Spencer Jones

IS 520

April 14, 2015

The Daily Report Generator

Executive Summary

For the past three years I've been employed as a Driving Instructor with Revolution Driving Instruction, a driver training company with four offices in Utah. Revolution requires each of its Instructors to send a Daily Report containing information about the hours worked each day and about the students who attended each class to the company's Internal Auditor. Based on these reports, the Auditor is tasked with compiling the payroll and with verifying that each class and each student's attendance has been appropriately documented with the government. The problem is that creating these reports on a daily basis is very tedious, especially since the majority of the information is already stored on company-shared Google Calendars and the task is therefore largely one of copying and pasting.

The system that I have built automates the creation of the Daily Reports and does it in a fraction of the time while requiring only a couple clicks and a few keystrokes from the user. This system—the Daily Report Generator—will request some information from the Instructor, allow them to sign in to the appropriate Google Calendar, identify the calendar events that correspond to that Instructor, extract the necessary information, and compile the Daily Report. The Generator will then present an editable draft of the report to the Instructor and provide options for immediately emailing the report to the auditor from within the program, copying the report to a Gmail draft for later attention, rerunning the report in case something was missed, or aborting and discarding the draft. Thus the tedious task of creating a Daily Report, which I have probably performed at least 700 times over the last three years, has been reduced to a couple clicks and a few keystrokes.

Solution Walkthrough

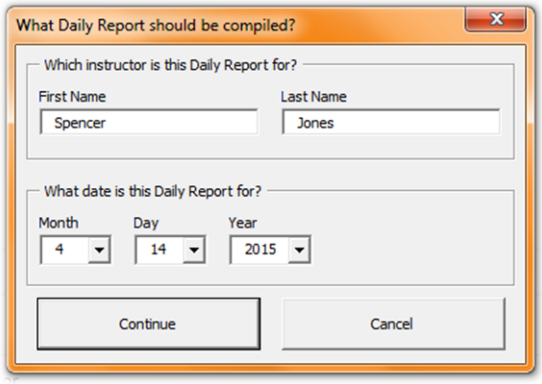
The Daily Report Generator goes through several steps in reaching the ultimate goal of producing the Daily Report. Each step is as follows:

1. Collect Instructor's name and the date of the report
2. Open a browser window, navigate to appropriate Calendar, and allow the Instructor to sign in

3. Identify the classes (calendar events) that were taught by the Instructor and which should be included in the report
4. Extract the appropriate information from each event based on what type of class or work was performed
5. Compile the report according to company standards
6. Allow the Instructor to edit and send or save the report
7. Sign out of the Google account and close the browser window

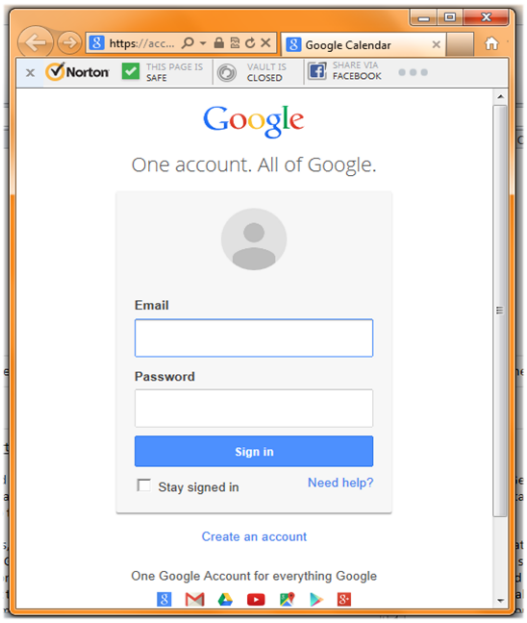
Step 1

In developing this program, I wanted to be able to make it available to coworkers so that they could benefit from the increased simplicity of filing these reports on a daily basis. So the first step naturally was to identify the Instructor that the report should be compiled for. It is at this time that the report date is also determined. This was accomplished with a simple userform which is displayed at the click of the custom button on the modified ribbon and which accepts the Instructor's first and last names and the report date. The first name will be used to select the appropriate information from the calendar and the last name is used only in compiling and signing the report. The fields for the date are automatically populated with today's date for convenience. All 12 months are available of course, and the days are updated upon selection of the month to correctly represent longer and shorter months as well as leap years. The year field is populated with the current year and the previous year. If the Instructor enters an improper date or fails in some other way to properly complete the userform, they are presented with a warning and are unable to proceed.



