Paper 090-2012

Using SAS® to Get a Date: Integrating Google® Calendar's API With SAS

William G. Roehl, CrossUSA, Burnsville, MN

ABSTRACT

Google offers a powerful API that allows the more analytically and visually powerful SAS to interact with and manipulate calendar data. By combining the ease of Google calendaring with the power of SAS, businesses can operate more efficiently, increase customer satisfaction, and analyze their calendar data in nearly limitless ways.

This paper illustrates how SAS can be used to create, modify, and directly interact with Google Calendar data via API calls. Using these calls, it is possible to retrieve available calendars; import all calendar data such as date, time, location, etc.; as well as add, delete, and modify preexisting calendars and their entries. This data can then be analyzed in any number of ways available through the power of SAS.

INTRODUCTION

Online calendars are easily accessible from multiple locations, difficult to misplace on a park bench, and render bad handwriting a problem of the past. In the business world, an online calendar serves many useful purposes, from scheduling ongoing reports to allowing customers to see project deadlines in an easy to understand fashion.

Calendars are extremely useful tools for easily visualizing time-related information in a format which is understandable by many. While publishing information to shared calendaring software has always been fairly easy with the iCalendar standard file format, retrieving data from the calendaring software has been a bit more difficult`; especially when you are looking to automate the process. Thankfully Google provides a well documented API to interact directly with their calendaring solution which allows us to push and pull data from Google Calendar in and out of SAS.

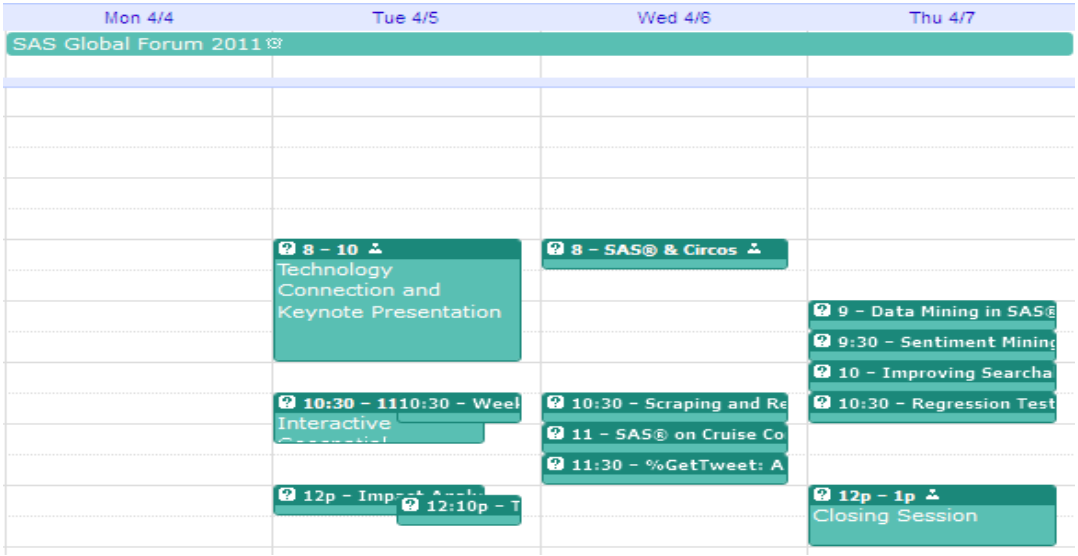
This paper illustrates how SAS can be used to create, modify, and directly interact with Google Calendar data via API calls. Using these calls, it is possible to retrieve available calendars; import all calendar data such as date, time, location, etc.; as well as add and delete calendars and their entries. This data can then be analyzed in any number of ways available through the power of SAS.

While the provided macros included with this paper do not replicate the available API functions in their entirety, the majority of calendaring functionality is available and includes pulling calendar lists and event data, creation and deletion of calendars and events.

The intended audience for this paper are SAS developers who have an interest in adding calendaring support to their code, are interested in integrating GData related APIs into their work, and who may be interested in manipulating Google Calendar data from within SAS.

Code was developed with SAS 9.1.3, cURL 7.20.1-SSL, and HTML Tidy (25MAR2009) running under Microsoft Windows® XP Professional.

GOOGLE CALENDAR ON THE WEB



Display 1. Sample Google Calendar Weekly View

'Display 1' above is a portion of the week-long view of SAS Global Forum 2011 in Las Vegas, NV from a Google Calendar account. The image shows what one SAS Global Forum attendee may have planned to see. While there were far more sessions than what is shown above, it gives the viewer a decent idea of the way the software presents your data to you on the web. However, while this format is easy for human viewing, it is certainly not conducive to any sort of automated analysis and obviously not meant for software to interact with the data. Thankfully by using the API and cURL through a few easy to understand SAS macros, interacting with the data becomes as easy as a few lines of code and 'F3'.

