

# Message Recall: Deployment Guide

Last Updated: Wed Oct 14, 2015

## 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 Enable the Admin SDK API](#)
  - [3.2 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.

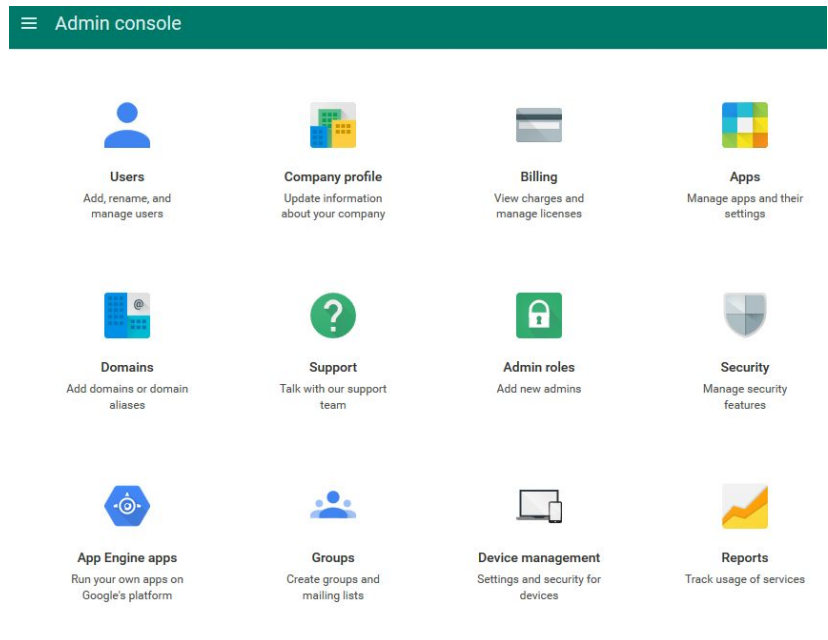
---

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

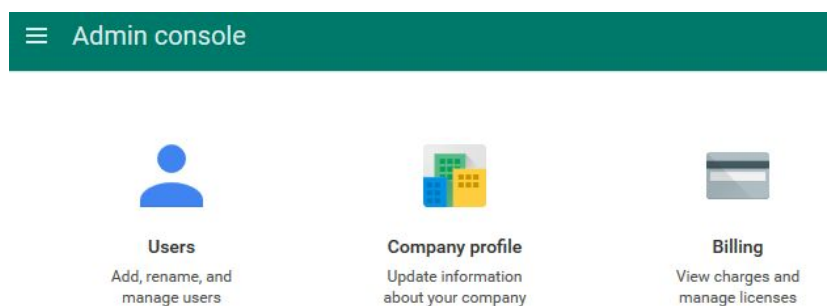
For additional reference, the following articles from the Google Support center discuss accounts and Admin roles:

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

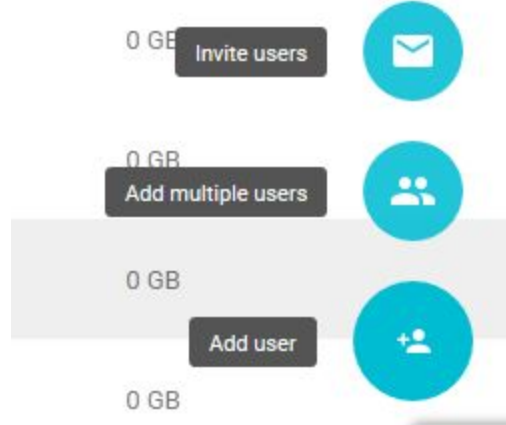
In your Google Apps Domain, while logged in as a Domain Admin, navigate to the Admin console at: <https://admin.google.com>.



Select 'Users' to create the Role account.



Add a new user by clicking 'Add user' that appears after clicking the + graphic.



Enter 'First name', 'Last name' and 'Primary email address' (we used message\_recall\_role for this example) then choose 'Create'.

On the next confirmation page, note the Password generated for the new account. You may choose to send email or print details.

Choose Done.

