William Matheson

2015-04-19

MBA 614 Final Project

Dr. Gove Allen

# Automated Email Scheduler for Gmail

## Executive Summary

This project aims to create an email scheduling solution through Microsoft's Excel VBA for small organizations and individuals who use Gmail as their primary email application. The benefits will help users efficiently communicate with clients, suppliers, applicants, recruiters, contacts, management, direct reports etc. Once the workbook is loaded up with pending drafts, the workbook will need to be scheduled to run as a task in Windows Task Scheduler to be run every hour.

Relationship management with clients, suppliers, applicants, and coworkers is an important facet of today's world of business. Employees who remember to maintain contacts with key individuals in a timely manner are able to excel. Email is becoming a critical channel for communication. Automated e-mails with canned messages are commonly used but lack the personal touch of a phone call or customized e-mail. One struggle with writing customized e-mails is making sure the message is sent to the recipient at the right time. During a busy week when work is piling up, it is easy to lose track of time to get an important e-mail out Friday morning and ends up being remembered Friday at the end of day. This can result in delayed projects, lost productivity, and slow turnaround times in communication. This effect is compounded when dealing with overseas contacts whose workday is opposite your own.

The business need is a tool that can assist users who want to schedule e-mails to send to their contacts at a specific time. This way, users can have a productive episode of drafting e-mails at any time of the day or night and have the messages sent out at scheduled times in the future. The benefit is not letting important communication between the user and their contact be at the mercy of a busy schedule.

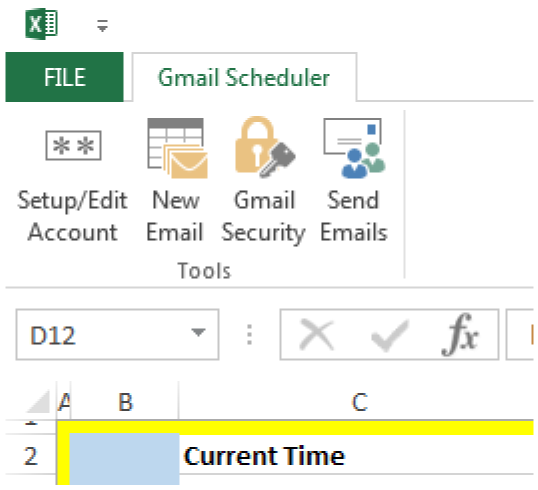
The most commonly used application for workplace e-mail is Microsoft Outlook. A feature to schedule e-mails at a specific time exists for Outlook and greatly benefits employees as they strive to manage relationships and communication pertaining to their work. Google for Work is a growing SaaS solution for small businesses which offers Gmail as an application for e-mails. The same functionality of scheduling e-mails does not exist for Google's Gmail. There are third party developers who have created Chrome Extensions which do offer scheduling functionality. However, these solutions come with subscription plans and have tiered service levels which can limit the number of scheduled e-mails you can schedule or send in a period of time.

