

# How to programmatically send SMS on the iPhone?

Asked 16 years, 4 months ago   Modified 4 years, 11 months ago

Viewed 308k times    Part of [Mobile Development](#) Collective



546

Does anybody know if it's possible, and how, to programmatically send a **SMS** from the `iPhone`, with the official SDK / Cocoa Touch?



MD

ios

objective-c

cocoa-touch

sms



Share

Improve this question

Follow

edited Dec 17, 2015 at 12:08



Nitin Gohel

49.7k ● 18 ● 106 ● 144

asked Aug 14, 2008 at 9:51



Marco

in ios 4 You can send sms with code but problem is your application will be closed – [Pankaj Kainthla](#) Feb 2, 2011 at 8:00

If using the official SDK is not a requirement for others, I wrote this post to show how to do this using Twilio: [twilio.com/blog/2016/11/...](http://twilio.com/blog/2016/11/...) – [Megan Speir](#) Feb 9, 2017 at 21:57

you can create a google spreadsheet off of your gmail, make the fields you want, then do `tools->script editor` and in a POST request use `MailApp.sendEmail` api to send email to a phone number. att's is

YOURNUMBER@mms.att.net, tmobile's is

YOURNUMBER@tmomail.net I think (ALL FREE)

– [Coty Embry](#) May 20, 2017 at 17:23

---

18 Answers

Sorted by:

Highest score (default)



## Restrictions

418



If you could send an SMS within a program on the iPhone, you'll be able to write games that spam people in the background. I'm sure you really want to have spams from your friends, "Try out this new game! It roxxers my boxxers, and yours will be too! roxxersboxxers.com!!!! If you sign up now you'll get 3,200 RB points!!"

Apple has restrictions for automated (or even partially automated) SMS and dialing operations. (Imagine if the game instead dialed 911 at a particular time of day)

Your best bet is to set up an intermediate server on the internet that uses an online SMS sending service and send the SMS via that route if you need complete automation. (ie, your program on the iPhone sends a UDP packet to your server, which sends the real SMS)

## iOS 4 Update

iOS 4, however, now provides a `viewController` you can import into your application. You prepopulate the SMS fields, then the user can initiate the SMS send within the controller. Unlike using the "SMS:..." url format, this allows your application to stay open, and allows you to populate both the *to* and the *body* fields. You can even specify multiple recipients.

This prevents applications from sending automated SMS without the user explicitly aware of it. You still cannot send fully automated SMS from the iPhone itself, it requires some user interaction. But this at least allows you to populate everything, and avoids closing the application.

The [MFMessageComposeViewController](#) class is well documented, and [tutorials](#) show how easy it is to implement.

## iOS 5 Update

iOS 5 includes messaging for iPod touch and iPad devices, so while I've not yet tested this myself, it may be that all iOS devices will be able to send SMS via MFMessageComposeViewController. If this is the case, then Apple is running an SMS server that sends messages on behalf of devices that don't have a cellular modem.

## iOS 6 Update

No changes to this class.

## **iOS 7 Update**

You can now check to see if the message medium you are using will accept a subject or attachments, and what kind of attachments it will accept. You can edit the subject and add attachments to the message, where the medium allows it.

## **iOS 8 Update**

No changes to this class.

## **iOS 9 Update**

No changes to this class.

## **iOS 10 Update**

No changes to this class.

## **iOS 11 Update**

[No significant changes to this class](#)

## **Limitations to this class**

Keep in mind that this won't work on phones without iOS 4, and it won't work on the iPod touch or the iPad, except, perhaps, under iOS 5. You must either detect the device

and iOS limitations prior to using this controller, or risk restricting your app to recently upgraded 3G, 3GS, and 4 iPhones.

However, an intermediate server that sends SMS will allow any and all of these iOS devices to send SMS as long as they have internet access, so it may still be a better solution for many applications. Alternately, use both, and only fall back to an online SMS service when the device doesn't support it.