### Create a new user ×

RecallFirst RecallLast

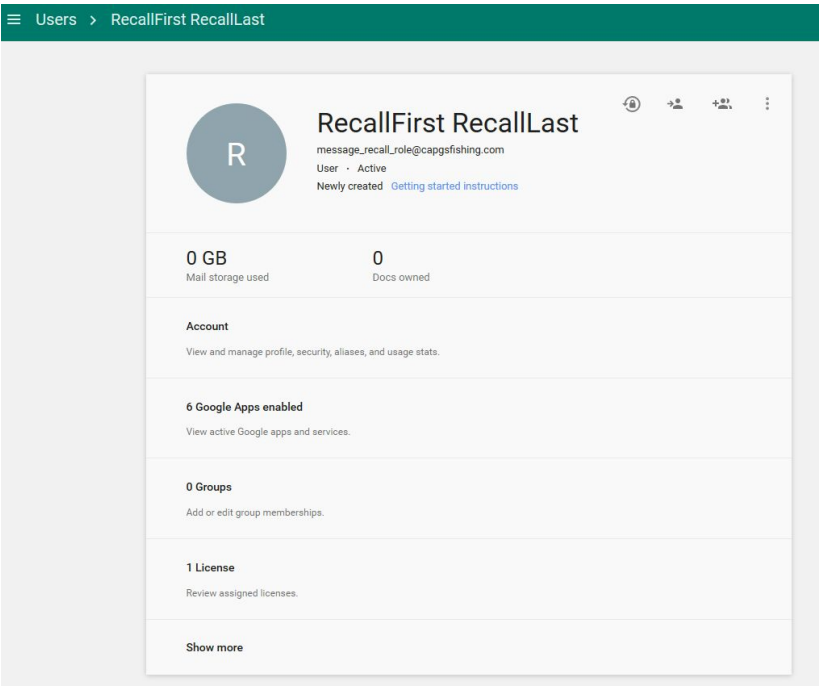
message\_recall\_role@gappslabs.com

Temporary password will be assigned - [Set Password](#)

ADDITIONAL INFO

CANCEL CREATE

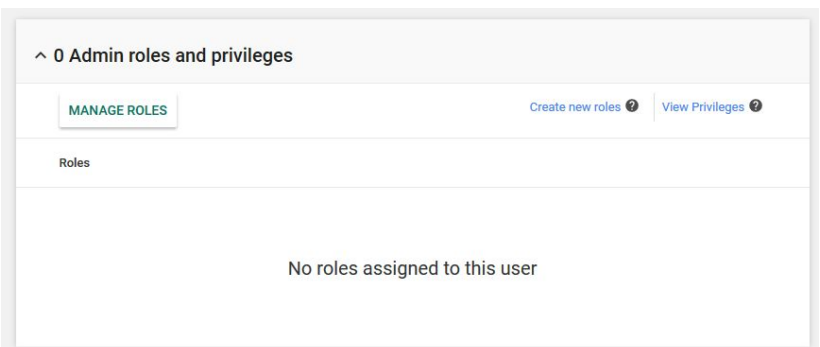
Select the new user,  
choose 'Show more' details.



Choose:  
'Admin roles and its  
privileges'.



Choose: 'Manage roles'.



Select: 'Super Admin' and choose 'Update Roles'.

Manage roles

Roles

☒ **Super Admin**  
Role for full administrative rights  
For all organizations

☐ **Groups Admin**  
Role to create and manage groups  
For all organizations

☐ **User Management Admin**  
Role to create, delete and update users  
▶ No organizations selected

☐ **Help Desk Admin**  
Role to manage support issues which requires access to user information and ability to reset passwords  
▶ No organizations selected

☐ **Services Admin**  
Role to manage services/applications  
For all organizations

UPDATE ROLES

CANCEL

You should notice your user now has '1' next to 'Admin roles and its privileges'.

