Automate the Creation of Google Calendar Events

VBA Final Project
Phil Shorten

Executive Summary

The BYU Intramurals office organizes and runs dozens of sports leagues during each season. These leagues include a number of sports ranging from football or soccer to inner tube water polo or badminton. The intramurals office publishes a schedule of each team's games on its website which contains all relevant information about each game including date, time, location and the opposing team's name. The system I've programmed will open the intramurals scheduling page on its website and allow the user to select his/her sport, division and team from a series of drop down lists. The program then imports the selected team's schedule into a new tab in the spreadsheet and creates a new Google calendar event (each game in its own Internet Explorer window) for each game in the schedule. All of the information from the schedule is automatically populated in the Google Calendar event form. The user can then review the information, invite additional teammates if desired and save the new event to his/her calendar.

While choosing a project, one of my main criteria was to do something that could be useful to BYU students but could also be applied in other useful ways both now and in the future. I decided that if I could use VBA to interact with Google calendar to automate the creation of calendar events/invitations then this would satisfy my criteria. It will save students who choose to participate in intramurals the time it takes to create individual events and reduces the possibility of making data entry errors. Working with the automation of Google Calendar events could have applications at home, in church callings and at work.

Implementation

Step 1: Import the Schedule
The first step was to navigate a web browser to the intramurals game schedules page. Using the 'agent' class written by Dr. Allen I created a new agent and used the openpage method to

navigate to the page at http://intramurals.byu.edu/vp.php?eval=studentschedules.season. This can be triggered by running the macro manually, or by simply clicking on the image of the "Y" on the "Instructions" worksheet (see Figure 1).

Figure 1: Introduction and Instructions

The program then effectively waits until the user has used the drop down menus to find the desired schedule and clicked on the "View Schedule" button (see Figure 2). This is done by entering a 'Do' loop which looks at the url each time through the loop. When the last 17 characters of the url are set to "viewScheduleFrame" this is an indication that the user has clicked the button and navigated to the page containing the desired schedule.

Figure 2: Schedule Selection Page

The program then uses the 'agent' class to save the source code to the same location as the current workbook, then creates and imports two copies of the outerhtml, which contains a table with the user's schedule, into two new worksheets in the workbook. One of these worksheets is used only by the program and is never seen by the user. It remains unseen by the user because screen updating is turned off before it is created, and the worksheet is deleted before screen updating is reinitiated. While it is invisible, the data is manipulated and formatted in this hidden worksheet so that it can be used with Google calendar. The second copy of the schedule is not deleted, manipulated or hidden. It appears in a new worksheet for the user upon completion of the program and is left in the same format as is found on the web page.

	A	B	C	D	E	F
1	Season Schedule					
2	Al-Quds[3] in Coed Innertube Waterpolo * Div II - Elite 8					
3	Game	Opponent	Date	Day	Time	Location
4	1	Bluth Bananas (2)[6]	1/11/2011	Tue.	7:30 PM	Lap Pool North
5	2	Team 2[2]	1/19/2011	Wed.	8:15 PM	Dive Tank
6	3	Water Furies[5]	1/28/2011	Fri.	8:15 PM	Dive Tank
7	4	Team Theo[1]	2/15/2011	Tue.	7:30 PM	Dive Tank
8	5	The Family[4]	2/23/2011	Wed.	9:00 PM	Lap Pool North

Figure 3: Imported copy of Intramurals schedule

Step 2: Format the Data for Compatibility with Google Calendar

Google has designed Google calendar such that it can accept CGI parameters added to the end of a URL to create an event. CGI parameters are snippets of code which tell the Google calendar form how to populate each field for a new calendar event. These snippets control the fields for editing the date, time, location, description and name fields. Examples of the CGI snippets are found in figure 4.