Step 2

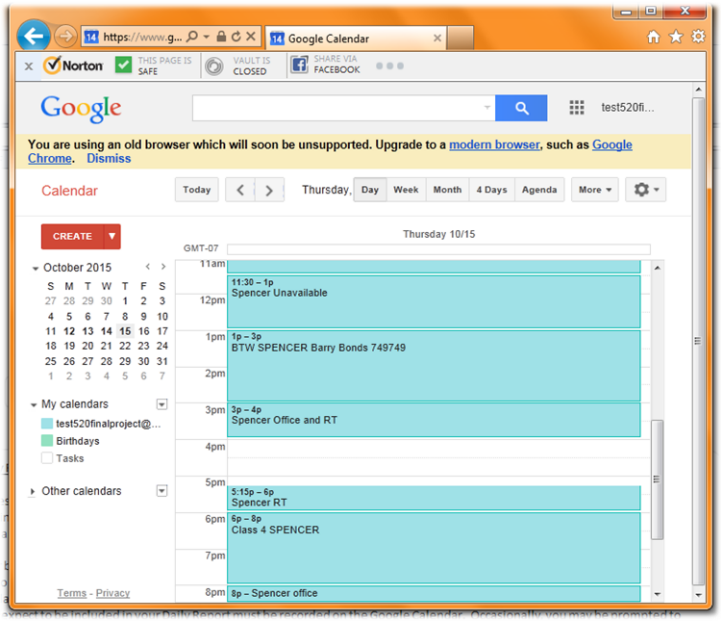
In this step a browser window (Internet Explorer) is opened and navigated to the Calendar and the report date. By manipulating the URL parameters the browser is easily navigated directly to a page displaying only the events for the specified date. The browser remains hidden until it is ready for input by the Instructor, at which point it makes itself available and the Generator waits for the Instructor to sign in before proceeding. If the Instructor does not sign in and closes the window, the Generator will detect this and ask if they would like to reopen



the window or abort the program. The code waits for the Instructor to log in and when it detects that the sign in was successful the browser window is hidden and it proceeds to the next step.

Step 3

Because information about hours and classes for several Instructors are stored on one Google Calendar (essentially one calendar per office), it was important to distinguish between events that belonged to the Instructor using the Daily Report Generator and others. This was accomplished by matching the Instructor's first name provided in Step 1 with the Instructor name that is always listed in the event title to indicate who is teaching the class. The Generator loops through the HTML tags on the page, searching for a match on the Instructor's name. The Generator also checks other criteria in order to ignore event titles that indicate that the Instructor is unavailable and to identify what type of class is being taught or what type of worked is being performed. When it identifies a qualifying event, the event is clicked on because further information is needed from inside the event. After the Generator has extracted the appropriate information (Step 4), it returns to the main page and continues the loop.



Step 4

Once inside the event page, the Generator extracts the appropriate information based on the type of work being performed and stores it in an array. There are essentially three types of events: Classroom training, Behind the Wheel training, and Office Hours/Road Tests. For each type of event, the Generator needs to collect the beginning and ending times of the class and calculate the duration. This was facilitated with a separate VBA function that I wrote specifically for this purpose. The function will look at the HTML, pull the times and calculate the duration, and finally concatenate the values and pass back a string that can be worked with. When a Classroom training event is selected, this is the only information that is needed from the calendar event page. When a Behind the Wheel class is selected, the program will gather the event title, the information stored in the event description, and the times. This information will before further manipulated and written into an entry to be included in the report and stored in an array. When an event indicating Office Hours is selected, the Generator will calculate the duration and add it to a running total of Office Hour work for the entire day. If any Road Tests were given during this time, the Generator will count them and add them to the totals for the day as well.

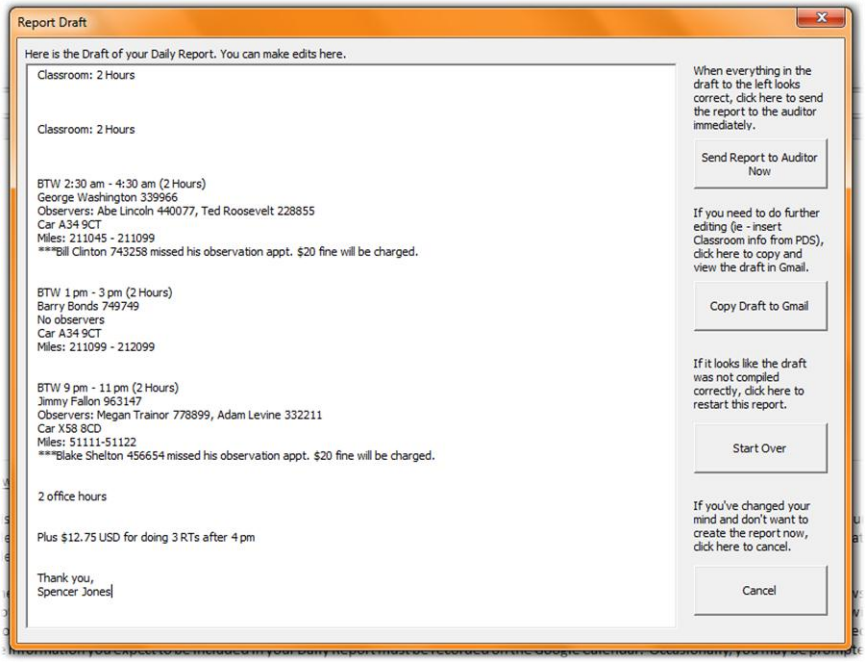
All of the information gathered during this step is stored in two separate string arrays and three global variables in order to facilitate ordering and totaling them properly when the Daily Report is compiled.