^ 1 Admin roles and privileges

MANAGE ROLES

Create new roles

View Privileges

Roles

Super Admin

Role for full administrative rights

For all organizations

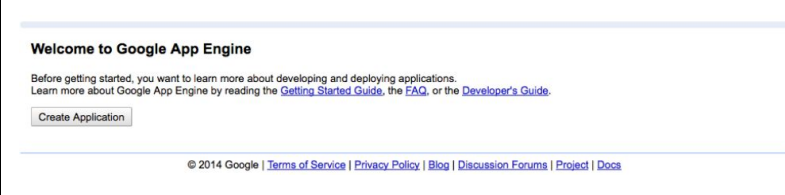
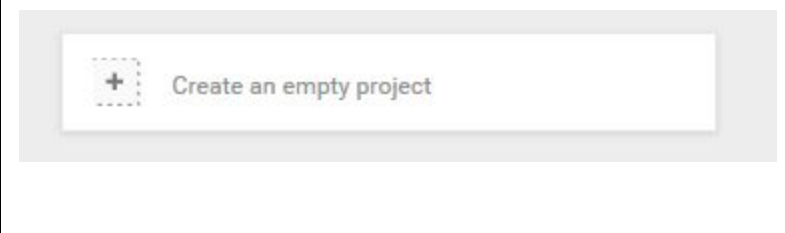
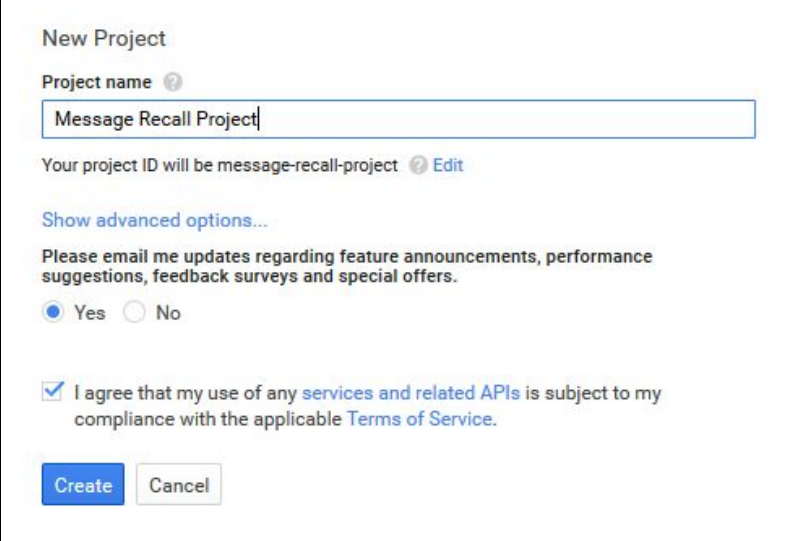
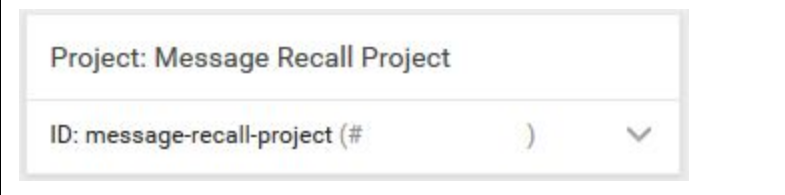

Sign out.

Sign back in as  
message\_recall\_role  
@yourdomain.com.

message\_recall\_role@gappslabs.com

## 2. Create the App Engine hosting environment

### 2.1 Create the application in App Engine

While logged in as message_recall_role, navigate to 'Google App Engine' at <a href="https://appengine.google.com">https://appengine.google.com</a> .	
Choose 'Create an empty project'.  (You may be requested to verify your account via text message or voice call.)	
Fill out the new project form.  A Project name like 'Message Recall <yourdomainname>' is recommended. This maps to an application named message-recall-<yourdomain>.  Select 'I agree...'  Choose 'Create'.	
You should see the new Project.	
If you return to 'Google App Engine' at <a href="https://appengine.google.com">https://appengine.google.com</a> you should see your new application project.	