Parameter Name	Value	Example
action (required)	This value is always TEMPLATE (all capitalized).	**action**=TEMPLATE
text (required)	Event title.	**text**=Brunch at Java Cafe
dates (required)	Date and time of the event, in UTC format. Append a capitalized letter "Z" to the end of times. Google Calendar will interpret the date and time for the user's time zone.	**dates**=20060415/20060416 for all day, April 15th 2006 **dates**=20060415T180000Z/20060415T190000Z for April 15th 2006 11:00am - noon Pacific Time
sprop (required)	Information to identify your organization, like your website address. Multiple sprop parameters are allowed. This information should be specified as type:value. The colon character should only be used to separate type and value.	**sprop**=website:www.javacafebrunches.com for website = www.javacafebrunches.com **sprop**=website:www.javacafebrunches.com& sprop;=name:Java Cafe for website = www.javacafebrunches.com and name = Java Cafe
add	Email address of the guest to invite. Multiple add parameters are allowed.	**add**=username1@domain.com for one guest **add**=username1@domain.com& add;=username2@domain.com for two guests
details	Description of the event. Simple HTML is allowed.	**details**=Try our Saturday brunch special: French toast with fresh fruit Yum!
location	Where the event will take place. Locations that work as Google Maps queries are recommended.	**location**=Java Cafe, San Francisco, CA
trp	Specifies whether the user's Google Calendar shows as "busy" during this event. The default value is false.	**trp**=true

Figure 4: CGI Snippets
Source: http://www.google.com/googlecalendar/event_publisher_guide_detail.html

When all of the pieces are placed in the URL as described above, the result is a very long but useful URL:

http://www.google.com/calendar/event?action=TEMPLATE&text=Intramurals%20game&dates=20110122T170000Z/20110122T180000Z&details=Opponent:%20DP's%20Crew[59]&location=East%20Smith%20Fieldhouse&trp=true&sprop=http%3A%2F%2Fbyu.intramurals.byu&sprop=name:BYU Intramurals
Figure 5: Example of Full URL String

Most of the fields were very easy to convert to the format outlined in the table in figure 4; the only change necessary was to replace all spaces with “%20”. This was done easily by using the ‘replace’ function in VBA.

The dates and times were more of a challenge as they needed to be converted into Coordinated Universal Time (UTC). UTC requires that the date and time be combined into a single string of numbers in 24-hour time, and based on Greenwich Mean Time. I wrote a subprocedure called ‘dateTimeConvert’ which manipulates the data in the worksheet to conform to UTC requirements (figure 6). Additional detail about this data conversion process is found below in the Learning and Conceptual Difficulties section.

Season Schedule																	
MO2V8ED[4] in Men's Flag Football		Start Date/Time								End Date/Time							
Game	Opponent	day	month	year		hr	min	sec		day	mon	year	hr	min	sec		location
1	I Am Tyler Clark[1]	03	02	2011	Tue.	01	00	00	PM	03	02	2011	02	00	00	PM	North University Field #1
2	Cheetahs[6]	03	09	2011	Tue.	01	00	00	PM	03	09	2011	02	00	00	PM	North University Field #1
3	Leah Wright[2]	03	12	2011	Sat.	16	00	00	AM	03	12	2011	17	00	00	AM	North University Field #3
4	107th Ward[5]	03	19	2011	Sat.	16	00	00	AM	03	19	2011	17	00	00	AM	North University Field #1

Figure 6: Formatted Data in Hidden Sheet for UTC Compatibility (color and labels added for clarity)

Step 3: Concatenate Data and Build URL String
Next, the 'buildString' function concatenates all of the required cells into a single string of characters which represent the start time/date and end time/date of the event like this:

dates=20110122T170000Z/20110122T180000Z

The dates code and all of the other CGI snippets are then used by the 'buildString' function which returns an entire string ready to be pasted into a URL, like the one above in figure 5.

Step 4: Create Google Calendar Events
Using the agent class again, I created new agent objects, one for each game, and used a 'for' loop to create a new IE window for each game.

```
firstRow = curRow

Do While Cells(curRow, 1) <> ""
    dateTimeConvert
    curRow = curRow + 1
Loop

totalRows = curRow - 4

curRow = firstRow

For x = 1 To totalRows
    createCalendarEntry
    curRow = curRow + 1
Next x
```

Figure 7: Code which Performs the Data Manipulation and Calendar Event Creation

The createCalendarEntry subprocedure calls the buildString function which returns the string needed for the URL. The createCalendarEntry subprocedure then opens an IR window for each new calendar event, passing it the appropriate information from the buildString function into the URL which, in turn, populates the form on the 'Create Event' page of the Google calendar event form (Figure 8).

