Locked Messages

By Mathew McCrackin
Charleston Southern University

Fall 2017

Table of Contents

1. Description:

Locked Messages is an extension for the iOS Messages that is designed to add a layer of interactivity to texts. A text can't be viewed until its locks are unlocked. Various lock types such as being in the right place or opening a message at the right time are implemented. The possibilities are up to you! Send your friend on a scavenger hunt, send them a secret birthday message, remind them to pack a jacket when it's raining, or let them try to crack a PIN! Combine multiple messages and sets of locks together to send your friend on a quest!

2. Difficulties:

The main difficulty I faced while creating this application was a lack of documentation of subtle nuances on Apple's end. When creating a Messages Extension for iOS, a special view controller for Messages must be utilized. However, Apple doesn't mention that you aren't allowed to close this view controller at any time or re-instantiate it. Due to this, I had to have each view in my application point back to the Messages Controller and utilize a flowchart-style logic to determine which view should be loaded. So the Messages Controller must load each child view and then each child view must point back to the Messages Controller - even if a child view wants to simply load another child view.

Another difficulty was learning about the various storage types utilized by iOS. For example, local message storage and settings are stored in a User Defaults file. This file type utilizes a dictionary of key-value pairs that are both strings. This implies that storing variables requires converting their data to be strings and loading variables requires converting strings to the proper data type. Another storage type is the queryItem. This type of data is stored in a URLQueryItem array which is then encoded to send data through a text. Like variables stored in User Defaults, the queryItems are string key-value pairs. I also had to convert both to and from strings for these variables as well.

These were the main difficulties I faced while working on the project. Every other issue was simple to resolve and could easily be fixed by referring to the Swift Handbook.

3. Future Improvements:

The main feature I would have liked to have added to the application would have been NFC-tag support. However, Apple restricts the use of NFC-tags to paid developers. It would have been an interesting feature to be able to lock messages with an NFC-tag or to load messages from one. It would have also been relatively easy to implement.

Another feature I would have liked to have implemented would be the ability to share multimedia images and videos with the application and not just text. However, this would require the use of a 3rd party server to upload and receive the media as the amount of information transferable in an encoded text message is limited.

4. Hardware/Software required for project

MacBook / iMac
MacOS High Sierra
xCode
iPhone / iPad
iOS 11+
Lightning to USB cable

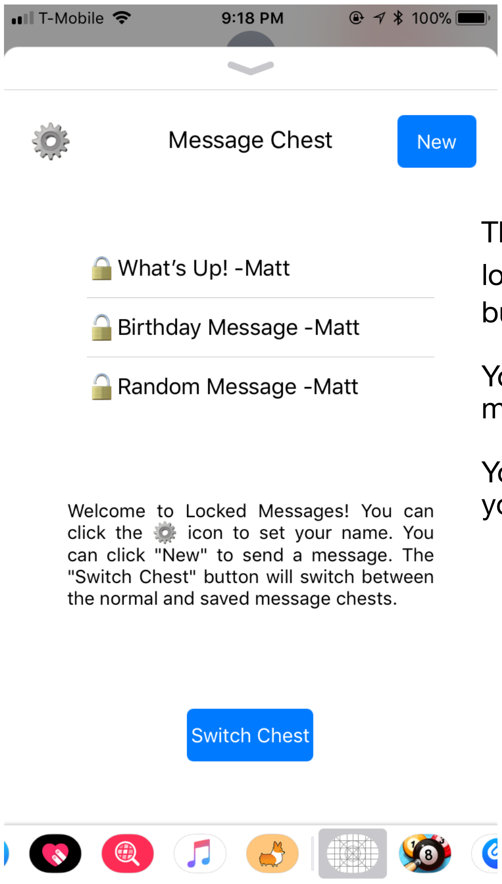
This application requires an iDevice to run. Technically, two are needed (as you will need someone to send messages). In order to run the application on an iDevice, it must be installed to the device. For this, an Apple Computer is required. Either a MacBook or an iMac will work so long as it is running the most recent OS (High Sierra).

On the Apple Computer, xCode must be installed. The code for the project should be imported using xCode. Then, the application may be installed by connecting the iOS 11+ device to the Apple Computer using the Lightning to USB cable and pressing CMD + F5 with the device targeted. For more detailed instructions for importing and running the project on a device, check Apple's xCode documentation for information on the most recent version.