## 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**.

Choose 'message-recall-<yourdomain>' to see Application details.	<div><b>My Applications</b></div> <div><div>◀ Prev 20 1-1 of 1 Next 20 ▶</div><table><thead><tr><th>Application</th><th>Title</th></tr></thead><tbody><tr><td><a href="#">message-recall-project</a></td><td>Message Recall Project</td></tr></tbody></table><div>You can create your new cloud project in the <a href="#">Google Developers Console</a>.</div></div>	Application	Title	<a href="#">message-recall-project</a>	Message Recall Project
Application	Title				
<a href="#">message-recall-project</a>	Message Recall Project				
Now, under Billing, choose: 'Billing Status'.	<div><b>Billing</b></div> <div><a href="#">Billing Status</a></div> <div><a href="#">Usage History</a></div> <div><a href="#">Transaction History</a></div> <div><a href="#">Billing Profile</a></div> <div><a href="#">Billing Settings</a></div>				
Now jump to the new Developers Console Billing page.	<div>Try the new Developers Console <a href="#">Billing page</a>.</div>				
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.	<div><b>Billing</b></div> <div>Enable billing to access the full set of services and increased usage limits.</div> <div><a href="#">Enable billing</a></div>				
You will need to approve the billing setup by clicking a link in the email inbox you use when you setup the payment instrument.	Google Billing: Verify your email address				



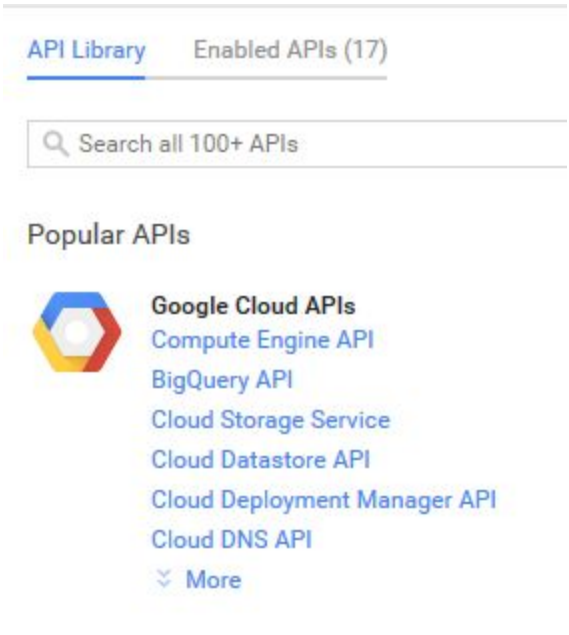
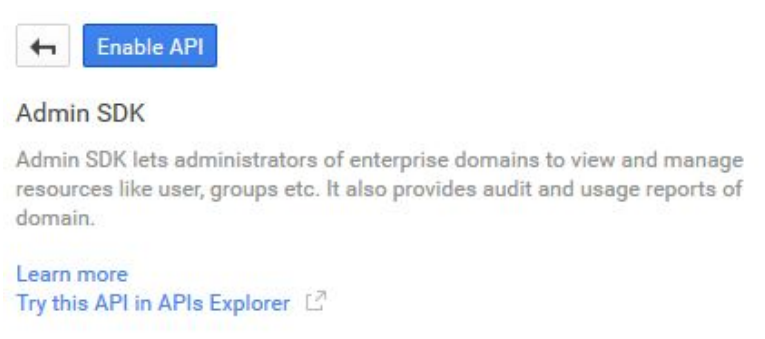
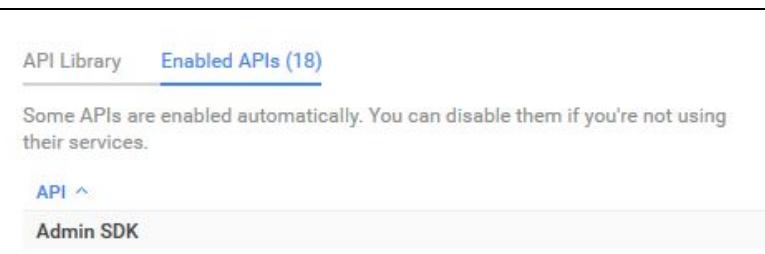
<p>The subject of the email should be 'Google Billing: Verify your email address'</p>	
---	--

Click the link in the message body to verify your email.