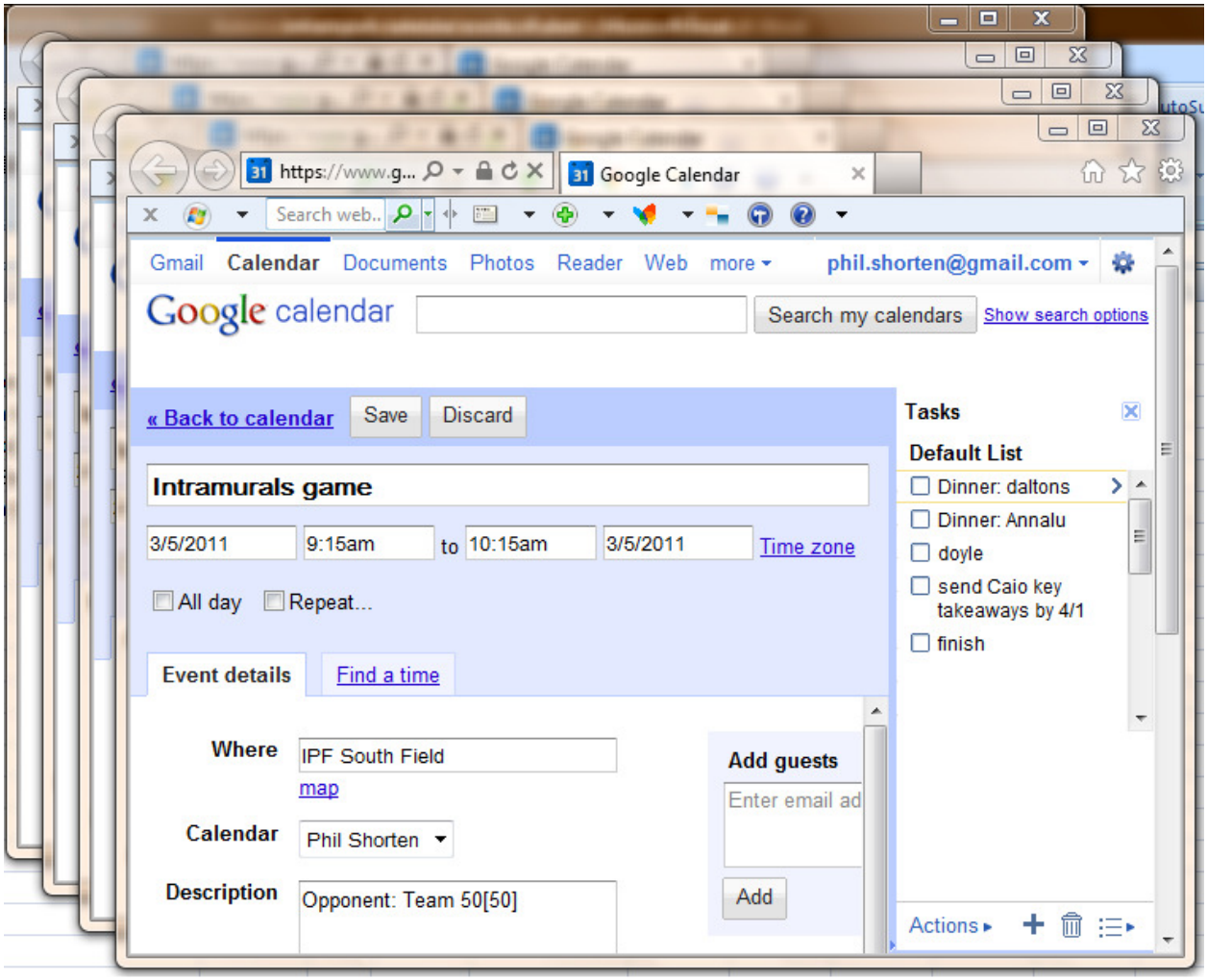

Figure 8: Each of the Four Windows Created with Calendar Events Pre-Populated

The example in figure 8 above was populated with the following string: https://www.google.com/calendar/render?action=TEMPLATE&text=Intramurals+game&dates=20110305T161500Z/20110305T171500Z&details=Opponent:+Team+50%5B50%5D&location=IPF+South+Field&trp=true&sprop=http://byu.intramurals.byu&sprop=name:BYU+Intramurals&gsessionid=OK&sf=true&output=xml