It is imperative that the application be run with location services enabled. When the app initially runs, it will ask for permission to utilize location services, if you accidentally do not allow these services they may be enabled in Settings -> Privacy from within iOS.

Two phones are also needed to properly run the application as the application works by users sending messages to each other via Apple's Messages.

5. Instructions / Documentation


T-Mobile 9:18 PM 100%

Message Chest

New

What's Up! -Matt

Birthday Message -Matt

Random Message -Matt

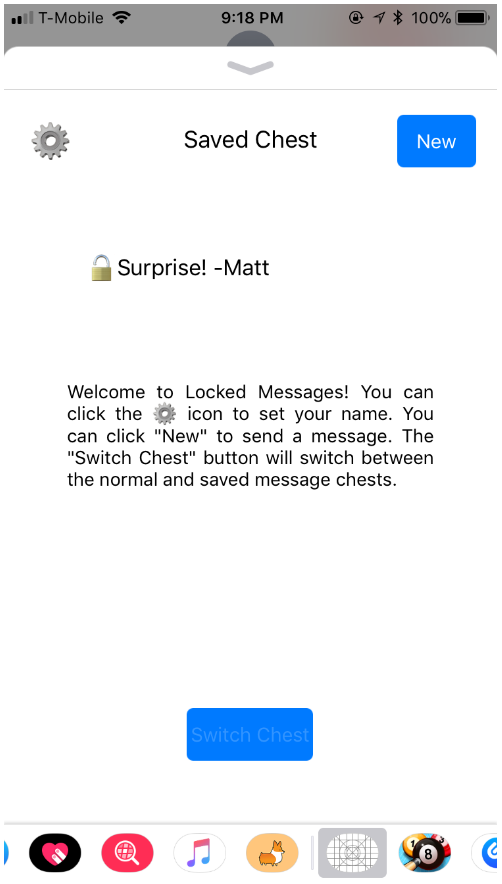
Welcome to Locked Messages! You can click the ⚙ icon to set your name. You can click "New" to send a message. The "Switch Chest" button will switch between the normal and saved message chests.

Switch Chest

This is the home screen that is displayed when the application is initially loaded. The ⚙ icon may be pressed to go to Settings. The "New" button will allow you to create a new locked message to send.

Your current messages are displayed in the message chest. Tap a message to view it.

You may also tap "Switch Chest" at the bottom to switch to viewing your Saved Chest.


