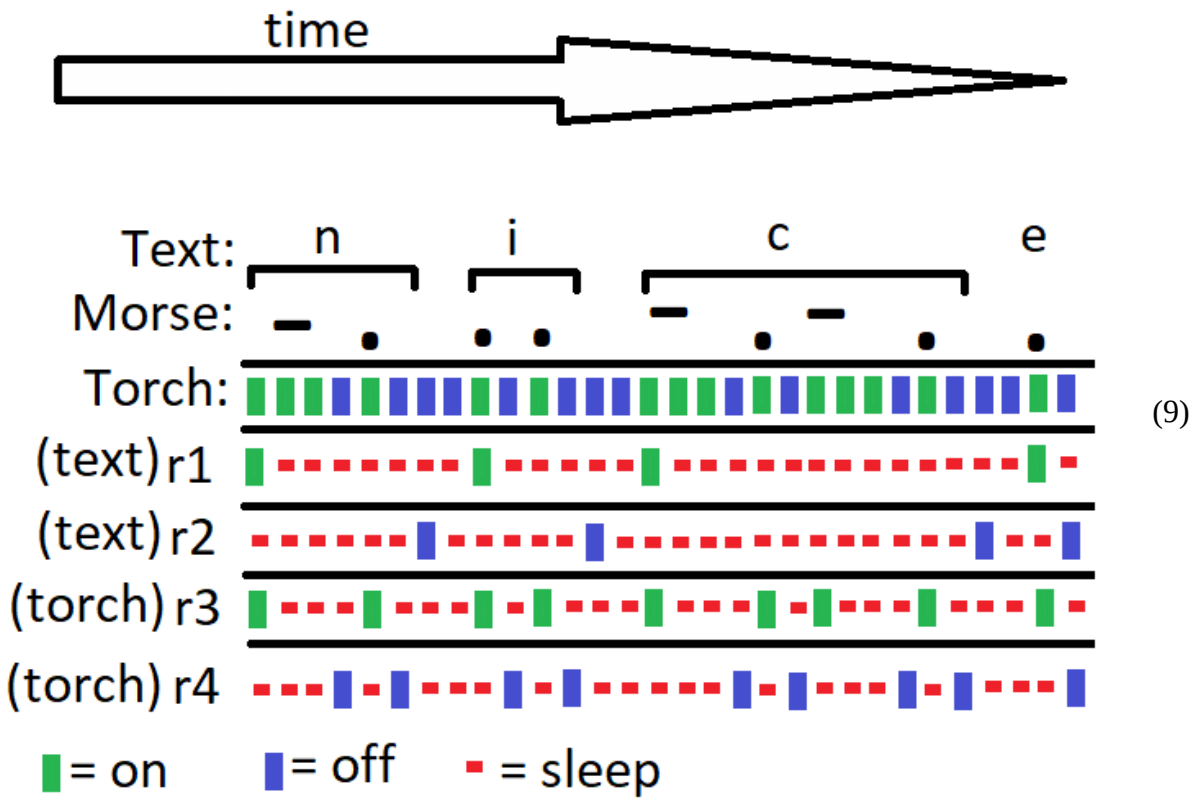
(6)


86% 19:28
MorseCode
LIPIRE
CLIPBOARD
Paste the received code here..
DECRYPT

(7)


86% 19:28
MorseCode
HELLO THERE
e app: .... . .-.. .-.. --- / - .... . .-. . /
DECRYPT

(8)


time
Text:
n
i
c
e
Morse:
Torch:
(text) r1
(text) r2
(torch) r3
(torch) r4
= on
= off
= sleep

(9)