

Message Recall: Deployment Guide

Last Updated: Fri Mar 28, 2014

CONTENTS

[Overview](#)

- [1. Create a Google Apps Domain Super-Admin 'Role' account](#)
- [2. Create the app engine hosting environment](#)
 - [2.1 Create the application in app engine](#)
 - [2.2 Enable billing for your application](#)
- [3. Configure the Google APIs Console Project](#)
 - [3.1 Create the Project](#)
 - [3.2 Enable the Admin SDK API](#)
 - [3.3 Create a Project Service Account](#)
- [4. Configure your Google Apps Domain for the app engine application](#)
 - [4.1 Add the app engine application to the domain](#)
 - [4.2 Authorize your application in the Google Apps Domain](#)
- [5. Prepare the source code](#)
 - [5.1 Get the source code](#)
 - [5.2 Make local updates](#)
 - [5.3 Create a service account certificate .pem file](#)
- [6. Download the Google App Engine SDK](#)
- [7. Upload the source code into the AppEngine host](#)
- [8. Test the running application](#)

[Appendices](#)

[Getting Help](#)

Overview

Message Recall is a Google-AppEngine-hosted application designed to run in a **single Google Apps domain**. The following instructions describe a detailed process for deploying the application:

1. Create a Google Apps Domain Super-Admin 'Role' account
2. Create the App Engine hosting environment
3. Configure the Google APIs Console Project
4. Configure your Google Apps Domain for the App Engine application
5. Prepare the source code
6. Download the Google App Engine SDK
7. Upload the source code into the App Engine host
8. Test the running application

The end to end process should take about 30 minutes (not including study time to learn about App Engine or applications).

NOTE: We will show examples using a Google Apps Domain named gappslabs.com.

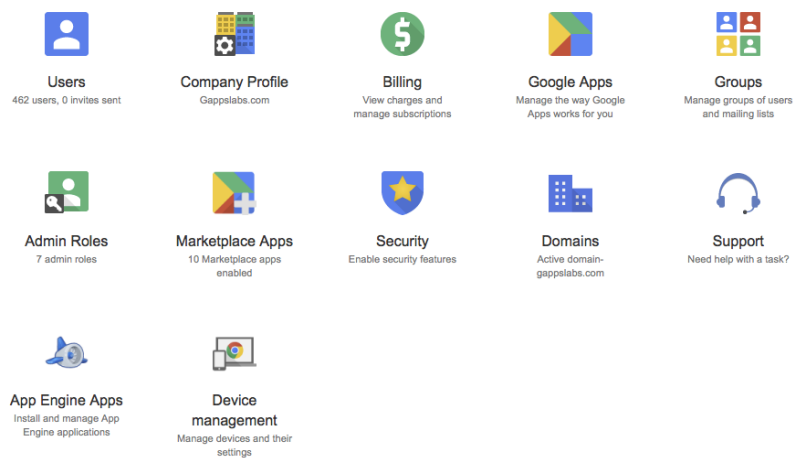
1. Create a Google Apps Domain Super-Admin 'Role' account

For additional reference, the following articles from the Google Support center discuss accounts and Admin roles:

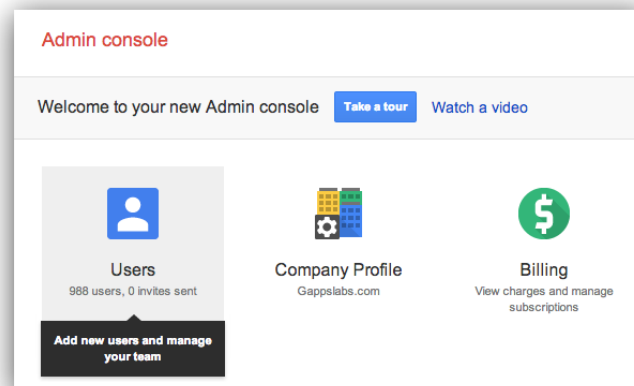
- [Adding users individually](#)
- [Pre-built administrator roles](#)
- [Assigning Administrator roles to users](#)