Share Improve this answer

Follow

edited Dec 29, 2019 at 11:26



Cœur

38.6k ● 26 ● 202 ● 276

answered Sep 12, 2008 at 15:00



Adam Davis

93.5k ● 60 ● 271 ● 333

---

12 And if you purchase that domain I will never be able to look at you the same way again. – [Adam Davis](#) Sep 12, 2008 at 15:03

---

81 I think it's ironic that somebody flagged this post as spam. Read between the lines, peoples! – [Randolpho](#) Sep 18, 2009 at 14:49

---

2 -1 because this reads as speculation. Also, see Jus' Wondrin's and rydgaze's answers on how to send in-app SMSes. – [Frank Shearar](#) Sep 27, 2010 at 7:55

---

9 @Frank - Updated my writeup to reflect the new iOS 4 features. Removed wishy-washy wording. – [Adam Davis](#) Oct 9, 2010 at 16:00

---

16 I really appreciate that you keep this post updated for new iOS versions. :) – Ciryon Mar 14, 2016 at 8:18

---



Here is a tutorial which does exactly what you are looking for: the `MFMessageComposeViewController` .

146



<http://blog.mugunthkumar.com/coding/iphone-tutorial-how-to-send-in-app-sms/>



Essentially:



```
MFMessageComposeViewController *controller = [[[MFMessageComposeViewController alloc] init] autorelease];
if([MFMessageComposeViewController canSendText])
{
    controller.body = @"SMS message here";
    controller.recipients = [NSArray arrayWithObjects:
    controller.messageComposeDelegate = self;
    [self presentViewController:controller animated:
    }
}
```

And a link to the docs.

<https://developer.apple.com/documentation/messageui/mfmessagecomposeviewController>

Share Improve this answer

Follow

edited Jun 28, 2017 at 14:47



Cœur

38.6k ● 26 ● 202 ● 276

answered Jul 24, 2010 at 19:09



Daniel Amitay

6,667 ● 7 ● 37 ● 43

---

still, the SMS form has to pop up. Any way to send in background? – [Raptor](#) Aug 2, 2012 at 11:02

---

5 You can certainly send SMS in the background using a service such as Twilio, but if you want to send it from the user's phone number, they have to approve the message via the above method. – [Daniel Amitay](#) Aug 2, 2012 at 13:13 ✎

---

3 anyone using the above code may wish to consider putting the `MFMessageComposeViewController *controller = ...` inside the if block. (the class method does not need to have an instance to do the test) – [unsynchronized](#) Dec 13, 2013 at 4:12

---

The link

`http://blog.mugunthkumar.com/coding/iphone-tutorial-how-to-send-in-app-sms/` says "502 Bad Gateway" on my laptop. Maybe the link is broken.

– [Nikita Vlasenko](#) Nov 27, 2015 at 21:41

---



100



1. You must add the MessageUI.framework to your Xcode project
2. Include an `#import <MessageUI/MessageUI.h>` in your header file
3. Add these delegates to your header file  
`MFMessageComposeViewControllerDelegate` &  
`UINavigationControllerDelegate`
4. In your `IBAction` method declare instance of `MFMessageComposeViewController` say  
`messageInstance`

5. To check whether your device can send text use `[MFMessageComposeViewController canSendText]` in an if condition, it'll return Yes/No

6. In the `if` condition do these:

1. First set body for your `messageInstance` as:

```
messageInstance.body = @"Hello from Shah";
```

2. Then decide the recipients for the message as:

```
messageInstance.recipients = [NSArray arrayWithObjects:@"87654321", nil];
```

3. Set a delegate to your `messageInstance` as:

```
messageInstance.messageComposeDelegate = self;
```

4. In the last line do this:

```
[self presentViewController:messageInstar
```

Share Improve this answer

Follow

edited Jul 17, 2015 at 14:28



Alex Zavatone

4,293 ● 38 ● 55

answered Dec 7, 2011 at 16:50



Najeebullah Shah

3,709 ● 4 ● 37 ● 49

---

2 Version note: `presentModalViewController:animated:` is deprecated; use `presentViewController:animated:completion:` instead. Also, remember to define the delegate method -



```
(void)messageComposeViewController:
(MFMessageComposeViewController *)controller
didFinishWithResult:(MessageComposeResult)result
```

if you want to know the results. – [Raptor](#) Oct 30, 2015 at 3:41

Please put the code for test. @Najeebullah Shah here or in github. – user285594 Feb 9, 2017 at 7:23



49

You can use a `sms:[target phone number]` URL to open the SMS application, but there are no indications on how to prefill a SMS body with text.