T-Mobile 9:18 PM 100%

Saved Chest

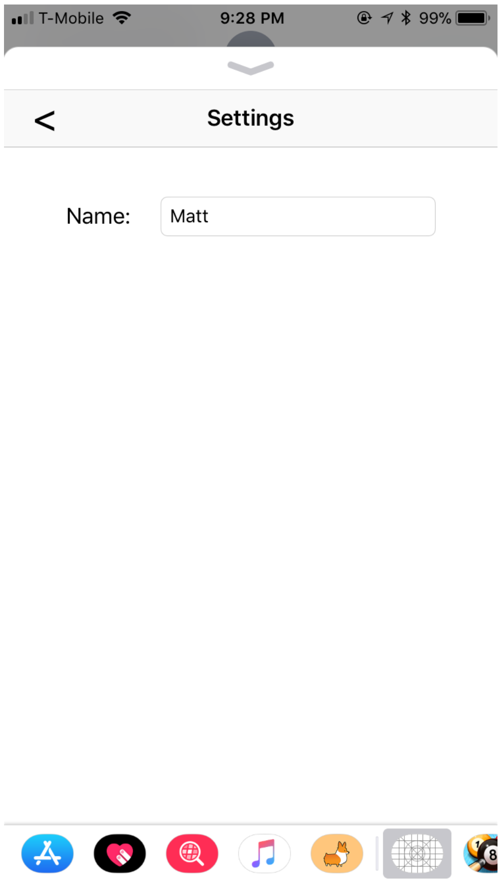
New

Surprise! -Matt

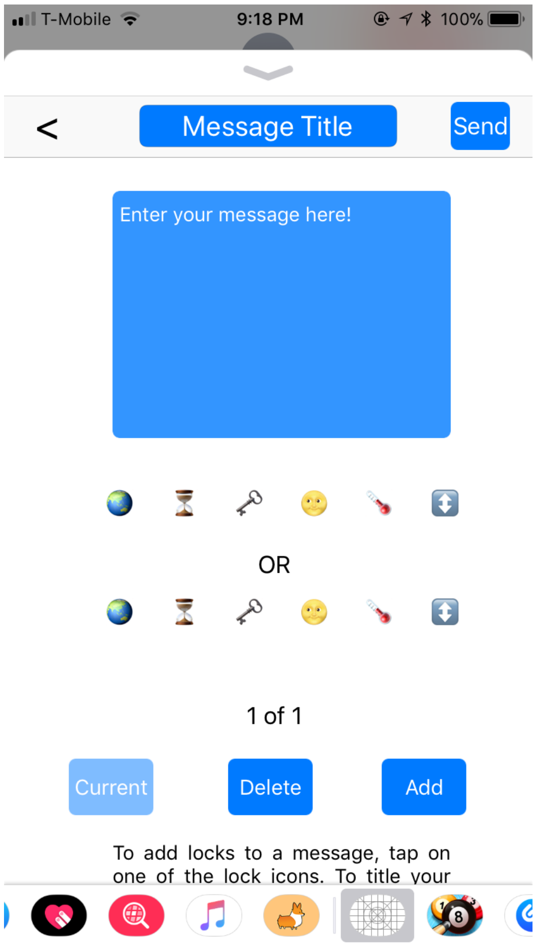
Welcome to Locked Messages! You can click the ⚙ icon to set your name. You can click "New" to send a message. The "Switch Chest" button will switch between the normal and saved message chests.

Switch Chest

Notice that pressing the "Switch Chest" button switches to the view of the Saved Messages



The Settings Menu is where you can set your name that will be used to sign your messages. If you do not set a name in Settings, your messages will be signed with the default "Sender" name.



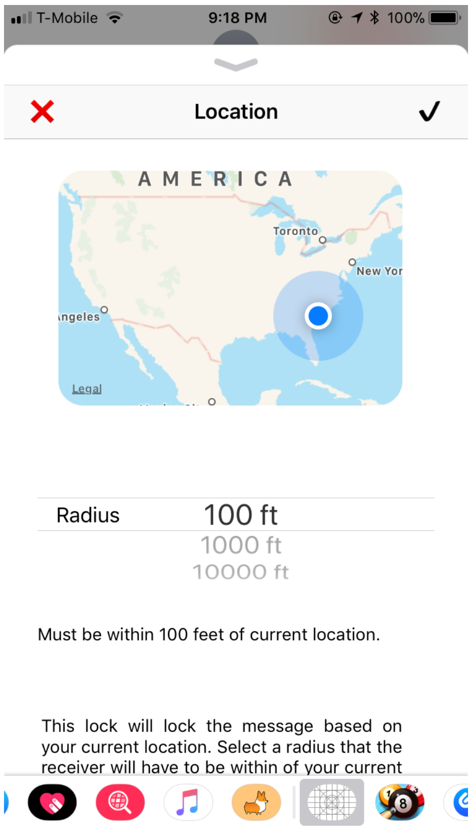
To add locks to a message, tap on one of the lock icons. To title your

This is the view for creating a message. It is reached by tapping the "New" button on the main menu. The Message Title may be tapped in order to modify the title of the message. The text-box may be tapped in order to type the message for a page.

Each lock icon may be pressed to add a lock to the current page. To add a page to the message, press "Add." The "Delete" button will delete the current page.

The buttons at the bottom double as navigation buttons and for adding pages to a locked message.

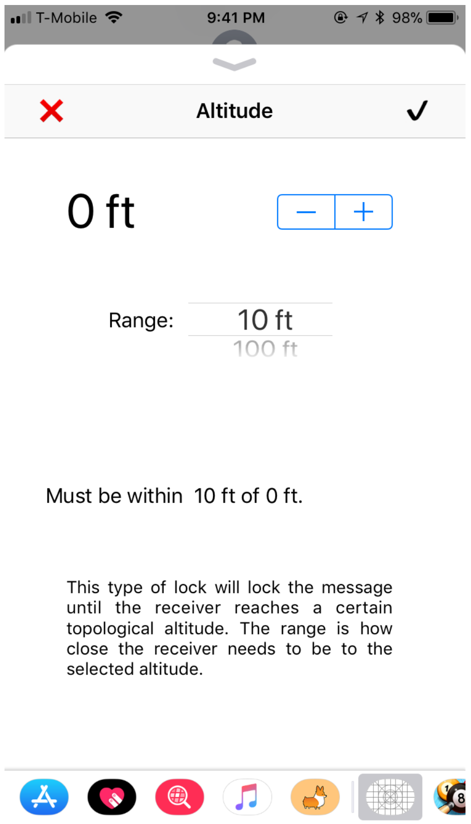
When you are satisfied with a message, the "Send" button is used to package it for sending to the receiver.