Step 5: End Program

Because I chose to leave the windows open until the user closes them, but did not want the program to continue running unnecessarily, I modified the agent class just slightly, commenting out the call to the terminateIE subprocedure which would've closed the IE window. This allows the program to end and lets the user review the information in each calendar event, make any additions or modifications she desires, then save the information to her calendar.

Just before terminating, the program turns screen updating back on, which reveals the extra copy of the schedule imported into a new worksheet called 'my schedule.' This may be helpful

if the user finds that she needs to access the schedule again and prefers to do this via a local file rather than navigating through the intramurals website.

Learning and Conceptual Difficulties

User Forms?
Many of the most time-consuming problems never saw the light of day in the final version of the project. A good example of this is my idea to use user forms. I got pretty hung up on the idea of trying to use VBA to create all of the inputs, modifications and outputs of data and didn't think through this process very well. I thought that I would import the data in the drop down menus on the intramurals website regarding sport, division and team into a user form, then allow the user to do the exact same thing that he would otherwise do on the website. A quick conversation with Dr. Allen helped me see that this would be redundant, unnecessary work and would add no additional value for the user. The decision was made to allow the user to select his own team on the website, and simply have the program wait until that action was taken before proceeding.

Modifying the Agent Class
During my meeting with Dr. Allen, he anticipated a problem with the agent class that I didn't know that I had yet. He explained that the class would have to be modified slightly by telling the agent class where to look for the 'view schedule' page on the intramurals. This was necessary because of the loop we used to wait for the 'view schedule' page to load once the user has selected his team. Dr. Allen made the necessary changes, which was very helpful because even though I understood what he was doing conceptually, I certainly don't understand how the agent class works well enough to make modifications to it, other than the very simple one I described in Step 5 above.

Converting to UTC Time
The most time consuming part of the project was trying to take a time and date like 2/17/2011 7:00 PM, and turn it into a valid UTC start time and end time. This was a problem that seemed to get more complicated the more I worked on it. The biggest issue is that the time needs to be converted to Greenwich Mean Time which is seven hours ahead of MDT. Since most intramurals games are in the evenings, this means that the conversion doesn't consist of simply adding seven hours to the "hours" field, but also has to handle updating the day, which has to handle many other implications. These include changing the month, if it's the last day of the month, and everything that comes with determining when the last day of the month is, including whether the month has 28, 29 (leap year was particularly painful), 30 or 31 days.

My guess is that there's a more efficient and less frustrating way to solve this issue, but I doubt there's a less educational one. I put each data point (day, month, year, hour, minute, second) into its own cell, converted them all to numbers, then proceeded to make the changes with a large number of 'if then' statements.

I'm sure that there's a way to manipulate these items as dates and times, which would've saved me a lot of time, but after searching for a couple of hours for a way to do this that made sense to me and that I could apply to my code (and get it to run) I gave up and decided to do it myself. Because Dr. Allen had already helped me to solve many of my other major issues, and alleviated much of the pain of doing this myself, I decided that I would like to tackle the date/time conversion myself, and am glad I took the time to learn it.

Differences between the Proposal and Final Deliverable

There are two differences between my proposal and the final deliverable, both of which were approved by Dr. Allen:

1) As mentioned before, I decided not to use a user form to allow the user to select her sport, division and team from drop down menus, as this would've essentially duplicated the functionality of the web site without adding any real value.

2) The other piece that I considered doing, was adding functionality which could send a text message to the user as a reminder before her games. In my help session with Dr. Allen we decided that because a similar functionality is already available in Google calendar which allows the user to choose to receive an email reminder, this added feature in the project wouldn't add additional value.

   If, however, I decided to create this functionality later for this or some other application, I could easily apply the code we went through in class to send text messages with VBA.