Share Improve this answer

edited Jun 7, 2019 at 6:09

Follow



Cœur

38.6k ● 26 ● 202 ● 276

answered Sep 12, 2008 at 14:41



millenomi

6,589 ● 4 ● 33 ● 34

react-native-communications uses the `&body=` parameter to prefill the text when opening the `sms:targetphonenumber&body=textyouwanttoprefillwith` for iOS – [Doug Voss](#) Dec 21, 2017 at 20:01

I just tried this on my iphone 8, and it worked great... thanks! – [Brad Parks](#) Jan 22, 2019 at 16:15

I need to send an sms to a specific number but with predefined (prefill) body text. Thos body text should be the manual selection from 4 possible options. E.g. The app opens and pop-ups "select reason". Then 4 options are available. Next the user selects one of these 4 options and then an automatic SMS is sent having as body text the



25



One of the systems of inter-process communication in MacOS is XPC. This system layer has been developed for inter-process communication based on the transfer of plist structures using libSystem and launchd. In fact, it is an interface that allows managing processes via the exchange of such structures as dictionaries. Due to heredity, iOS 5 possesses this mechanism as well.

You might already understand what I mean by this introduction. Yep, there are system services in iOS that include tools for XPC communication. And I want to exemplify the work with a daemon for SMS sending. However, it should be mentioned that this ability is fixed in iOS 6, but is relevant for iOS 5.0—5.1.1. Jailbreak, Private Framework, and other illegal tools are not required for its exploitation. Only the set of header files from the directory `/usr/include/xpc/*` are needed.

One of the elements for SMS sending in iOS is the system service `com.apple.chatkit`, the tasks of which include generation, management, and sending of short text messages. For the ease of control, it has the publicly available communication port `com.apple.chatkit.clientcomposeserver.xpc`. Using the XPC subsystem, you can generate and send messages without user's approval.

Well, let's try to create a connection.

```
xpc_connection_t myConnection;

dispatch_queue_t queue =
dispatch_queue_create("com.apple.chatkit.clientcompose
DISPATCH_QUEUE_CONCURRENT);

myConnection =
xpc_connection_create_mach_service("com.apple.chatkit.
queue, XPC_CONNECTION_MACH_SERVICE_PRIVILEGED);
```

Now we have the XPC connection myConnection set to the service of SMS sending. However, XPC configuration provides for creation of suspended connections —we need to take one more step for the activation.

```
xpc_connection_set_event_handler(myConnection, ^(xpc_o
xpc_type_t xtype = xpc_get_type(event);
if(XPC_TYPE_ERROR == xtype)
{
NSLog(@"XPC sandbox connection error: %s\n", xpc_dicti
XPC_ERROR_KEY_DESCRIPTION));
}
// Always set an event handler. More on this later.

NSLog(@"Received a message event!");

});

xpc_connection_resume(myConnection);
```

The connection is activated. Right at this moment iOS 6 will display a message in the telephone log that this type of communication is forbidden. Now we need to generate a dictionary similar to xpc\_dictionary with the data required for the message sending.

```

NSArray *recipient = [NSArray arrayWithObjects:@"+7 (9

NSData *ser_rec = [NSPropertyListSerialization dataWith
format:200 options:0 error:NULL];

xpc_object_t mydict = xpc_dictionary_create(0, 0, 0);
xpc_dictionary_set_int64(mydict, "message-type", 0);
xpc_dictionary_set_data(mydict, "recipients", [ser_rec
length]);
xpc_dictionary_set_string(mydict, "text", "hello from

```

Little is left: send the message to the XPC port and make sure it is delivered.

```

xpc_connection_send_message(myConnection, mydict);
xpc_connection_send_barrier(myConnection, ^{
NSLog(@"The message has been successfully delivered");
});

```

That's all. SMS sent.

Share Improve this answer

Follow

edited Jul 17, 2015 at 14:28



Alex Zavatone

4,293 ● 38 ● 55

answered Oct 25, 2012 at 7:17



isox

539 ● 5 ● 9

- 
- 1 You shouldn't use XPC to send SMS. The App produced won't be approved by Apple. Use MessageUI Framework instead – [Raptor](#) Oct 30, 2015 at 3:33 ✎
-

If you are using this technique for app that you use for your organisation, and app is not going through app store, you have no worries. – [Martin Berger](#) Jun 10, 2019 at 10:48

Getting "XPC sandbox connection error: Connection invalid" on iOS 12 – [Martin Berger](#) Jul 10, 2019 at 12:02