Radius 100 ft
1000 ft
10000 ft

Must be within 100 feet of current location.

This lock will lock the message based on your current location. Select a radius that the receiver will have to be within of your current

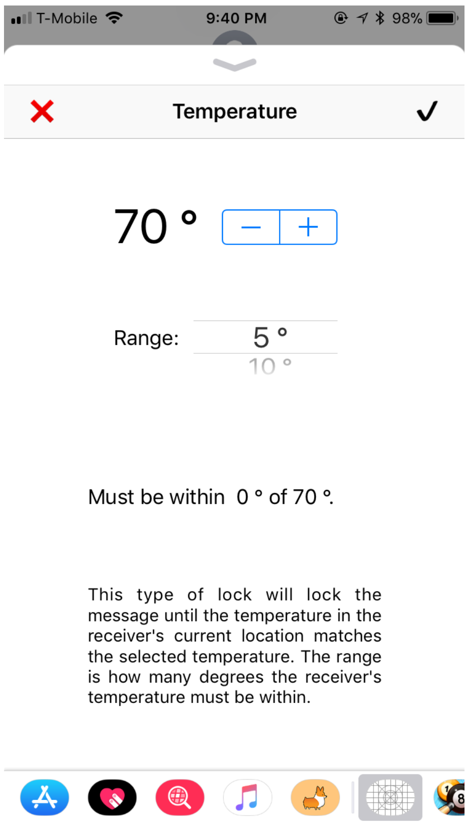


0 ft

Range: 10 ft
100 ft

Must be within 10 ft of 0 ft.

This type of lock will lock the message until the receiver reaches a certain topological altitude. The range is how close the receiver needs to be to the selected altitude.



70 °

Range: 5 °
10 °

Must be within 0 ° of 70 °.

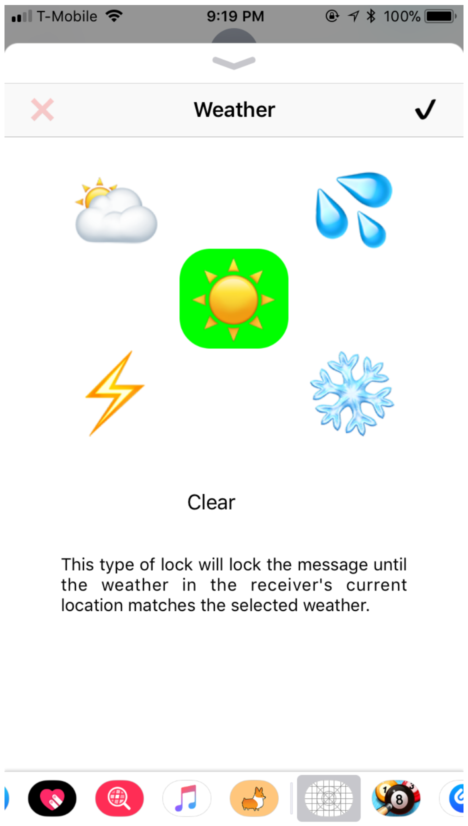
This type of lock will lock the message until the temperature in the receiver's current location matches the selected temperature. The range is how many degrees the receiver's temperature must be within.

Pictured are the Location, Altitude, and Temperature based locks being applied to a message. All three are similar.

The Location lock utilizes your current location to lock the message. The Altitude and Temperature locks lock the message based on a value you select.

However, all three locks utilize a range or radius that the user must be within in order to unlock the message.