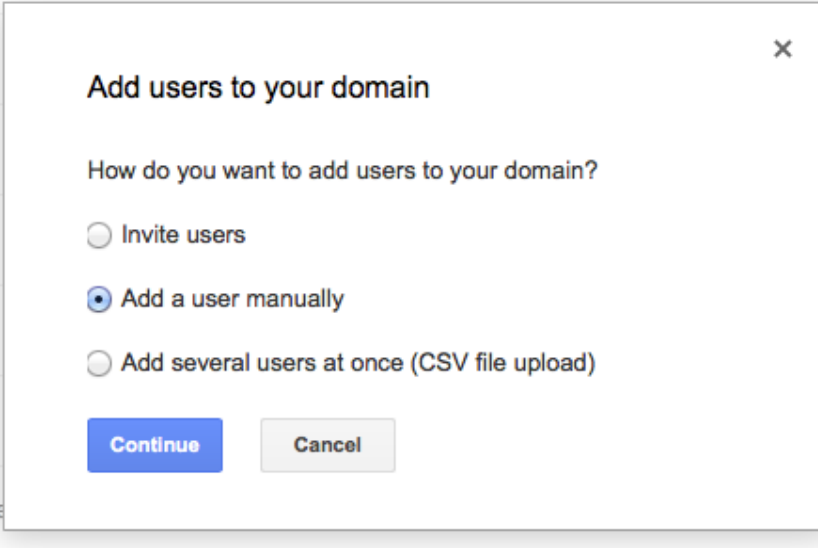
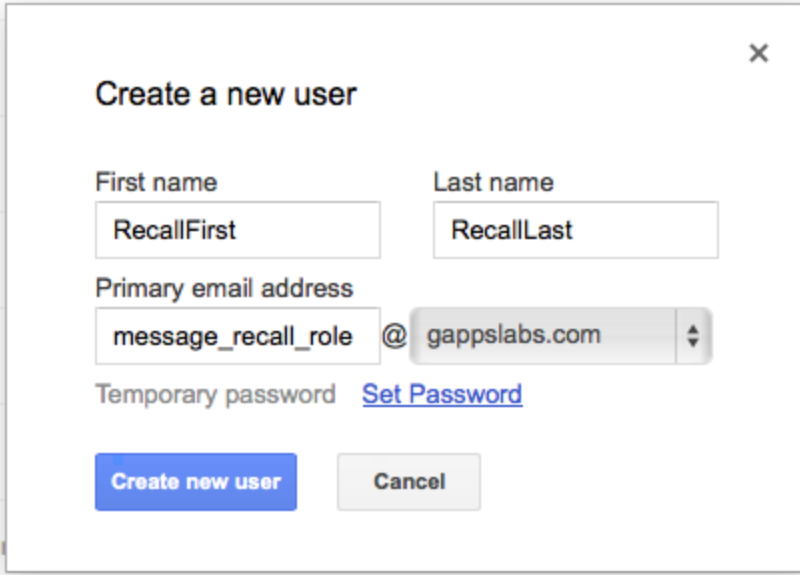
In your Google Apps Domain, while logged in as a Domain Admin, navigate to the Admin console at: <https://admin.google.com>.

Admin console



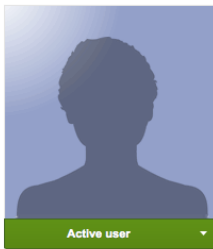
Select 'Users' to create the Role account.



<p>Add a new user by clicking 'Add more users' on the top-right toolbar.</p>	
<p>Select: 'Add a user manually' and choose 'Continue'.</p>	
<p>Enter 'First name', 'Last name' and 'Primary email address' then choose 'Create new user'.</p> <p>On the 'Getting started instructions' note the password generated for the new account.</p>	

Edit the new user, choose
'Show more' details.

← Users > RecallFirst RecallLast



gappslabs.com

RecallFirst RecallLast

message_recall_role@gappslabs.com

🕒 1:09 PM PST 📄 0 🗂️ 0 GB

Profile

Edit this user's profile, reset this user's password, add an alternative email address that points to an existing user account and more.

7 Google Apps enabled

View all Google apps and services enabled for this user.

0 Groups

Review this user's current group memberships and manage memberships.

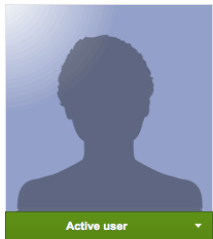
1 License

View the user's assigned licenses and estimated monthly charges.

Show more

Choose:
'Admin roles and its
privileges'.

← Users > RecallFirst RecallLast



gappslabs.com

RecallFirst RecallLast

message_recall_role@gappslabs.com

🕒 1:09 PM PST 📄 0 🗂️ 0 GB

Profile

Edit this user's profile, reset this user's password, add an alternative email address that points to an existing user account and more.

7 Google Apps enabled

View all Google apps and services enabled for this user.

0 Groups

Review this user's current group memberships and manage memberships.