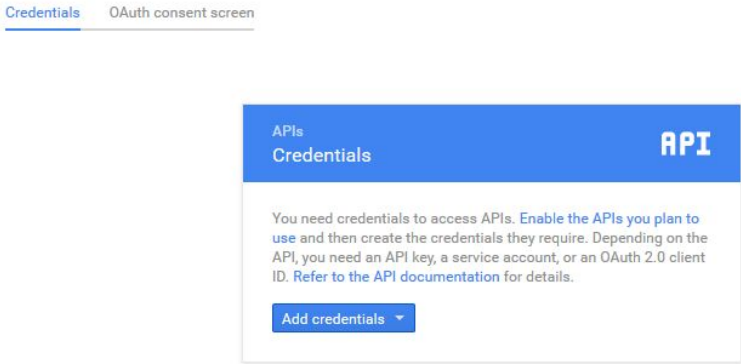
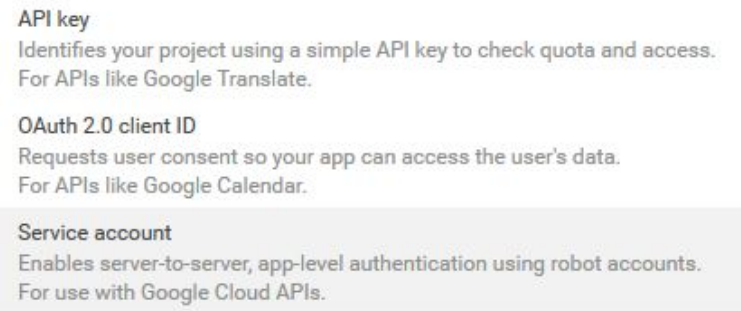
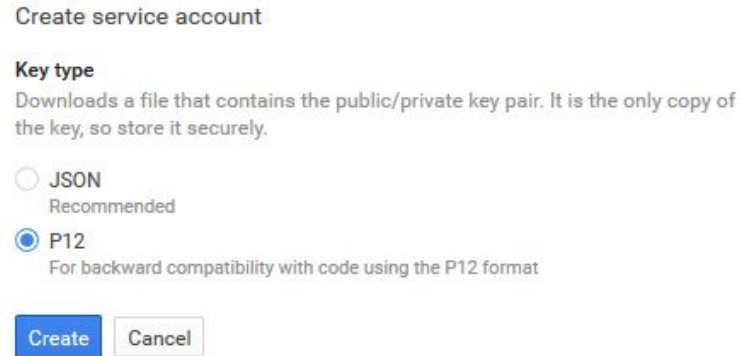
---

## 3. Configure the Google APIs Console Project

### 3.1 Enable the Admin SDK API

<p>In the Developer Console for your project, choose: 'APIs &amp; auth'.</p> <p>Then choose APIs.</p> <p>This will drop you into the APIs panel.</p>	
<p>Type 'Admin SDK' in the Search box.</p> <p>Choose 'Admin SDK' in the results.</p> <p>Choose 'Enable API' to enable this API.</p>	
<p>The 'Admin SDK' should now be listed in the 'Enabled APIs' list.</p>	