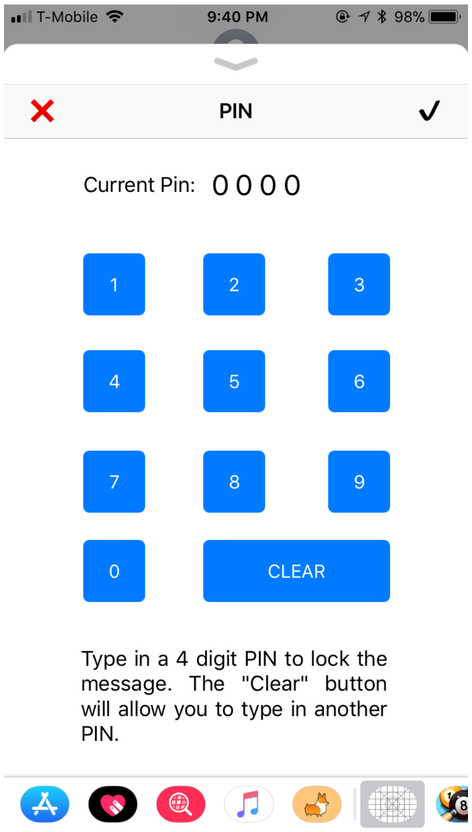
Notice for any lock type that ✗ and ✓ will remove a lock and add the current lock respectively.



Clear

This type of lock will lock the message until the weather in the receiver's current location matches the selected weather.

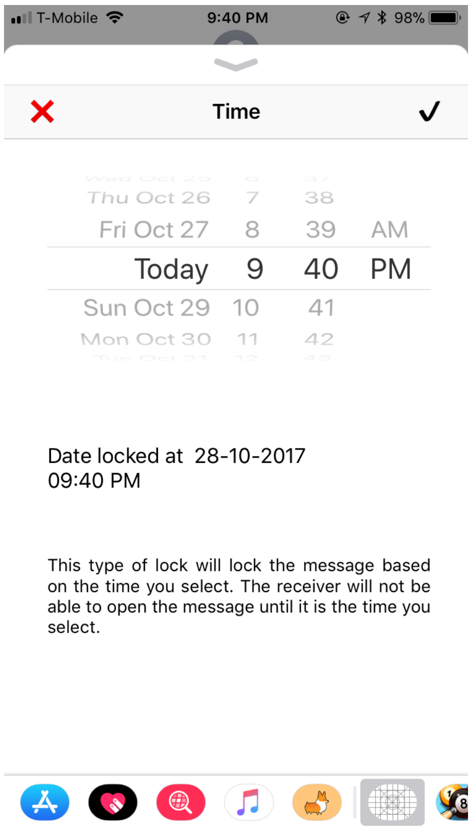
Tap a weather icon to change the lock for a weather lock.



Current Pin: 0 0 0 0

Type in a 4 digit PIN to lock the message. The "Clear" button will allow you to type in another PIN.

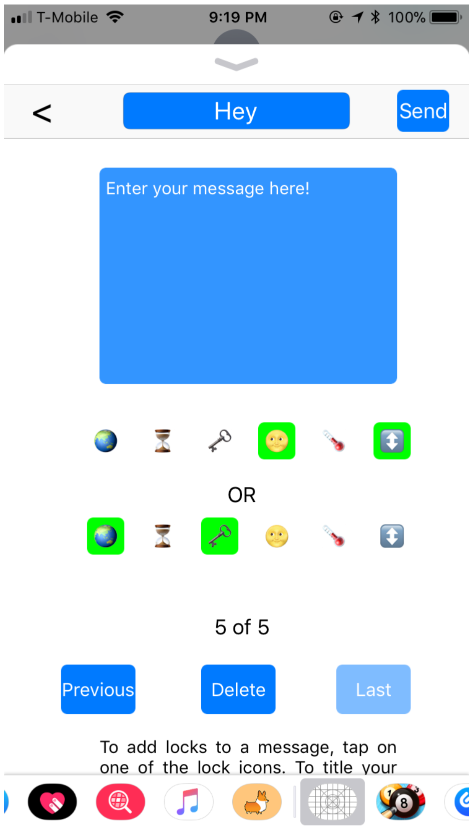
Enter in a 4 digit pin utilizing the keypad to set a PIN lock. The CLEAR button may be used to reset the current PIN.



Date locked at 28-10-2017
09:40 PM

This type of lock will lock the message based on the time you select. The receiver will not be able to open the message until it is the time you select.