1 License

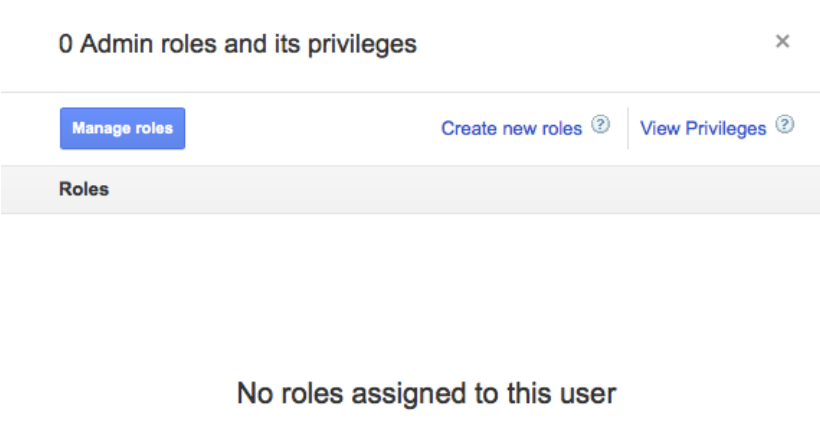
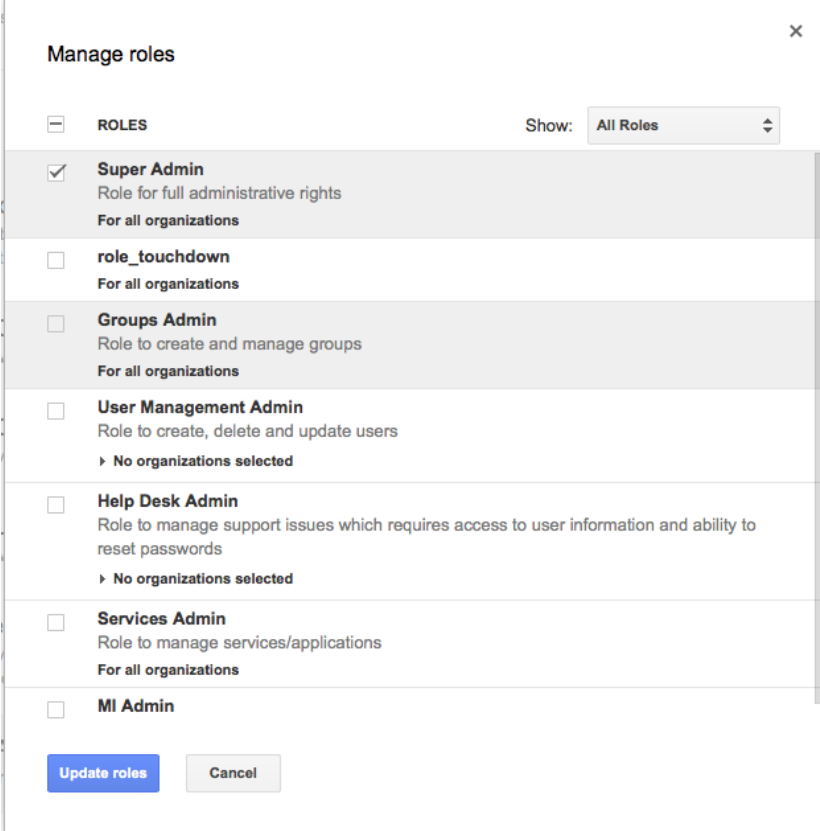
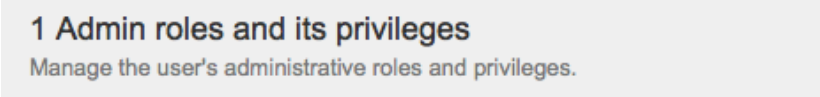
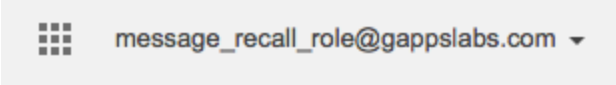
View the user's assigned licenses and estimated monthly charges.

Security

Review the user's 2-step verification enrollment, password strength, authorized access from third-party apps, and other security settings.

0 Admin roles and its privileges


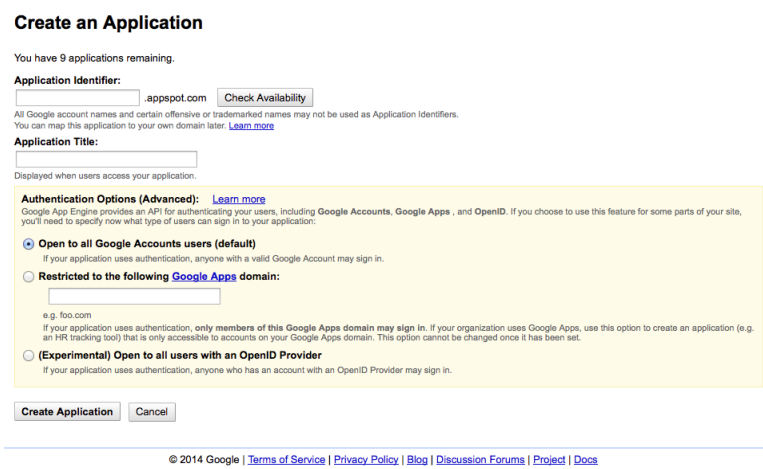
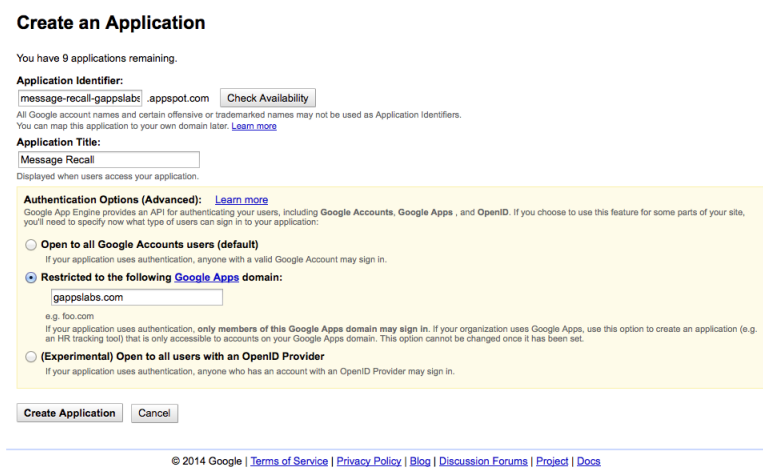
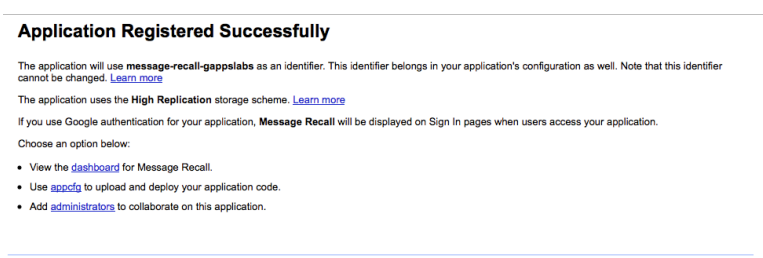
Manage the user's administrative roles and privileges.

