**SpinSMS – Flexible Salesforce SMS Messaging**

Hello and welcome to SpinSMS, the premier solution for sending and receiving text messages from and to the force.com platform. This guide should serve as a quick walkthrough on how to install and configure SpinSMS for your Salesforce organization.

**How does it work?**

Ancient Chinese secret! Nah, just kidding. At its core, SpinSMS utilizes Twilio (a 3rd party telcom company) for the actual sending of the messages. To get them the data SpinSMS takes advantage of their REST API and sends message data via HTTP post requests. Basically, there is a trigger attached to the SpinSMS message object. Whenever a new object is created, the phone number and message field on that record are sent to Twilio who then uses that information to send an SMS message to the desired recipient.

For handling inbound messages, SpinSMS utilizes Salesforce sites to host a visualforce page that Twilio can send content to Twilio sends the inbound message data, the visualforce page eats the data, and turns it into an SMS message record. Also Twilio passes along some cool geography info about where the message came from, so we record that to. You can use that information to figure out where in the world users are texting you from.

**How many messages can I send?**

Good question. Twilio doesn’t have a limit, they just have a rate cap and queue. So if you send too many the messages get queued and delivered as fast as possible. SpinSMS is bulk safe, so it can handle imports and processes that cause a lot of activity. The only real limiting factor are the Salesforce governor limits. Salesforce enforces hard rules, with an iron fist about how much stuff you can do, and it is possible to butt up against those rules when sending lots of messages. In spring 2011 Salesforce dramatically reduced the limits (by 75%) making life a lot easier, and hitting the wall much less likely. The biggest potential snag is the limits on HTTP requests, as they are the most stringent, and required to send messages to Twilio. Also, SpinSMS uses one @future callout for each message sent. You organization is given 200 @future callouts per full user license. For now all the developers at SpinSMS can say is, if you hit the limits, try to find a way to adjust your workflow. If you think there is a bug, or that SpinSMS is not making the best use of governor limits, feel free to let us know. You can review the current governor limits here.

http://www.salesforce.com/us/developer/docs/apexcode/Content/apex\_gov\_limits.htm

**How can I make it send messages?**

SpinSMS is pretty flexible and there are a number of ways you can make it send messages. You can mix and match and use any approach that makes sense for your business. All of these methods are just different ways to do the same thing, which is create a SpinSMS message record. Once a message record is created, that causes the sending of the message. You can even come up with your own process, as long as a message record is created in the end, the SMS message will go out.

**I) Workflow rules via email services**

A common desire is to utilize workflow rules to send SMS messages. SpinSMS accomplishes this by use of email services. You configure a workflow rule that sends a special email template to an email service you configure that takes those messages and turns them into SpinSMS message objects. Once the object is created, the SMS message is queued for delivery. This of course requires that you configure an email service, and workflow rule, and also create the email template. Force.com and SpinSMS make all these steps pretty painless. You’ll be sending messages in no time.

**II) Via email by use of email services**

Like we mentioned before, SpinSMS comes with an email services handler built in. All you have to do is create the email services address, and link it to the SpinSMS handleInboundEmail method. Once you have that, you can just email the given address using the special email format and that will trigger an SMS message to be sent. Using this approach any process that can send email can also now send SMS messages. The email you send to the email services handler must contain the following XML. It can contain other data (though it will be discarded) but the following must be present.

<?xml version="1.0"?>   
<emailData>   
<phoneNumber>PHONE NUMBER</phoneNumber>   
<message>MESSAGE!</message>  
 </emailData>

Of course you can hard values there, or use Salesforce merge fields since this is going in an email template.

**III) Manual creation of SMS records**

If you only send a few SMS messages, or want them to be created via an import job or some other one off process you can simple create SpinSMS message records by hand. Once they are saved Spin will work its magic and the message will be on the way. Again, just make sure the emails that are sent use the special email data format. You can use static content, or Salesforce merge fields to populate the phone number and message data.

<?xml version="1.0"?>   
<emailData>   
<phoneNumber>PHONE NUMBER</phoneNumber>   
<message>MESSAGE!</message>  
 </emailData>

**IV) A custom button**

If you want to allow your users to send a pre-constructed SMS message by clicking a button or link SpinSMS lets you do that to. Simply create your custom button, set it to use onClick javaScript and past the following code.

{!REQUIRESCRIPT("/soap/ajax/10.0/connection.js")}

{!REQUIRESCRIPT("/soap/ajax/10.0/apex.js")}

var myvar = sforce.apex.execute("SpinSMS","triggerSendMessage", {targetPhone:"5555555555",message:"This is a test message"});

window.alert(myvar);

In this example the phone number and message are hard coded into the button and would not by dynamic. By instead using Salesforce merge fields {!object.field} you can pass in the phone number and message from information found on the page the button exists on. For example you could do

{!REQUIRESCRIPT("/soap/ajax/10.0/connection.js")}

{!REQUIRESCRIPT("/soap/ajax/10.0/apex.js")}

var myvar = sforce.apex.execute("SpinSMS","triggerSendMessage", {targetPhone:"{!contact.mobilePhone}",message:"Your name is {!contact.firstname} "});

window.alert(myvar);

Here we have passed in information found on the contact record into the message to be sent. Pretty nifty eh?

You can of course modify the script as you desire. For example you can remove the pop up alert by simply removing window.alert(myvar);   
line. The possibilities here are pretty endless.

**V) HTTP Post or Get request to Visualforce Page – Not available yet, Sorry!**

If you have a Salesforce site configured (which you probably do to handle inbound messages) you can also leverage that to send messages as well. This approach is handle for letting other web software trigger sending of SMS messages. Simply create an HTTP request pointed a publically available instance of the SpinSMSSendMessage visualforce page, and include the following arguments/fields.

targetPhone – a comma separated list of phone numbers to send the included message to.  
message – a text message to send.  
token – the integration Token as specified on the active SpinSMS config object.