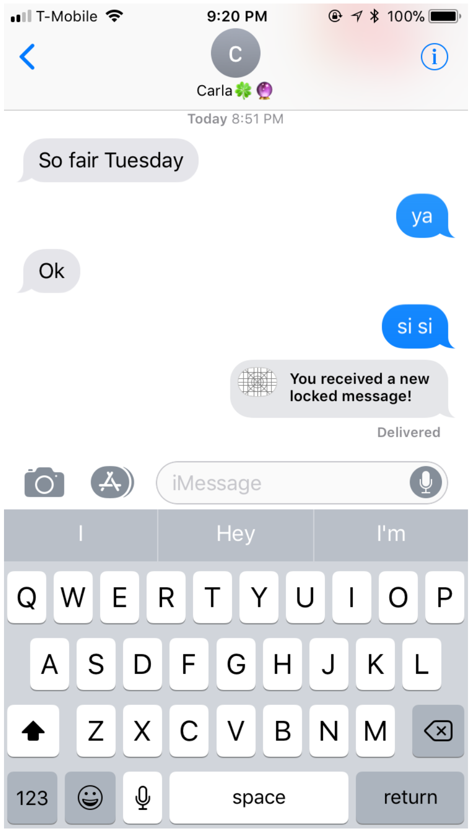
A time lock is set by simply selecting a date and a time with a calendar scroll-wheel.

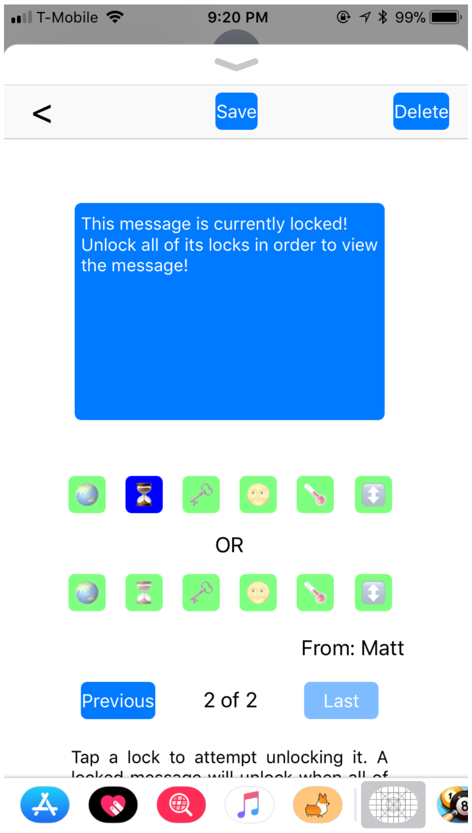


A multi-page message is packaged and sent using the “Send” button.

Notice that locks that are used are highlighted green.

The receiver may open the message by tapping the icon in the conversation.



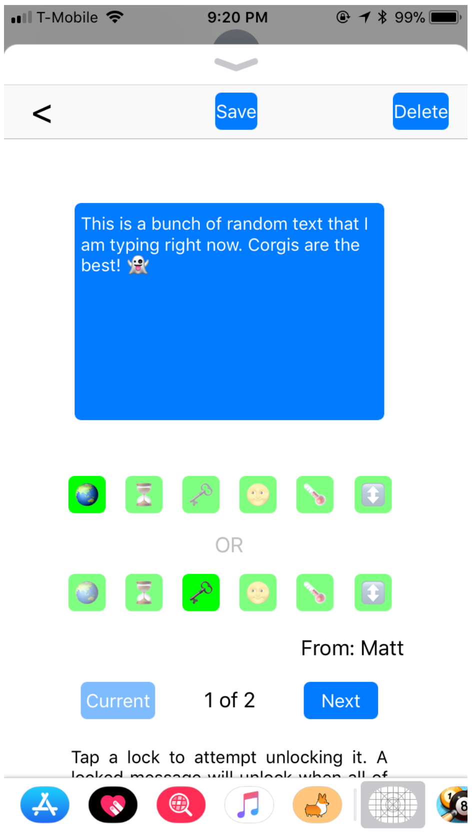

T-Mobile 9:20 PM 99%
<
Save
Delete
This message is currently locked! Unlock all of its locks in order to view the message!
OR
From: Matt
Previous
2 of 2
Last
Tap a lock to attempt unlocking it. A locked message will unlock when all of