<p>Choose: 'Manage roles'.</p>	
<p>Select: 'Super Admin' and choose 'Update roles'.</p>	
<p>You should notice your user now has '1' next to 'Admin roles and its privileges'.</p>	
<p>Sign out. Sign back in as</p>	

message_recall_role @yourdomain.com.	
---	--

2. Create the App Engine hosting environment

2.1 Create the application in App Engine

<p>While logged in as message_recall_role, navigate to 'Google App Engine' at https://appengine.google.com.</p>	
<p>Choose 'Create Application'.</p> <p>(You may be requested to verify your account via text message or voice call.)</p> <p>You should see the: 'Create an Application' page.</p>	
<p>Fill out the new application form.</p> <p>An Application Identifier of 'message-recall-yourdomain' should be fine.</p> <p>Select 'Restricted to the following 'Google Apps domain' and enter your domain.</p> <p>Select 'I accept these terms.'</p> <p>Choose 'Create Application'.</p>	
<p>You should see: 'Application Registered Successfully'.</p>	

2.2 Enable billing for your application

This application uses the IMAP mail API which uses sockets. To use sockets we must 'Enable Billing' in the application. Our use of the sockets is very small so any actual billing is very small.

However, Message Recall's use of backends is not free. It is **highly recommended** that you track your Billing Status on the application dashboard frequently. Running out of daily billing quota will cause your application to show a **OverQuotaError**.

In the app engine application, choose: 'Billing Status'.	Billing Billing Status Usage History Transaction History Billing Profile Billing Settings
Now choose 'Enable Billing'. You will need to establish a payment instrument (credit card) with a billing address. The default budgets are fine to start.	Billing Status: Free Enable billing if your application needs resources beyond the free quotas. Learn more <input type="button" value="Enable Billing"/> Transfer app to a premier account
You will need to approve the billing setup by clicking a link in the email inbox you use when you setup the payment instrument. The subject of the email should be 'Google Billing: Verify your email address' Click the link in the message body to verify your email.	Google Billing: Verify your email address

3. Configure the Google APIs Console Project

3.1 Create the Project

From the 'Application Registered Successfully' page, click the 'dashboard' link.

Application Registered Successfully

The application will use **message-recall-gapplabs** as an identifier. This identifier belongs in your application's configuration as well. Note that this identifier cannot be changed. [Learn more](#)

The application uses the **High Replication** storage scheme. [Learn more](#)

If you use Google authentication for your application, **Message Recall** will be displayed on Sign In pages when users access your application.

Choose an option below:

- View the [dashboard](#) for Message Recall.
- Use [appcfg](#) to upload and deploy your application code.
- Add [administrators](#) to collaborate on this application.

© 2014 Google | [Terms of Service](#) | [Privacy Policy](#) | [Blog](#) | [Discussion Forums](#) | [Project](#) | [Dota](#)

Click 'Application Settings'

Application: message-recall-gapplabs [High Replication]

[Community Support](#) « [My Applications](#)

Main

[Dashboard](#)

[Instances](#)

[Logs](#)

[Versions](#)

[Cron Jobs](#)

[Task Queues](#)

[Quota Details](#)

Data

[Datastore Indexes](#)

[Datastore Viewer](#)

[Datastore Statistics](#)

[Blob Viewer](#)

[Prospective Search](#)

[Text Search](#)

[Datastore Admin](#)

[Memcache Viewer](#)

Administration

[Application Settings](#)

[Permissions](#)

[Blacklist](#)

[Admin Logs](#)

ⓘ You need to upload and deploy an application before you can make Web history.

Read about using [appcfg](#) to upload and deploy one.

Charts ⓘ

No Data Available

Instances ⓘ

App Engine Release	Total number of instances	Average QPS*	Average Latency*	Average Memory
Billing Status: Free - Settings				
Quotas reset every 24 hours. Next reset: 4 hrs ⓘ				
Resource	Usage			

You can notice that a 'Google APIs Console Project Number' has been assigned. This is where an application 'registers' its intent to use Google APIs.

Click the project number (ours is 1035314471864) to bring up the 'Google Developers Console'.

Main

[Dashboard](#)

[Instances](#)

[Logs](#)

[Versions](#)

[Cron Jobs](#)

[Task Queues](#)

[Quota Details](#)

Data

[Datastore Indexes](#)

[Datastore Viewer](#)

[Datastore Statistics](#)

[Blob Viewer](#)

[Prospective Search](#)

[Text Search](#)

[Datastore Admin](#)

[Memcache Viewer](#)

Administration

[Application Settings](#)

[Permissions](#)

[Blacklist](#)

Basics

Application Title:

Message Recall

Displayed if users authenticate to use your application.

Application Identifier:

message-recall-gapplabs

Use this identifier in the application's app.yaml or appengine-web.xml.

Service Account Name:

message-recall-gapplabs@appspot.gserviceaccount.com

Use this name when interacting with external services on behalf of your application.

Application Default Version URL:

No version of application is deployed yet.

Application Identifier Alias:

message-recall-gapplabs.appspot.com

Between 6 and 30 characters. Provides an alternative URL to access your application through [appspot.com](#). It can be used to enable Channel, XMPP, Email, and SSL access for your application.

[http://message-recall-gapplabs.appspot.com](#)

Datastore Replication Options:

High Replication

Uses a highly replicated Datastore that synchronously replicates data across multiple locations simultaneously.

Google APIs Console Project Number:

1035314471864

The Google APIs Console allows you to configure other Google APIs, like Translate or Cloud Storage.