Add the MessageUI.Framework and use the following code

24



```
#import <MessageUI/MessageUI.h>
```

And then:



```
if ([MFMessageComposeViewController canSendText]) {  
    MFMessageComposeViewController *messageComposer =  
    [[MFMessageComposeViewController alloc] init];  
    NSString *message = @"Your Message here";  
    [messageComposer setBody:message];  
    messageComposer.messageComposeDelegate = self;  
    [self presentViewController:messageComposer animated  
    ]  
}
```

and the delegate method -

```
- (void)messageComposeViewController:(MFMessageCompose  
*)controller  
    didFinishWithResult:(MessageComposeResult  
    [self dismissViewControllerAnimated:YES completi  
    ]  
}
```

Follow



Alex Zavatore

4,293 ● 38 ● 55

answered May 10, 2013 at 21:38



Bharat Gulati

796 ● 6 ● 12

---

hey thats great , but can we do this functionality from background? – [Raj](#) Apr 5, 2014 at 7:17

---

- 3 Apple will not allow you to send message without users approval. He has to send message/mail by manually pressing the button. Alternatively, you can use a custom service by sending emails/numbers to your backend and then sending. Though you cannot do this directly on iPhone – [Bharat Gulati](#) May 20, 2014 at 21:19
- 



You can use this approach:

21



```
[[UIApplication sharedApplication]openURL:[NSURL URLWithString:@"sms:MobileNumber"]]
```



iOS will automatically navigate from your app to the messages app's message composing page. Since the URL's scheme starts with sms:, this is identified as a type that is recognized by the messages app and launches it.