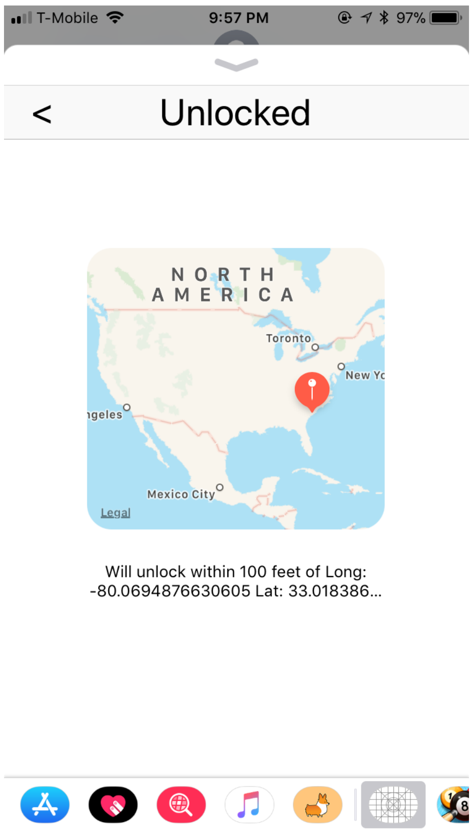
On the left, we have an example of a received message that is currently locked. Blue icons mean a page is locked.

On the right, we have a page that is unlocked. Notice that all the icons are green.

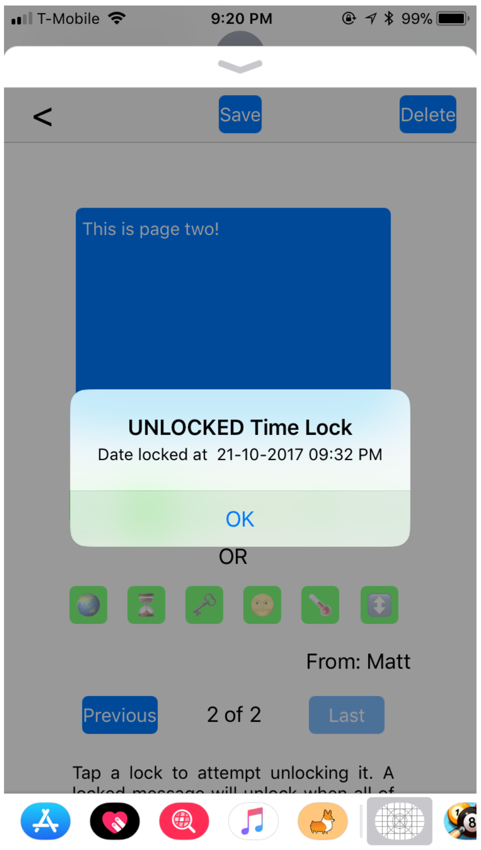
Also, there is a “Save” button because the message was loaded from the normal chest.

Pressing “Save” will transfer it to the saved chest and pressing “Delete” will permanently delete the message.


T-Mobile 9:20 PM 99%
<
Save
Delete
This is a bunch of random text that I am typing right now. Corgis are the best!
OR
From: Matt
Current
1 of 2
Next
Tap a lock to attempt unlocking it. A locked message will unlock when all of


Unlocked

Will unlock within 100 feet of Long: -80.0694876630605 Lat: 33.018386...


This is page two!

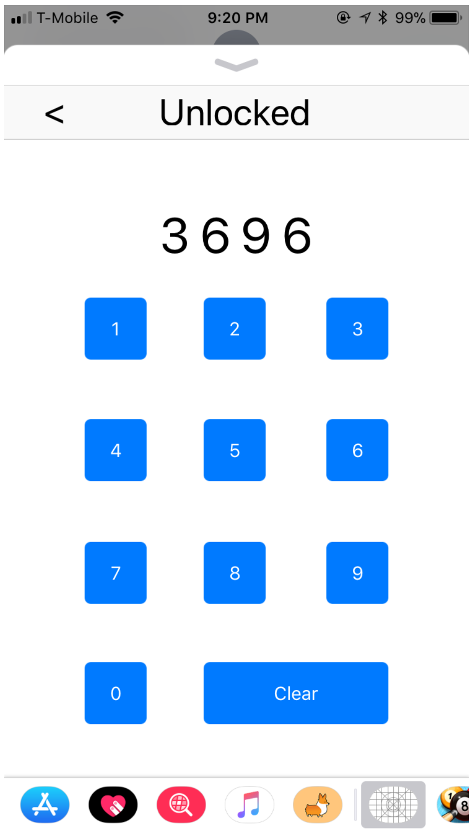
UNLOCKED Time Lock

Date locked at 21-10-2017 09:32 PM

Tap a lock to attempt unlocking it. A locked message will unlock when all of

Attempting to open a location lock will take you to a screen showing you your current location.