Cookie Expiration:

Default (1 Day) ⓘ

App Engine uses a cookie to keep users logged in to your application.

Authentication Type:

Google Apps domain ⓘ

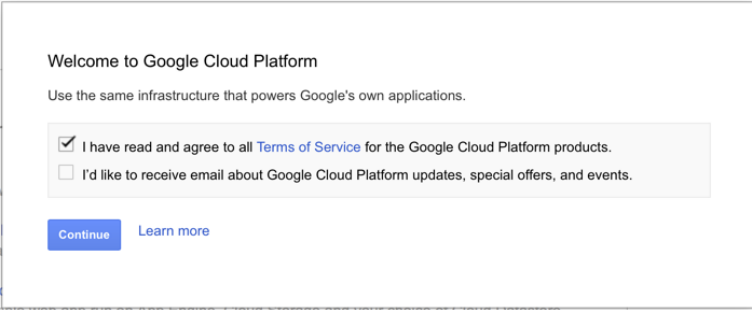
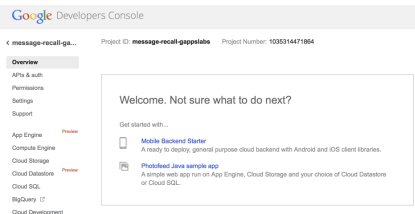
Users with a valid account in the selected Google Apps domain can sign in if the [Google Accounts API](#) is used for authentication. [Learn more](#)

Authentication Domain:

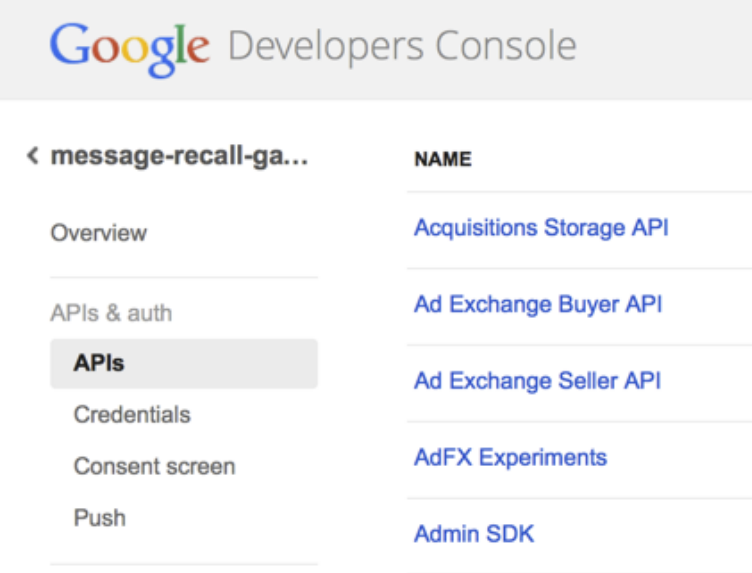
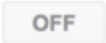
gapplabs.com

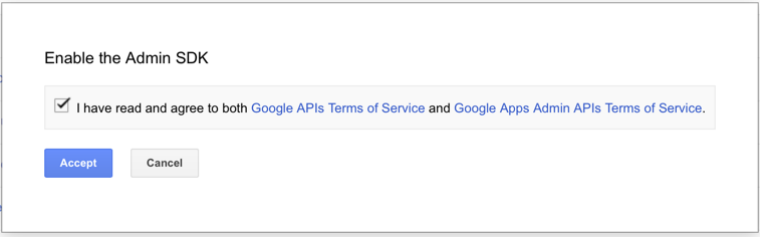

WARNING: Changing this will impact who has permission to access your app, including opening it up to users who previously did not have access. Verify that this is correct before submitting.

This may only be edited by apps that have no user properties in their datastore.

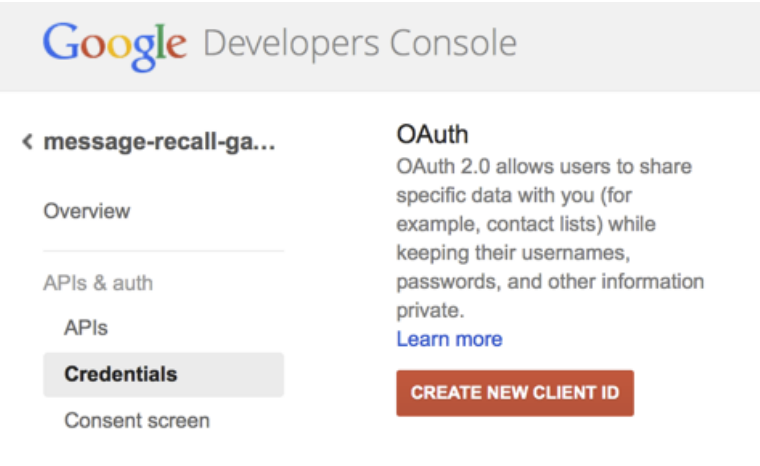
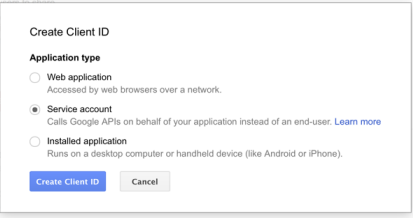
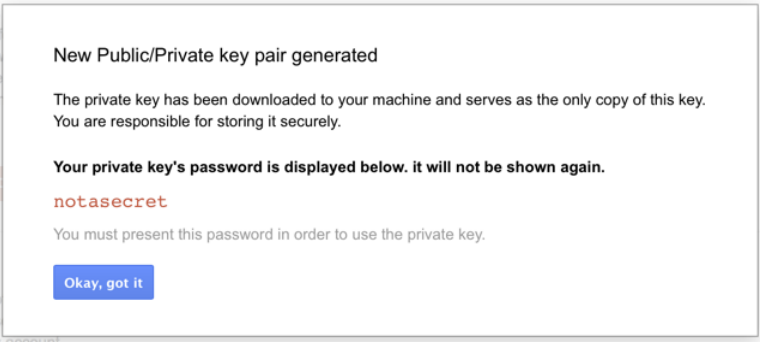
<p>This is creating a new project. Select to agree to the 'Terms of Service' and choose 'Continue'.</p>	
<p>You now have an Google API Console Project (Developers Console) to work with your app engine application.</p>	