Step 5

In this step, the draft of the Daily Report is compiled. Classroom training is always listed first simply with a line for each Classroom class that was given and how long the class was. The Generator loops through a string array containing each entry of Classroom training for as many Classroom classes as were given by the Instructor on the report date. Next, Behind the Wheel classes are listed. Again, a loop through the string array containing each Behind the Wheel entry is used for as many Behind the Wheel classes as were given. Behind the Wheel entries in the report include the times of the class, the student driver's name, the names of any students who observed, the license plates of the car that was used, and the mileage on the car at the beginning and end of the class. This information was collected for each class in Step 4. Finally, an entry is written to the report for the total time spent doing Office Hours. A line is included in the report for Road Tests only if any were given on the report date. The line includes the count of Road Tests given and the total commission that should be paid, distinguishing between Tests given before or after 4 pm because of a different commission rate. The report is then signed using the Instructor's first and last names that were provided in Step 1.

Step 6

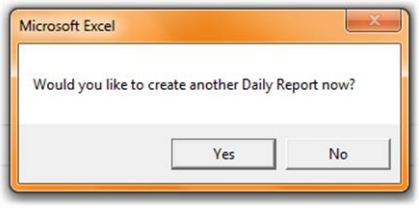
In Step 6, the Instructor is presented with another userform. This userform contains the draft of the Daily Report that has been compiled. The Instructor is able to edit any lines or entries as they see fit, with the changes being saved before proceeding. The Instructor has several options of how to proceed from this point. The first is to send the Daily Report immediately to the Auditor. Because the Instructor has already signed in to their Google account, the simplest way to accomplish this was to open a browser window to a Gmail draft with the various fields auto-populated and send the report, all of which will occur in a hidden window



that the Instructor will never see. The second option is the copy the report draft to a Gmail draft but not to send it right away. This allows the Instructor further editing capabilities and the ability to save the draft for later attention. This is accomplished in essentially the same manner, but this time with a visible browser window. The third option available to the Instructor is to recompile the Daily Report. The feature may be important if the instructor sees something that was not reported correctly. Finally, the Instructor has the option of cancelling the report and discarding the draft.

Step 7

The final step is to sign out of the Google account to protect the Instructor's privacy and to close the browser windows. Before this happens, however, the Instructor is given the option of generating another Daily Report. This will be a useful feature if the Instructor has gotten behind on sending the reports and needs to send in two or three at a time. If the Instructor chooses to create another Daily Report, they are taken back to Step 1 to choose the new report date and continue from there.



Learning and Conceptual Difficulties

I feel that I learned a great deal in putting this project together. I have programmed in other languages, but none have been similar to VBA. In working on this project, I learned more about looping in VBA and about arrays. Perhaps the area where I learned the most was working with objects and object variables. This was completely new to me and took a little while to get used to at first. Because of the nature of my project, I also had to learn a little bit about HTML in order to reach my goal.

One of the biggest difficulties that I ran into came early on because I thought that it would be better to try and work with Google Chrome. In my experience, Internet Explorer and some of Google's pages don't play very nicely together and so I started my project automating Google Chrome using a package I downloaded called Selenium. It was useful and I was able to make some progress. But by the time it came to extracting event details, I realized that I was spinning my wheels too much trying to work with a tool I was unfamiliar with and had almost no documentation and that, as Dr. Allen had said in class, Internet Explorer is the only browser which makes itself readily available for manipulation in VBA. At that point, I made the switch and began using Dr. Allen's "agent" class and things went a lot more smoothly from there.

Assistance

The main assistance that I received in completing this project was the use of the "agent" class that Dr. Allen had written and provided to us as a means of manipulating Internet Explorer. Having access to this tool made this project far easier and more manageable. I also made use of

the tools we had downloaded in class for the ribbon modifications. Aside from these things, I did not receive much help from anyone except the internet for an occasional question about syntax or other minor details. The one person that I did consult was my wife, who knows nothing about programming, purely as a resource for an opinion on aesthetics and presentation.