## Implementation

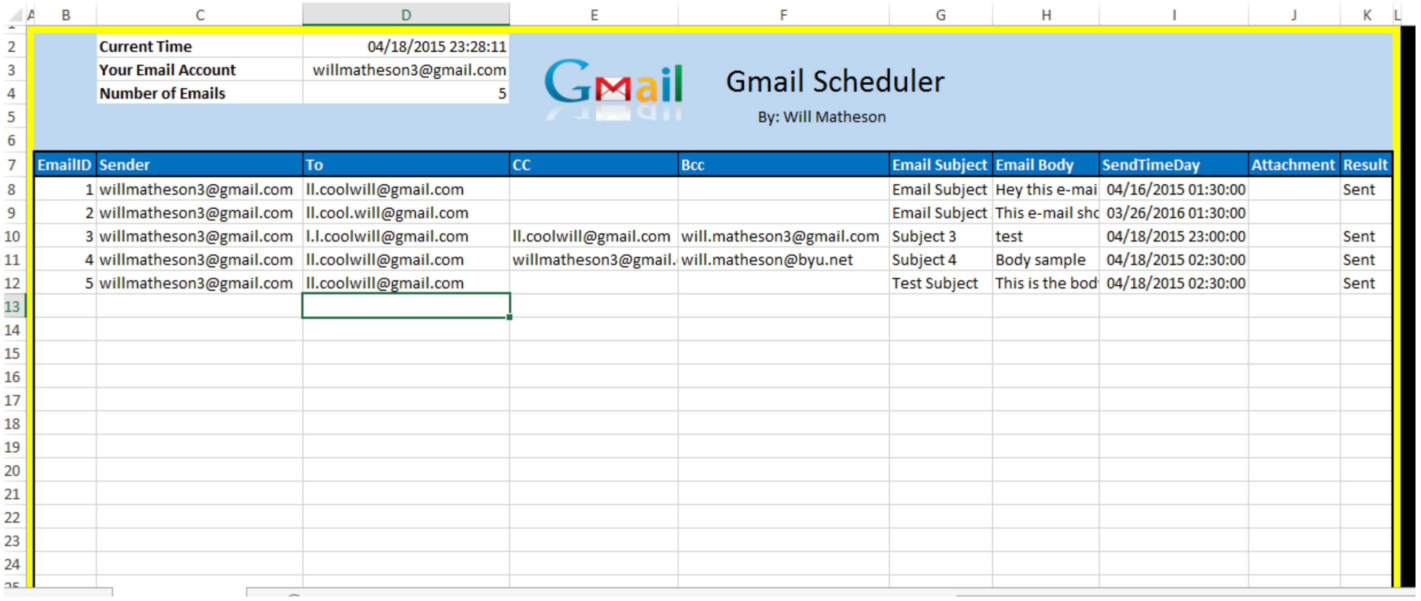
Overview of functions –

The main functions of the Gmail Scheduler tool are setting up your account information, drafting a new e-mail, sending scheduled e-mails, and toggling the Gmail account's settings to allow "less secure apps" like VBA to be able to send e-mails.

There are two main components of the user interface. The first is the Excel Ribbon which has no other Excel features and looks like this.



The second component to the UI is the dashboard where the e-mails are stored. It looks like this –



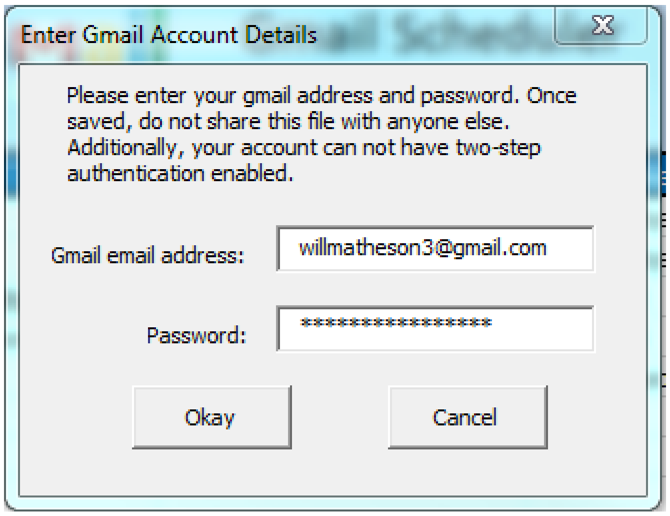
| | |
|---|---|
| Current Time | 04/18/2015 23:28:11 |
| Your Email Account | willmatheson3@gmail.com |
| Number of Emails | 5 |

Gmail Scheduler

By: Will Matheson

| EmailID | Sender | To | CC | Bcc | Email Subject | Email Body | SendTimeDay | Attachment | Result |
|---|---|---|---|---|---|---|---|---|---|
| 1 | willmatheson3@gmail.com | ll.coolwill@gmail.com | | | Email Subject | Hey this e-mai | 04/16/2015 01:30:00 | | Sent |
| 2 | willmatheson3@gmail.com | ll.cool.will@gmail.com | | | Email Subject | This e-mail shc | 03/26/2016 01:30:00 | | |
| 3 | willmatheson3@gmail.com | l.l.coolwill@gmail.com | ll.coolwill@gmail.com | will.matheson3@gmail.com | Subject 3 | test | 04/18/2015 23:00:00 | | Sent |
| 4 | willmatheson3@gmail.com | ll.coolwill@gmail.com | willmatheson3@gmail. | will.matheson@byu.net | Subject 4 | Body sample | 04/18/2015 02:30:00 | | Sent |
| 5 | willmatheson3@gmail.com | ll.coolwill@gmail.com | | | Test Subject | This is the bod | 04/18/2015 02:30:00 | | Sent |

I will begin with the breakdown of the ribbon. First, the workbook needs to be initialized with the users e-mail address and password. When the workbook opens, the tool checks to see if there is a functioning account. If not, the **frmGmailAccountDetails** form will open up. This form asks for the users e-mail and password.