A quick example of how to utilize this code to interact with the Google Calendar API would be to pull down the list of events shown above in 'Display 1' during SAS Global Forum 2011. The SAS code would require two macros calls to meet this end result. The first, %GCAL_AUTH, handles the authentication of the user.

%GCAL_AUTH

Parameter	Description
U	Google Accounts username (generally a GMail address)
P	Google Accounts password

Macro code:

```
%macro gcal_auth (u,p);
        %global auth;
        %let AUTHURL = c:\sas\curl.exe -o c:\temp\gcal-resp.txt -k -s -L
                        --url https://www.google.com/accounts/ClientLogin
                        --data-urlencode Email=&u.
                        --data-urlencode Passwd=&p.
                         -d service=cl;

        options noxwait;
        x "&AUTHURL";

        filename response 'c:\temp\gcal-resp.txt';
        data _null_;
                infile response lrecl=500 dlm='=';
                format var $8. value $500.;
                input var value;
                if var = 'Auth';
                put value=;
                call symputx('AUTH',trim(value));
        run;
%mend gcal_auth;
```

Example:

```
/* Authenticate */
%gcal_auth(username@gmail.com,password);
```

The %GCAL_AUTH macro builds a URL from the arguments passed (username, password) and passes them to cURL to execute. While SAS has its own built in PROC HTTP function which is generally the preferred method for this type of call, because Google's APIs require HTTPS, it is far easier to go this route instead.

Once the URL is passed and the username and password are accepted, Google returns an authentication value which is stored in the global macro variable &AUTH for later use.

The second macro called is the is the %GCAL_GETEVENTS macro. This macro retrieves Google Calendar data from the API, parses the returned XML and places the resulting data in a dataset for use in whatever way the SAS user can dream up.

%GCAL_GETEVENTS

Parameter	Description
URL	The URL to pass through to the Google Calendar API

OUTPUT	The name of the SAS data set to output the parsed XML data

Example:

%***gcal_getEvents***(url=https://www.google.com/calendar/feeds/username%**40**gmail.com/***private***/full?start-min=**2011**-**04**-**03**&start-max=**2011**-**04**-**09**&max-results=**50000**,output=SASGF2011);

The %GCAL_GETEVENTS macro builds a URL to pass to the API which will eventually return an XML file containing the event data requested. In the example above the call is made to request all the events (up to a maximum of 50,000 for the dates encompassing SAS Global Forum 2010. The returned XML includes all of the data on the 29 events on the test calendar as provided by the API. While SAS is able to read XML files into datasets using XML Mapper, the included code for this project instead uses regular expressions to parse the returned XML into a dataset.

Macro code excerpt:

```
/* Fix the XML and make it human readable */
data gcal_calendar_data(where=(id) drop=lineT);
        set gcal_calendar_data;
        format calendar_name calendar_id calendar_url calendar_desc $256.
                id $10.;

        /* We have a new calendar entry, increment the rank variable */
        if prxmatch('/<entry>/',line) then rank + 1;

        /* IF blocks handle parsing of XML - fairly self explanatory */
        if prxmatch("/<title type='text'>/",line) then do;
                line = prxchange("s/<title type='text'>//",-1,line);
                line = prxchange('s/<\/title>//',-1,line);
                id = 'title';
                output;
        end;
        if prxmatch ('/content type/',line)
          and line ne "<content type='text' />" then do;
                line = prxchange('s/^.*text.>//',-1,line);
                line = prxchange('s/<\/content>.*$//',-1,line);
                id = 'desc';
                output;
        end;

        retain rank;
run;
```

Simple regular expressions search for and modify lines of XML read in by SAS to remove the XML tags and leave only the wanted information to save in the dataset. First the PRXMATCH function is called to find the XML tag desired and then the PRXCHANGE function is called twice. The first PRXCHANGE call removes the text of the XML tag leading up to the desired information and the second removes everything after the desired information up through the end of the line. The “id = ‘foo’” line creates a new variable to name each piece of information and is later used by the TRANSPONSE procedure to put all the final information in one row per calendar entry.

Example reporting dataset created (subset):

Title	Desc	Start	End
My SASGF11 Talk!	**Mine!**	**2011-04-06T10:30:00.000-05:00**	**2011-04-06T10:50:00.000-05:00**

Table 2. %GCAL_GETEVENTS macro table output

In addition to the two examples provided here, similar SAS code runs behind the other API calls currently available as part of this submission:

%GCAL_GETLIST – pulls a list of all the calendars available to the current authenticated user

%GCAL_CREATECALENDAR – creates a new Google calendar (not an event, a calendar)

%GCAL_DELETECALENDAR – deletes a preexisting Google calendar

%GCAL_CREATEEVENT – creates an individual calendar event

Using the included example code for each of the previously defined macros other SAS programmers should be able to easily develop the remainder of the API calls as their own program requirements demand.