## 3.2 Create a Project Service Account

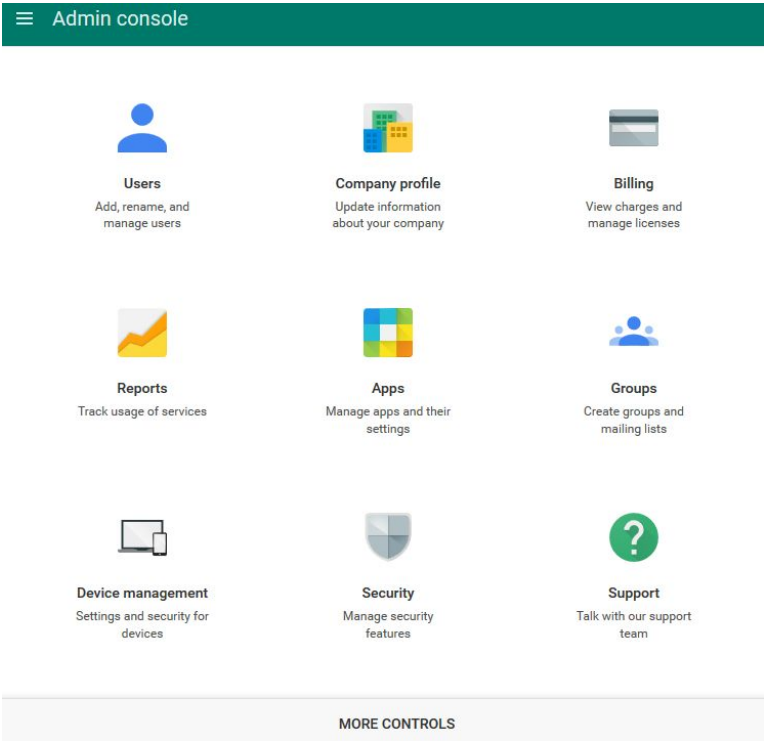
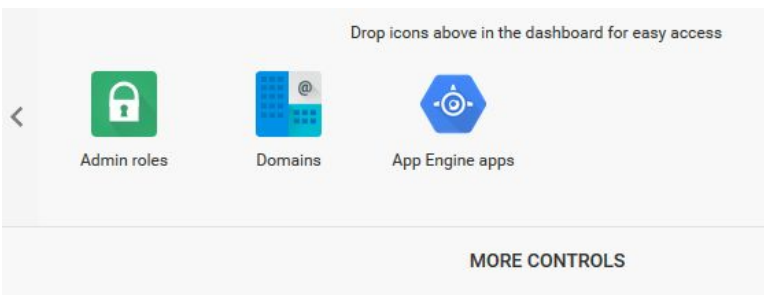
<p>Choose 'Credentials' from the API Console Project.</p> <p>Then, choose 'Add credentials' to create your new OAuth 2.0 API application client.</p>	
<p>Choose 'Service account'.</p>	
<p>Choose 'P12' for backward compatibility.</p> <p>Click 'Create'.</p>	

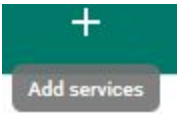

<p>A file is generated and downloaded to your local system with your private key material. The file has a name of the form 'XXXX-privatekey.p12'.</p> <p>Choose 'Close' to proceed.</p>	<p>New public/private key pair</p> <p>Message Recall Project-7768782e41c7.p12 has been downloaded to your computer and serves as the only copy of this key. Store it securely.</p> <p>This is the private key's password. It will not be shown again. You must present this password to use the private key. <a href="#">Learn more</a></p> <div>notasecret</div> <div>Close</div>						
<p>Your OAuth 2.0 client information is now displayed in the Developer Console.</p> <p>Click the email address to see all details.</p> <p>Take special note of the 'Client ID' and 'Email address' as both will be needed later.</p>	<p>Service account</p> <table><tr><td>Client ID</td><td>xxxx.apps.googleusercontent.com</td></tr><tr><td>Email address</td><td>xxxx@developer.gserviceaccount.com</td></tr><tr><td>Certificate fingerprints</td><td>#####</td></tr></table> <div>Done</div>	Client ID	xxxx.apps.googleusercontent.com	Email address	xxxx@developer.gserviceaccount.com	Certificate fingerprints	#####
Client ID	xxxx.apps.googleusercontent.com						
Email address	xxxx@developer.gserviceaccount.com						
Certificate fingerprints	#####						

---


## 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 <a href="https://admin.google.com">https://admin.google.com</a>.</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>Jump to the 'Google Developers Console' to continue.</p>	<p><b>This feature has moved to the <a href="#">Google Developers Console</a>.</b></p>