Enter Gmail Account Details

Please enter your gmail address and password. Once saved, do not share this file with anyone else. Additionally, your account can not have two-step authentication enabled.

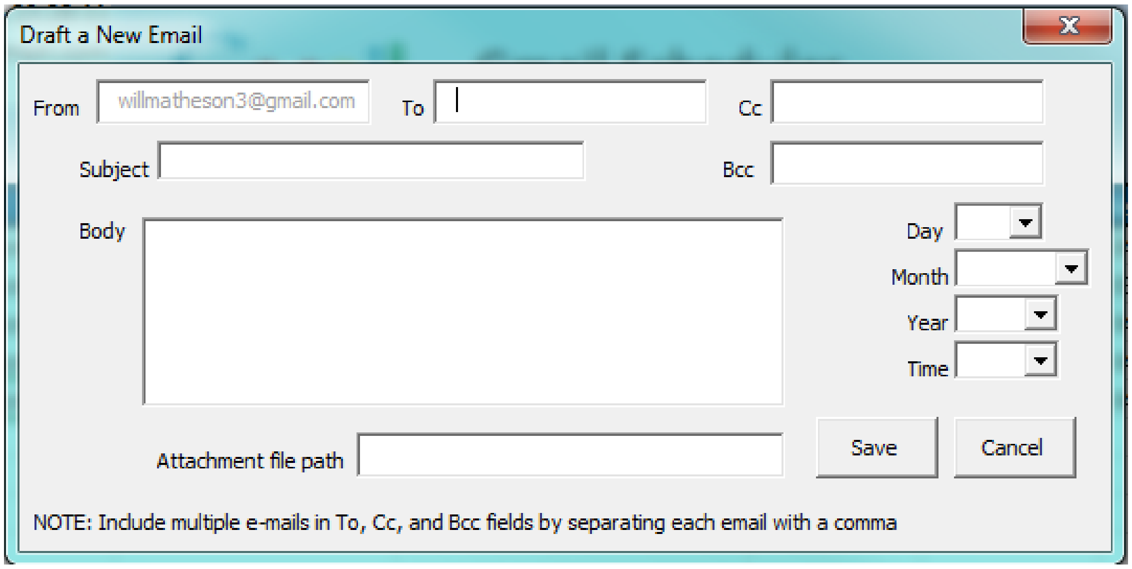
Gmail email address: willmatheson3@gmail.com

Password: *****************

The form notifies the user to keep the workbook safe if they choose to use it because it will then have their email and password saved in the workbook. It also lets the user know that an account with the two-step verification enabled will not work. If the workbook already has an active account, then the user can access this form and edit the email settings by clicking the **Setup/Edit Account** button in the ribbon.

The Gmail address uses a Regular Expression validator sub found online to help validate the e-mail to ensure it is a valid email address. I modified the sub to make it only work with *gmail.com* e-mail addresses. If no e-mail or password is entered, the user will be notified to fill them out. The password is obscured from the user's view. These two variables are put on the **GmailAccountDetails** sheet, a very hidden sheet. It is important to note that this tool only allows one e-mail account to be the main e-mail at once.

Next, the **New Email** button activates the frmDraftEmail form and pulls up the following UI.