Share Improve this answer

edited Jul 20, 2015 at 19:43

Follow



Alex Zavatore

4,293 ● 38 ● 55

answered Feb 26, 2014 at 7:13



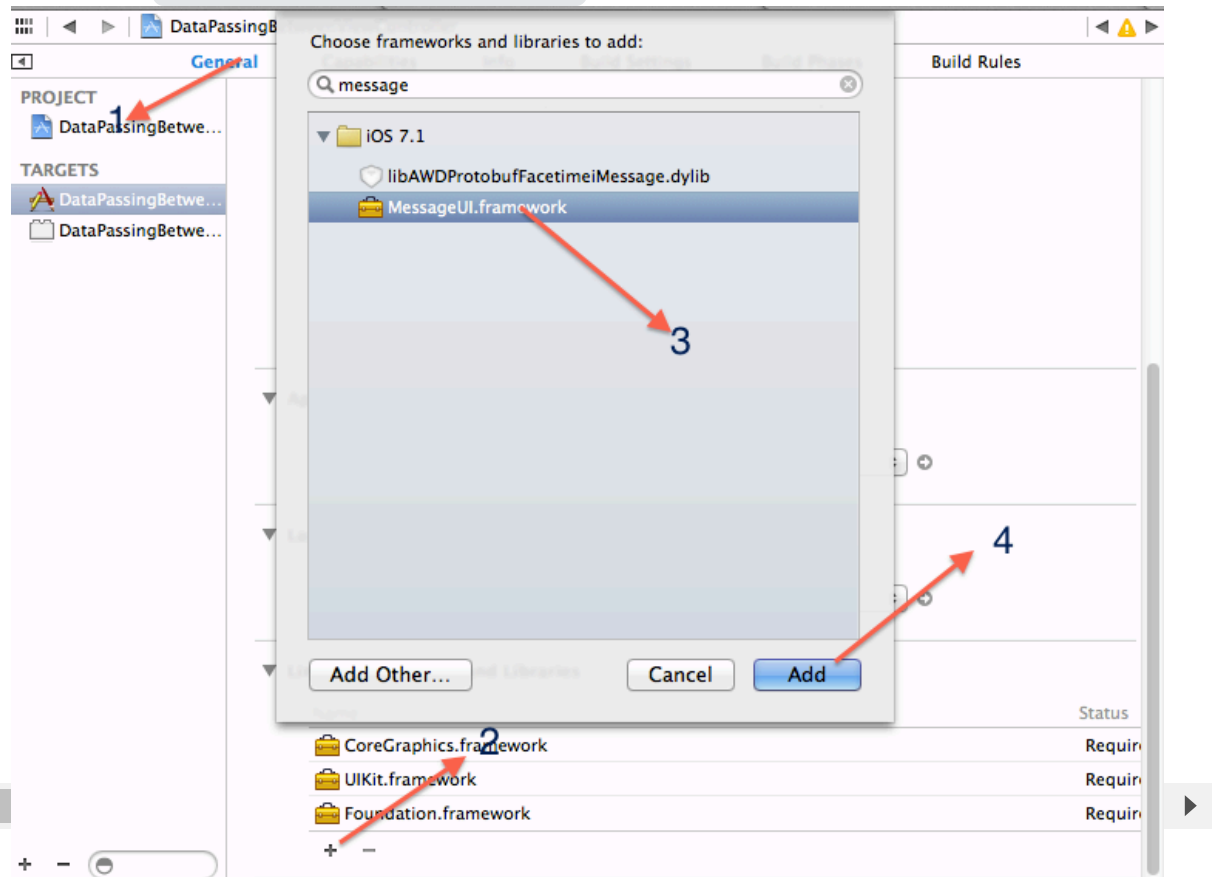
Dinesh Reddy Chennuru

491 ● 5 ● 10

## Follow this procedures

14

1 .Add `MessageUI.Framework` to project



2 . Import `#import <MessageUI/MessageUI.h>` in .h file.

3 . Copy this code for sending message

```
if ([MFMessageComposeViewController canSendText]) {  
    MFMessageComposeViewController *messageComposer =  
    [[MFMessageComposeViewController alloc] init];  
    NSString *message = @"Message!!!";  
    [messageComposer setBody:message];  
    messageComposer.messageComposeDelegate = self;  
}
```

```
[self presentViewController:messageComposer animat  
}
```

4 . Implement `delegate` method if you want to.

```
- (void)messageComposeViewController:(MFMessageCompose  
*)controller didFinishWithResult:(MessageComposeResult  
  
    ///your stuff here  
  
    [self dismissViewControllerAnimated:YES completion  
}
```

Run And GO!

Share Improve this answer

edited Nov 17, 2014 at 8:21

Follow


answered Jul 29, 2014 at 10:32



**Tunvir Rahman Tusher**

6,611 ● 2 ● 38 ● 33

---

worth mentioning that you'll probably want to run `[self  
dismissViewControllerAnimated:YES  
completion:nil];` inside  
`messageComposeViewController:  
didFinishWithResult:` callback method. Else it will just  
hang there. – [SaltyNuts](#) Nov 16, 2014 at 19:35 

---



```
//Add the Framework in .h file
```





```
#import <MessageUI/MessageUI.h>
#import <MessageUI/MFMailComposeViewController.h>

//Set the delegate methods

UIViewController<UINavigationControllerDelegate, MFMess

//add the below code in .m file

- (void)viewDidAppear:(BOOL)animated{
    [super viewDidAppear:animated];

    MFMessageComposeViewController *controller =
    [[[MFMessageComposeViewController alloc] init] aut

    if([MFMessageComposeViewController canSendText])
    {
        NSString *str= @"Hello";
        controller.body = str;
        controller.recipients = [NSArray arrayWithObje
                                @"", nil];
        controller.delegate = self;
        [self presentViewController:controller an

    }

}

- (void)messageComposeViewController:
(MFMessageComposeViewController *)controller
    didFinishWithResult:(MessageComposeRe

{
    switch (result)
    {
        case MessageComposeResultCancelled:
            NSLog(@"Cancelled");
            break;
        case MessageComposeResultFailed:
            NSLog(@"Failed");
            break;
        case MessageComposeResultSent:
            break;
        default:
```

```
        break;
    }
    [self dismissModalViewControllerAnimated:YES];
}
```

Share Improve this answer

answered Aug 27, 2012 at 12:33

Follow



Rushabh

3,203 ● 5 ● 30 ● 52

---

you should see this link: [blog.mugunthkumar.com/coding/...](http://blog.mugunthkumar.com/coding/...) it will help you – Banker Mittal Oct 4, 2012 at 13:03

---



6

Here is the Swift version of code to send SMS in iOS. Please noted that it only works in real devices. Code tested in iOS 7+. You can read more [here](#).