<p>Notice the project is 'Message Recall Project' and choose 'Continue'.</p>	<p>Select or create a project</p> <p>The Google Developers Console uses projects to manage resources. To use this feature, select an existing project or create a new one.</p> <p>Select a project</p> <div> <div>Message Recall Project (message-recall-project)</div> <div></div> </div> <p>Continue</p>
<p>Click the 'Add services' + graphic on the top-right.</p>	
<p>In 'Other Services', enter the name of the app engine application created: 'message-recall-&lt;yourdomain&gt;'.</p> <p>Choose 'Add It now'.</p>	<p>Other Services</p> <div>  <div> <p>Google App Engine</p> <p>Provide existing Google App Engine services to your users.</p> </div> </div> <p>Enter App ID:*</p> <div> <div>message-recall-project</div> <div></div> </div> <p>ADD IT NOW <a href="#">Learn More</a></p>
<p>Select the box to activate the service 'Agreement'.</p> <p>Choose 'Activate this service'.</p>	<p>You have requested that the 'message-recall-gappslabs' service be added</p> <p><small>Please be careful when adding non-Google services to your domain, and make certain you know and trust the developer adding this service to your domain.</small></p> <p><small>Please accept the Google App Engine terms and conditions to continue</small></p> <div> <p><a href="#">Printable Version</a></p> <p>GOOGLE APP ENGINE TERMS AND CONDITIONS</p> <p>Use of the Google App Engine is governed by the terms of service available at <a href="http://code.google.com/appengine/terms.html">http://code.google.com/appengine/terms.html</a>. By adding an application developed using the Google App Engine service (the "Service") to your Google Apps domain, you understand and agree to the following:</p> <ul style="list-style-type: none"> <li>a. Service is developed by a third party, not by Google. The Service provides you the</li> </ul> <p>Agreement * <input checked="" type="checkbox"/> I accept. Continue to add this service. <a href="#">Learn more</a></p> <p><small>By selecting "I accept. Continue to add this service," you are agreeing to the Google App Engine terms and conditions. You can create custom web addresses for this service at any time on the service settings page.</small></p> <div> <div>Activate this service</div> <div>Cancel</div> </div> </div>

## 4.2 Authorize your application in the Google Apps Domain

Return to the 'Admin console' and choose 'Security'	 <p><b>Security</b> Manage security features</p>
Choose 'Show more'.	<p><b>Show more</b></p>
Choose 'Advanced settings'.	<p><b>Advanced settings</b> Manage advanced security features such as Single Sign On, authentication, and integrating Google Apps with internal services.</p>
Under 'Authentication' choose 'Manage API client access'.	<p><b>Authentication</b></p> <p><a href="#">Manage OAuth domain key</a> Allows admins to access all user data without needing login credentials. ?</p> <p><a href="#">Federated Login using OpenID</a> Allows users to sign-in to 3rd party websites using their capgsfishing.com account, without giving away their credentials.</p> <p><a href="#">Manage API client access</a> Allows admins to control access to user data by applications that use OAuth protocol.</p>
<p>In the 'Client Name' field, enter the Service Account 'Client ID' from step 3.2 earlier.</p> <p>It should be of the form: ##.apps.googleusercontent.com</p>	<p><b>Authorized API clients</b></p> <p><b>Client Name</b></p> <p>1035314471864.apps.goc</p> <p>Example: www.example.com</p>
<p>In the 'One or More API Scopes' box, enter the following scopes string:</p> <p><a href="https://mail.google.com/">https://mail.google.com/</a></p> <p>Choose 'Authorize'</p>	<p><b>One or More API Scopes</b></p> <p><a href="https://www.googleapis.com/auth/admin.directory">https://www.googleapis.com/auth/admin.directory.</a> <b>Authorize</b></p> <p>Example: http://www.google.com/calendar/feeds/ (comma-delimited)</p>
You should notice an entry for your service account now listed.	<p><b>Email (Read/Write/Send)</b> <a href="https://mail.google.com/">https://mail.google.com/</a> <b>View users on your domain</b> <a href="https://www.googleapis.com/auth/admin.directory.user.readonly">https://www.googleapis.com/auth/admin.directory.user.readonly</a></p>