Draft a New Email

From willmatheson3@gmail.com

NOTE: Include multiple e-mails in To, Cc, and Bcc fields by separating each email with a comma

The **From** field is automatically populated with the data on the hidden sheet **GmailAccountDetails**. Since this field is the one that is sent through the Gmail sender sub, this field is unchangeable by the user. There are no validators on the To, Cc, and Bcc e-mails so the user needs to ensure that the e-mails are correctly written and separated by commas. The Subject, Body, and one of the To, Cc, or Bcc e-mails needs to be populated. There are if conditions that test for these criteria before the user can choose save. The Day, Month, Year, and Time are combo boxes that get their data from another very hidden sheet called **Dates**. The date items are concatenated into a correctly formatted string and entered into the appropriate field when saved. This correct format is important to make the scheduler work. The attachment field is a string that requires the user to give the file path for the file they would like to attach to their e-mail.

When the form is saved, all the data is put into a row in the main dashboard with an email ID that is the count of the number of current e-mails.

The next ribbon item is the **Gmail Security**. When clicked, this runs the **gmailLogin** and **toggleLesssecureapps** subs. The **agent** module that allows for IE objects needs to be enabled. The intention was to have IE open up, log into Gmail, then load the less secure apps security page which allows VBA to send e-mails through gmail. The functionality of this feature is ***not working in the current version*** and the user will need to turn off the security settings manually if they want to use this tool. I was able to get the tool to turn off the security settings once logged in but occasionally it would run into an error while logging in. Also, it was difficult to use the agent module to identify a tag in the HTML by its inner text. After much searching online for assistance on this matter, I decided to save the completion of this feature for version 2.0. Since this is usually a one-time things, it shouldn't be a hindrance using the tool long-term.

The last ribbon is the core functionality of the tool. It takes the current time and determines whether or not the emails in the list should be sent using the **timeValidate** function. If the **timediff** variable is more than 0, then the e-mail should be sent because it means that the send time is in the past. When the e-mail is sent, the response puts a "Sent" or "Error: not sent" message in the result column (column k) to show whether or not the e-mail was sent.

## Evaluation and Implementation

The overall evaluation of the project is that it does complete the core functionality of being able to schedule e-mails and send them when the time is passed. Further implementation of being able run this workbook every hour will need to use the following code saved as a .vbs file –

```
Sub gmailrunner()
‘ taken from http://stackoverflow.com/questions/22771185/how-to-set-recurring-
schedule-for-xlsm-file-using-windows-task-scheduler
dim eApp
set eApp = GetObject("C:\excel\tester.xlsm")
eApp.Application.Run "tester.xlsm!test"
set eApp = nothing
End Sub
```

A schedule in Windows Task Scheduler will need to be set up to run this exe – C:\windows\system32\cscript.exe with the added argument of ~\gmailrunner.vbs where the ~ is wherever that file is located on your local computer. The recommended schedule would be to have this run every hour during work hours.

As a result of this being run with Windows Task Scheduler, it is important to know that the tool will only work on a computer that is on during the time the e-mail should be sent.

**Learning and Conceptual Difficulties**

Learning –

The following are things I learned as a result of the project.

- How to focus on core functionality and avoid getting bogged down on non-critical details
- How to run an Excel Macro on a schedule using Windows Task Scheduler
- How to Toggle the Security Settings in Gmail
- How to format date columns in VBA
- How to validate email addresses using regular expressions
- How to find elements in HTML by using XPath
- How to protect sheets and cells
- Hot to concatenate strings to create a date format
- How to hide sheets in a workbook and use them

Challenges –

I struggled in the following areas.

- Including a date picker on the Create an e-mail form. I wanted this really bad but I couldn't access it.
- Finding an element by its inner text
- Validating a string of e-mail addresses separated by commas
- Protecting the sheet in a way that wasn't annoying to the user
- How to get time zones of the system
- How to integrate with Gmail's API

To include in version 2.0 –

- A clear sent e-mails button
- Handle time zones
- Finish the toggle of security settings
- Fetch existing drafts in Gmail account
- Easier to schedule with Windows task scheduler
- Signature support
- HTML support
- Clean up date drop down boxes; try to get calendar date picker

## Assistance

Most of the code in this project (around 80%) is original. The agent class module and the SendThruGmail function were used as a starting point and were from Gove Allen's demonstrations from class. Although, the code for these pieces are widely used on the internet and versions of it could be found in many places.

The validate e-mail address function is the only other things borrowed and modified on in the project. The source for this is listed in the code.