1) Create a new Class which inherits MFMessageComposeViewControllerDelegate and NSObject:



```
import Foundation
import MessageUI

class MessageComposer: NSObject, MFMessageComposeViewC
    // A wrapper function to indicate whether or not a
    from the user's device
    func canSendText() -> Bool {
        return MFMessageComposeViewController.canSendT
    }

    // Configures and returns a MFMessageComposeViewCo
    func configuredMessageComposeViewController(textMe
, textBody body:String) -> MFMessageComposeViewControll
```

```

        let messageComposeVC = MFMessageComposeViewCon
        messageComposeVC.messageComposeDelegate = self
        this property to self, so that the controller can be d
        messageComposeVC.recipients = textMessageRecip
        messageComposeVC.body = body
        return messageComposeVC
    }

    // MFMessageComposeViewControllerDelegate callback
    controller when the user is finished with it
    func messageComposeViewController(controller:
    MFMessageComposeViewController!, didFinishWithResult r
    MessageComposeResult) {
        controller.dismissViewControllerAnimated(true,
        }
    }

```

## 2) How to use this class:

```

func openMessageComposerHelper(sender:AnyObject ,withI
NSIndexPath) {
    var recipients = [String]()

    //modify your recipients here

    if (messageComposer.canSendText()) {
        println("can send text")
        // Obtain a configured MFMessageComposeViewCon
        let body = Utility.createInvitationMessageText

        let messageComposeVC =
        messageComposer.configuredMessageComposeViewController
        body)

        // Present the configured MFMessageComposeView
        // Note that the dismissal of the VC will be h
        messageComposer instance,
        // since it implements the appropriate delegat
        presentViewController(messageComposeVC, animat
        nil)
    } else {
        // Let the user know if his/her device isn't a

```

```
self.displayAlertViewWithTitle("Cannot Send Text Message")
{
    UIAlertView *alertView = [[UIAlertView alloc] initWithTitle:@"Cannot Send Text Message"
                                                             message:@"Your device is not able to send text messages."
                                                             delegate:nil
                                                             cancelButtonTitle:@"OK"
                                                             otherButtonTitles:nil];
    [alertView show];
}
```

Share Improve this answer

answered May 23, 2015 at 7:40

Follow



thomasdao

982 ● 12 ● 27



3



There is a class in iOS 4 which supports sending messages with body and recipients from your application. It works the same as sending mail. You can find the documentation here: [link text](#)

Share Improve this answer

edited Oct 28, 2011 at 20:11

Follow



logancautrell

8,772 ● 3 ● 40 ● 50

answered Sep 10, 2010 at 8:32



Pankaj Kainthla

2,479 ● 5 ● 31 ● 61



3



```
- (void)sendSMS:(NSString *)bodyOfMessage recipientList:(NSArray *)recipientList
{
    UIPasteboard *pasteboard = [UIPasteboard generalPasteboard];
    UIImage *ui = resultingImage;
    pasteboard.image = ui;
    [[UIApplication sharedApplication] openURL:[NSURL URLWithString:@"sms:"]];
}
```

Share Improve this answer

answered Sep 9, 2013 at 11:17

Follow



mandeep

384 ● 1 ● 4 ● 10

---

4 how useful are your parameters ? – [martinsurleweb](#) Mar 5, 2015 at 21:50

---

[[UIApplication sharedApplication] openURL:[NSURL URLWithString:@"sms:"]]; – [Amr Angry](#) Jul 16, 2017 at 13:23

---



3



//call method with name and number.

```
-(void)openMessageViewWithName:(NSString*)contactName
*)phone{

    CTTelephonyNetworkInfo *networkInfo=[[CTTelephonyNetwo

    CTCarrier *carrier=networkInfo.subscriberCellularProvi

    NSString *Countrycode = carrier.isoCountryCode;

    if ([Countrycode length]>0)        //Check If Sim Inserte
    {

        [self sendSMS:msg recipientList:[NSMutableArray ar
    }
    else
    {

        [AlertHelper showAlert:@"Message" withMessage:@"No

    }

}
```

//Method for sending message

```

- (void)sendSMS:(NSString *)bodyOfMessage recipientsList:(NSArray *)recipients{
    MFMessageComposeViewController *controller1 = [[MFMessageComposeViewController alloc] init] ;
    controller1 = [[MFMessageComposeViewController alloc] initWithText:bodyOfMessage recipients:recipients];
    if([MFMessageComposeViewController canSendText])
    {
        controller1.body = bodyOfMessage;
        controller1.recipients = recipients;
        controller1.messageComposeDelegate = self;
        [self presentViewController:controller1 animated:YES completion:nil];
    }
}

```

Share Improve this answer

edited Mar 11, 2015 at 5:21

Follow

answered Mar 10, 2015 at 12:48



**ALOK KUMAR**

745 ● 5 ● 7



2

If you want, you can use the private framework `CoreTelephony` which called `CTMessageCenter` class.