3.2 Enable the Admin SDK API

<p>In the Developer Console for your project, choose: 'APIs & auth'.</p> <p>This will drop you into the APIs panel.</p>	
<p>We need to enable the 'Admin SDK' API.</p> <p>Choose the gray 'OFF' button on the 'Admin SDK' line to enable it.</p>	

<p>To enable the 'Admin SDK', select to accept the 'Terms of Service' and choose 'Accept'.</p>	
<p>The 'Admin SDK' should now be enabled 'ON'.</p>	

3.3 Create a Project Service Account

<p>Choose 'Credentials' from the API Console Project.</p> <p>Then, choose 'CREATE NEW CLIENT ID' to create your new OAuth 2.0 API application client.</p>	
<p>Select 'Service account' and then choose 'Create Client ID'.</p>	
<p>A file is generated and downloaded to your local system with your private key material. The file has a name of the form 'XXXX-privatekey.p12'.</p> <p>Choose 'Okay, got it' to proceed.</p>	

Your OAuth 2.0 client information is now displayed in the Developer Console.

Take special note of this 'Service Account Client ID' as it will be needed later.

Take special note of this 'Service Account Email address' as it will be needed later.

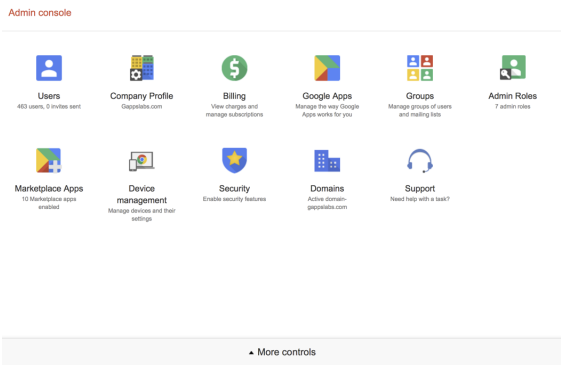
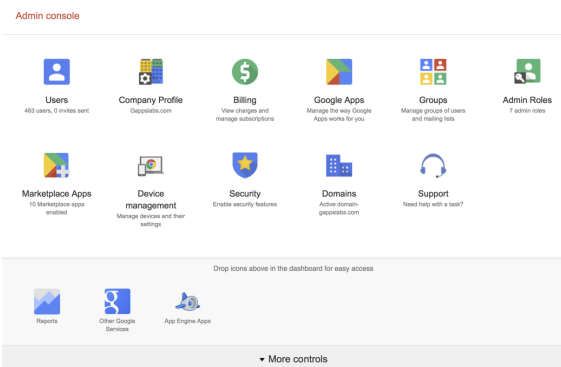
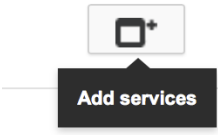
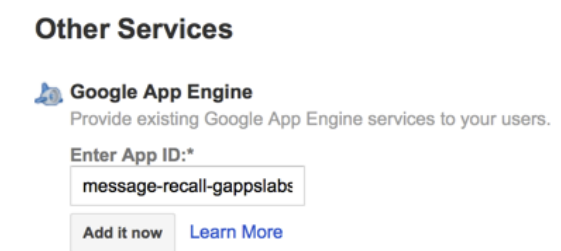
Service Account

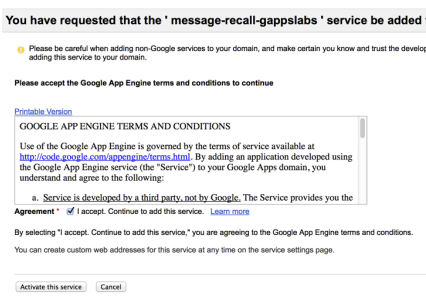
Client ID	1035314471864.apps.googleusercontent.com
Email address	1035314471864@developer.gserviceaccount.com
Public key fingerprints	a8dd2b654abda97c68d69d9060f9a4899900396

[Generate new key](#)[Download JSON](#)[Delete](#)