Attempting to open other locks (besides the PIN lock) will display a popup either confirming or denying that you have met the requirements for the lock and displaying the requirements.


Unlocked

3696

Unlocking a PIN is very similar to locking with a PIN. You enter in digits using the keypad. If the correct combination is entered, the top bar will display “Unlocked.”

Clear may be used to reset your current attempt at cracking the PIN.

6. Extended Application and Lock Descriptions

Locked Messages is an extension for the Messages Application on the iOS platform. It is compatible with both iPhones and iPads (it is designed with iPhones in mind). The application allows users that are using the application to send messages that are locked to each other. A message may be locked with a combination of up to six different types of locks. The receiver will only be able to view the content of a message when all of these locks are unlocked.

Location Lock: This type of lock is based off of the sender's current GPS location. The sender sets a radius of the current location that the receiver must be within for the message to unlock.

Time Lock: This type of lock will unlock when the time on the receiver's phone matches the time specified by the sender.

PIN Lock: This type of lock allows the sender to set a 4-digit PIN that the message will be locked with. The receiver will have to attempt to crack the PIN.

Weather Lock: The sender selects a type of weather (clear, cloudy, rainy, snowy, thundery). The weather in the receiver's current location must match the selected weather for the lock to unlock.

Temperature Lock: The sender selects a temperature value and a range that the receiver's temperature must be within in order to unlock the message.

Altitude Lock: The sender selects an altitude and a range that the receiver's topological altitude must be within in order to unlock the message.

A "Locked Message" consists of multiple pages. Each page contains a message and two sets of locks. One of the two sets of locks must all be unlocked in order to display the content of the message for a page. A message can be up to 5 pages in length.

The application has the ability to send and received locked messages. Received messages are stored locally on the device and are stored in "chests." There are two chests, one is a regular chest for received messages and the other is a special chest for messages specifically saved by the user. The saved chest has a storage limit of 50 messages while the regular chest has no such limit. Senders are unable to open their own messages, and a new message may not be added to the chest when a copy of it already exists in the chests.

7. Framework and API Documentation

The application is implemented on the iOS platform. It is implemented using Apple's Swift programming language. Apple's official The Swift Programming Language eBook was used to research the Swift language. The reader may obtain a copy of this document from the iTunes Store for free on their iDevice to learn more about Swift.

Several Frameworks and APIs were utilized in order to create this application:

> Messages Framework: This Framework was required as the application is an extension of the Messages application. The View Controller for the application inherits from the Messages View Controller. Also, the MSMessages class is utilized to package messages for sending and to unpackaged messages for receiving. This framework provides overall functionality for the application.
> **https://developer.apple.com/documentation/messages**
>
> Maps Framework: This Framework was required to access GPS features on the device. It is used to get the current GPS coordinates of a user. The longitude and latitude values are extracted from this coordinate. This framework provides functionality for location, weather, temperature, and altitude based locks.
> **https://developer.apple.com/maps/**
>
> OpenWeatherMap API: This API is utilized for its weather-based services. The user's current location is based to OpenWeatherMap in order to get the current weather type and the temperature in the user's location when necessary. A unique API key is necessary in order to access weather information from OpenWeatherMap.
> **https://openweathermap.org**

All other features of the application are implemented using built-in Swift features and custom classes. More extensive information about these classes is included in the comments of their code.

Before starting this project, I had no experience with both mobile app development and Swift. The Swift Programming Language proved to be a valuable resource for quickly learning about Swift. A majority of the time researching was spent figuring out which frameworks needed to be included in order to make features work. Time was also spent discovering which features would not be viable without utilizing 3rd party APIs.

8. Troubleshooting:

In order to initially run the application, one must enable it in the settings after installing from xCode. This can be done by going to **Settings -> General -> Device Management -> Trust**

Another potential issue is the application not showing up within Messages. To solve this, tap the **Settings Icon** within the Messages app list, then tap "**Edit**" and ensure that Locked Messages is enabled.