There are a few methods to send sms.



Share Improve this answer

edited May 1, 2013 at 9:22

Follow



**Paras Joshi**

20.5k ● 11 ● 59 ● 70

answered Jan 29, 2010 at 2:15



**new soul**

322 ● 1 ● 7



---

1 He specifically asked if this were possible using the official SDK. – [Quentamia](#) Jul 26, 2011 at 18:44

---

1 Can you provide more information about the private api? I have no problem to use private framework because I don't need to publish it to App Store. – [Bagusflyer](#) Oct 3, 2014 at 2:56

---



Use this:

1



```
- (void)showSMSPicker
{
    Class messageClass =
    (NSStringFromClass(@"MFMessageComposeViewController"))

    if (messageClass != nil) {
        // Check whether the current device is configured to send
        messages
        if ([messageClass canSendText]) {
            [self displaySMSComposerSheet];
        }
    }
}

- (void)messageComposeViewController:(MFMessageCompose
*)controller didFinishWithResult:(MessageComposeResult)result
{
    //feedbackMsg.hidden = NO;
    // Notifies users about errors associated with the
    switch (result)
    {
        case MessageComposeResultCancelled:
        {
            UIAlertView *alert1 = [[UIAlertView alloc]
            message:@"SMS sending canceled!!!" delegate:self cancelButtonTitle:@"OK", nil];
            [alert1 show];
            [alert1 release];
        }
    }
}
```

```

        // feedbackMsg.text = @"Result: SMS sending ca
        break;

        case MessageComposeResultSent:
        {
            UIAlertView *alert2 = [[UIAlertView alloc]
message:@"SMS sent!!!" delegate:self cancelButtonTitle
otherButtonTitles:@"OK", nil];
            [alert2 show];
            [alert2 release];
        }

        // feedbackMsg.text = @"Result: SMS sent";
        break;

        case MessageComposeResultFailed:
        {
            UIAlertView *alert3 = [[UIAlertView alloc]
message:@"SMS sending failed!!!" delegate:self cancelB
otherButtonTitles:@"OK", nil];
            [alert3 show];
            [alert3 release];
        }

        // feedbackMsg.text = @"Result: SMS sending fa
        break;

        default:
        {
            UIAlertView *alert4 = [[UIAlertView alloc]
message:@"SMS not sent!!!" delegate:self cancelButtonTitle
otherButtonTitles:@"OK", nil];
            [alert4 show];
            [alert4 release];
        }

        // feedbackMsg.text = @"Result: SMS not sent";
        break;
    }

    [self dismissModalViewControllerAnimated: YES];
}

```



Share Improve this answer

Follow

edited May 28, 2012 at 15:04



user577537

answered May 18, 2012 at 5:47



Rock

1,510 ● 1 ● 12 ● 28



```
[[UIApplication sharedApplication]openURL:[NSURL URLWithString:
```

1

This would be the best and short way to do it.



Share Improve this answer

Follow

edited Aug 4, 2014 at 12:38



brasofilo

26k ● 15 ● 93 ● 184

answered Aug 4, 2014 at 12:36



Rinku Sadhwani

86 ● 1 ● 4



1

You can present MFMessageComposeViewController, which can send SMS, but with user prompt(he taps send button). No way to do that without user permission. On iOS 11, you can make extension, that can be like filter for incoming messages , telling iOS either its spam or not. Nothing more with SMS cannot be done



Share Improve this answer

Follow

answered Nov 26, 2017 at 1:33



Максуд Даудов

521 ● 5 ● 17



0

You need to use the **MFMessageComposeViewController** if you want to show creating and sending the message in your own app.



Otherwise, you can use the **sharedApplication** method.



Share Improve this answer

edited Jul 17, 2015 at 14:28



Follow



Alex Zavatore

4,293 ● 38 ● 55

answered Mar 30, 2015 at 6:31



parvind

897 ● 10 ● 22



**Highly active question.** Earn 10 reputation (not counting the [association bonus](#)) in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.