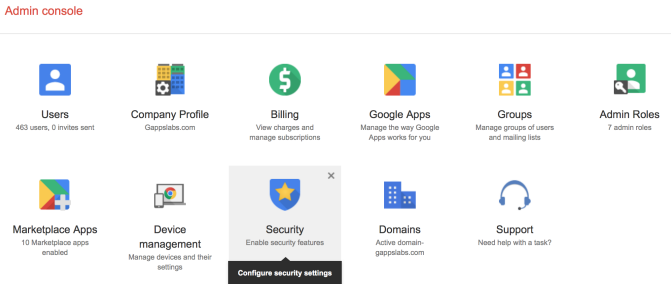
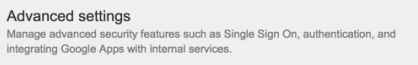
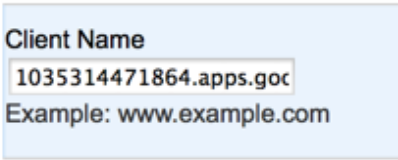
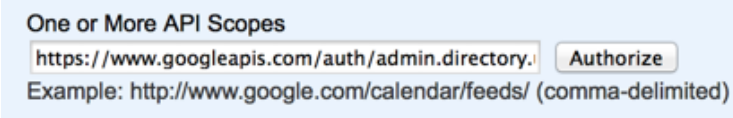
4. Configure your Google Apps Domain for the App Engine application

4.1 Add the App Engine application to the domain

<p>Navigate back to the 'Admin console' for your domain at https://admin.google.com.</p> <p>Choose 'More controls' at the bottom to view 'App Engine Apps'.</p>	
<p>Choose 'App Engine Apps' to view a list of registered applications.</p>	
<p>Click the 'Add services' button on the top-right.</p>	
<p>In 'Other Services', enter the name of the app engine application created: 'message-recall-yourdomain'.</p> <p>Choose 'Add It now'.</p>	

<p>Select the box to activate the service 'Agreement'.</p> <p>Choose 'Activate this service'.</p>	
---	--

4.2 Authorize your application in the Google Apps Domain


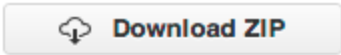
<p>Choose 'Security'</p>	
<p>Choose 'Advanced settings'</p>	
<p>Under 'Authentication' choose 'Manage OAuth Client access'.</p>	<p>Manage third party OAuth Client access Allows you to control access to user data by third party applications that use the OAuth protocol.</p>
<p>In the 'Client Name' field, enter the 'Service Account Client ID' from step 3.3 earlier.</p> <p>It should be of the form: ##.apps.googleusercontent.com</p>	
<p>In the 'One or More API Scopes' box, enter the following scopes string:</p> <p>https://www.googleapis.com/auth/admin.directory.user.readonly,https://mail.google.com/</p> <p>Choose 'Authorize'</p>	

You should notice an entry for your service account now listed.

Email (Read/Write/Send) <https://mail.google.com/>
<https://www.googleapis.com/auth/admin.directory.user.readonly>

5. Prepare the source code

5.1 Get the source code

Browse to the github.com page for googleapps-message-recall .	 google / googleapps-message-recall
Click the 'Download ZIP' button.	
Create an empty folder for the project source code. Copy the downloaded googleapps-message-recall-master.zip file to the empty folder. Unzip the project source code.	<pre>\$ unzip ./googleapps-message-recall-master.zip</pre>
Notice the source files under a new folder: googleapps-message-recall-master.	<pre>\$ ls -l ./googleapps-message-recall-master/ -rw-r--r--@ 1 adminuser xxxx 1697 Mar 18 11:29 README.md drwxr-xr-x@ 28 adminuser xxxx 952 Mar 18 11:29 message_recall</pre>

5.2 Make local updates

In the extracted source, edit the app.yaml file. Change the 'application' from 'message_recall' to your app engine 'Application Identifier'. Save your new app.yaml file.	From: application: message-recall To: application: message-recall-gapplabs
In the extracted source, edit the service_account.py file. Change the SERVICE_ACCOUNT_NAME to reflect the 'Service Account Email address' discovered in step 3.3.	From: SERVICE_ACCOUNT_NAME = ('000000000000-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx' '@developer.gserviceaccount.com') To: <your Service Account Email address>

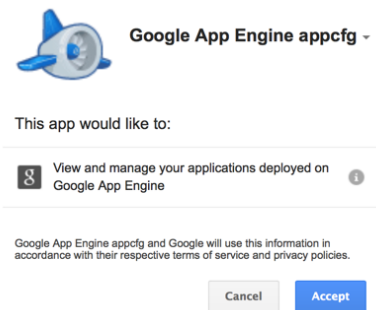
5.3 Create a service account certificate .pem file

Find the file downloaded 'XXXX-privatekey.p12' file and copy it to your application folder.	googleapps-message-recall-master/message_recall/
Use the openssl tool to convert the PKCS12 certificate file to PEM format. Note: openssl is available on Linux and Windows Note the output filename of 'messagerecall_privatekey.pem' is required.	<pre>\$ openssl pkcs12 -in xxxx-privatekey.p12 -out messagerecall_privatekey.pem -nodes -nocerts</pre>
When prompted for a password, supply the one presented earlier: <i>notasecret</i> .	Enter the password: notasecret
You should see the following output response and a new file created: messagerecall_privatekey.pem.	MAC verified OK
You can now remove the privatekey.p12 file.	<pre>\$ rm xxxx-privatekey.p12</pre>
Edit the new messagerecall_privatekey.pem file. Delete the Bag Attributes lines. Delete the Key Attributes line. The .pem file should begin with the 'BEGIN PRIVATE KEY' line and end with the 'END PRIVATE KEY' line.	<pre>Bag Attributes friendlyName: privatekey localKeyID: 54 69 6D 65 20 31 33 38 31 37 37 36 30 34 33 31 37 34 Key Attributes: <No Attributes> -----BEGIN PRIVATE KEY----- MIICdwIB..... -----END PRIVATE KEY-----</pre>
The edited file must be placed in the root source folder next to the app.yaml file.	googleapps-message-recall-master/message_recall/

6. Download the Google App Engine SDK

Find your platform, download and install the 'Google App Engine SDK for Python'	https://developers.google.com/appengine/downloads#Google_App_Engine_SDK_for_Python
Your installation process may be a single unzip or an executable installation program.	