## 5. Prepare the source code

### 5.1 Get the source code

Browse to the github.com page for <a href="#">googleapps-message-recall</a> .	 <b>google</b> / <b>googleapps-message-recall</b>
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.	<b>From:</b> application: message-recall  <b>To:</b> 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.	<b>From:</b> SERVICE_ACCOUNT_NAME = ( '000000000000-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx' '@developer.gserviceaccount.com' )  <b>To:</b> <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 <a href="#">Windows</a>  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: &lt;No Attributes&gt;  -----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'	<a href="https://developers.google.com/appengine/downloads#Google_App_Engine_SDK_for_Python">https://developers.google.com/appengine/downloads#Google_App_Engine_SDK_for_Python</a>
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 <a href="#">message_recall_role@yourdomain.com</a> user.</p>	<pre>\$ appcfg.py --noauth_local_webserver 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>The dialog box shows the Google App Engine logo and the title 'Google App Engine appcfg'. It asks 'This app would like to:' and lists 'View and manage your applications deployed on Google App Engine'. At the bottom, there are 'Cancel' and 'Accept' buttons. A small disclaimer at the bottom states: 'Google App Engine appcfg and Google will use this information in accordance with their respective terms of service and privacy policies.'</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-gappslabs, version: 1 Uploading index definitions. Uploading task queue entries.</pre>
<p>Now update the queue setup.</p>	<pre>\$ appcfg.py update_queues .</pre>
<p>Now upload the back-end code.</p>	<pre>\$ appcfg.py backends . update</pre>
<p>You should notice the following progress on your console confirming the deployment.</p>	<pre>Starting update of app: message-recall-gappslabs, backend: recall-backend Getting current resource limits. Scanning files on local disk. ... Compilation starting. Compilation completed. Starting deployment.</pre>

	<pre>Checking if deployment succeeded. Deployment successful. Checking if updated app version is serving. Completed update of app:   message-recall-gapplabs, backend: recall-backend</pre>
--	---

---

## 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 - Recall Message

[Home](#)[Recall Message](#)[History](#)[About](#)[Sign out](#)

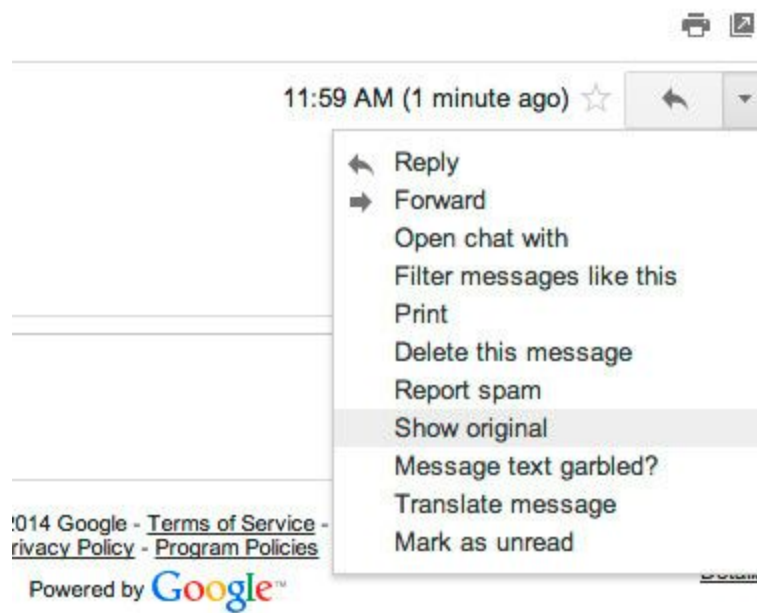
Use this tool to remove an Email message from your domain-users mailboxes. This is useful in critical situations when an email message has been inadvertently sent to your domain users.

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

The email will be deleted 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 if the message was only sent to 1 user. This indicates the user was identified, is not suspended, and the message was able to be deleted.



## Message Recall - Task

[Home](#)[Recall Message](#)[History](#)[About](#)[Sign out](#)

Tip: Refresh frequently to see updates.

<b>State</b>	
<b>Owner</b>	
<b>Message-ID</b>	
<b>Start (UTC)</b>	20151024 10:20:49
<b>Stop (UTC)</b>	20151024 10:21:06
<b>Elapsed (m:s)</b>	
<b>Total Users</b>	
<b>Users Processed</b>	
<b>Users with Messages Recalled</b>	1

[View Users](#)[View Report](#)[Debug Task](#)



# Appendices

## Getting Help

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

<a href="#">Help with Google App Engine</a>
<a href="#">The Google App Engine SDK for Python</a>
<a href="#">Google Apps Documentation &amp; Support: Sign in to your Admin console</a>
<a href="#">Google APIs Console Help</a>
<a href="#">The Google Apps Admin SDK</a>
<a href="#">The googleapps-message-recall Google Group</a>