7. Upload the source code into the AppEngine host

<p>Upload the front-end code.</p> <p>From your application source root folder (the folder that contains app.yaml), run the appcfg.py command that was installed with the 'Google App Engine SDK for Python'.</p> <p>Authenticate as your message_recall_role@yourdomain.com user.</p> <p>Note: The --oauth2 flag will use your credentials. If you wish to use the credentials of an account created previously, run appcfg.py without the --oauth2 flag and it will prompt you for an email address and password.</p>	<pre>\$ appcfg.py --oauth2 update .</pre>
<p>When prompted, choose 'Accept' to allow the appcfg program to upload the source code to your app engine hosting environment.</p>	
<p>You should notice the following progress on your console confirming the initial application upload (deployment).</p>	<pre>Compilation starting. Compilation completed. Starting deployment. Checking if deployment succeeded. Deployment successful. Checking if updated app version is serving. Completed update of app: message-recall-gapplabs, version: 1 Uploading index definitions. Uploading task queue entries.</pre>
<p>Now upload the back-end code.</p>	<pre>\$ appcfg.py --oauth2 backends . update</pre>

You should notice the following progress on your console confirming the deployment.

```
Starting update of app: message-recall-gapplabs,  
backend: recall-backend  
Getting current resource limits.  
Scanning files on local disk.  
...  
Compilation starting.  
Compilation completed.  
Starting deployment.  
Checking if deployment succeeded.  
Deployment successful.  
Checking if updated app version is serving.  
Completed update of app:  
    message-recall-gapplabs, backend:  
recall-backend
```

8. Test the running application

A. Navigate to the application and view the landing page:

<https://message-recall-gapplabs.appspot.com>



B. Navigate to the History page:

NOTE: The initial application data store indices need to be constructed. If you see the following message on the History page, initial index construction has not yet completed.

You can also check progress on initial index creation from the AppEngine administrator page under 'Datastore Indexes'.



C. Try recalling a message:

Message Recall - Create Recall Task

[Home](#)[Recall Message](#)[History](#)[About](#)[Sign out \(george@capgsfishing.com\)](#)

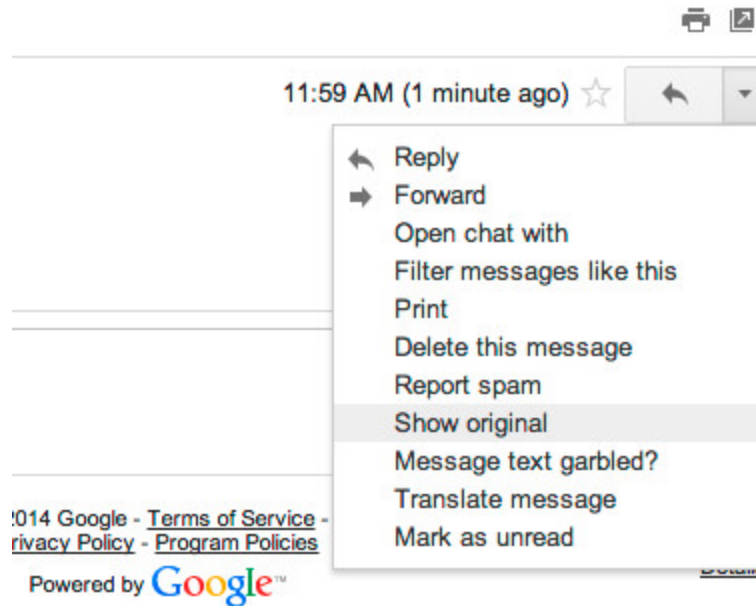
If you have ever left your computer unlocked and then a neighbor sent an inappropriate email to your vacationing boss, this is the application for you.

Supply the Message-ID and all Google Apps domain users will be checked.

The email will be purged from any active user that has received it.

Enter a Message-ID for the email to purge.

1. From Gmail, send a message to any domain user.
2. From your Gmail 'Sent Mail' folder, open the sent message.
3. You can see detailed information about your message by clicking the 'Show original'.



4. Note the Message-ID of your sent message (it is the text inside of the <> brackets).
5. Navigate to the 'Recall Messages' page of your application.
6. Enter the Message-ID of your sent message in the textbox.
7. Click 'Submit'.
8. You are dropped into the Message Recall 'Task' page. Refresh it frequently to see status of your recall task.
9. You may choose the 'View Report' page to see a different view of progress.

10. The Task state starts at '*Getting Users*' and progresses through '*Recalling*' to '*Done*'.

11. When the application completes, you should notice the following on the 'Task' page:

Users with Messages Recalled	1
-------------------------------------	---

This indicates you sent the message to one user who is not suspended, it was identified and deleted.

Appendices

Getting Help

If you need additional assistance setting up and/or configuring your application, please try the following resources:

Help with Google App Engine
The Google App Engine SDK for Python
Google Apps Documentation & Support: Sign in to your Admin console
Google APIs Console Help
The Google Apps Admin SDK
The googleapps-message-recall Google Group