REFERENCES

- Daniel Stanberg. (2011). cURL Manual. http://curl.haxx.se/docs/manual.html
- Google Inc. (2011). Google Analytics Developer Docs. http://code.google.com/apis/calendar/data/2.0/
- Dirk Paehl. (2009). HTML Tidy for Win32. http://www.paehl.com/open_source/?HTML_Tidy_for_Windows

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

William Roehl
CrossUSA
13754 Frontier Ct #106
Burnsville, MN 55337
broehl@cross-usa.com
http://www.cross-usa.com

APPENDIX

SAS <-> GOOGLE CALENDAR API SOURCE CODE

```
/*************************************************************************
\PROGRAM INFORMATION
Project : SAS Global Forum 2012 Submission
Purpose : Interact with the Google Calendar API
Inputs  : GCal API
Outputs : GCal API, Tables
Notes   : Utilizes cURL and tidy

PROGRAM HISTORY
2011-09-06 WR   Initial program developed.
2012-03-02 WR   Changes to API
                Changes to code to work with 9.3 deployment at CrossUSA.
*************************************************************************\;

/* Authenticate and grab the AUTH variable for later use (globally) */
%macro gcal_auth (u,p);
        %global auth;
        %let AUTHURL = c:\sas\curl.exe -o c:\temp\gcal-resp.txt -k -s -L
                        --url https://www.google.com/accounts/ClientLogin
                        --data-urlencode Email=&u.
                        --data-urlencode Passwd=&p.
                         -d service=cl;

        options noxwait;
        x "&AUTHURL";

        filename response 'c:\temp\gcal-resp.txt';
        data _null_;
                infile response lrecl=500 dlm='=';
                format var $8. value $500.;

                input var value;
                if var = 'Auth';

                put value=;
                call symputx('AUTH',trim(value));
        run;
%mend gcal_auth;

/* Retrieve user calendar list */
%macro gcal_getList ();
        %let GCAL_CALL = c:\sas\curl.exe -ksLo c:\temp\gcal-resp.txt
        --header "Authorization: GoogleLogin auth=&AUTH."
        --url https://www.google.com/calendar/feeds/default/allcalendars/full
          && c:\sas\tidy.exe -xml -indent -wrap 256 -quiet -m
          c:\temp\gcal-resp.txt;

        options noxwait noquotelenmax;
        %sysexec &GCAL_CALL.;

        /* Read API's XML response into a dataset */
        filename response 'c:\temp\gcal-resp.txt';
        data gcal_calendar_list;
                format line $256.;
                infile response lrecl=256 DLM='0A'x;;
                input line;
        run;

        /* Grab URLs for the calendars and strip extraneous tags */
        data gcal_calendar_list(where=(id));
                set gcal_calendar_list;
```

```
                format id $25.;

                /* New calendar entry */
                if prxmatch('/<entry>/',line) then rank + 1;

                /* ID */
                if prxmatch ('/<id>/',line) then do;
                        line = prxchange('s/^.*<id>//',-1,line);
                        line = prxchange('s/<\/id>.*$//',-1,line);
                        id = 'calendar_id';
                end;

                /* Name of calendar */
                if prxmatch ('/title type/',line) then do;
                        line = prxchange('s/^.*text.>//',-1,line);
                        line = prxchange('s/<\/title>.*$//',-1,line);
                        id = 'calendar_name';
                end;

                /* Calendar description - optional */
                if prxmatch ('/summary/',line) and
                  line ne "<summary type='text' />" then do;
                        line = prxchange('s/^.*text.>//',-1,line);
                        line = prxchange('s/<\/summary>.*$//',-1,line);
                        id = 'calendar_desc';
                end;

                /* Feed URL */
                if prxmatch ('/<content type=.*application.*atom.xml/',line) then
                do;
                        line = prxchange("s/^.*src=.?//",-1,line);
                        line = prxchange("s/full.*$/full/",-1,line);
                        id = 'calendar_url';
                end;

                /* Put each calendar URL entry on a single line */
                proc transpose data=gcal_calendar_list let
                                        out=gcal_calendar_list(drop=_NAME_ rank where=(calendar_url ne ''));
                        var line;
                        id id;
                        by rank;
                run;
        run;
%mend gcal_getList;

/* Retrieve calendar data */
%macro gcal_getEvents (URL=,output=);
        %let GCAL_CALL = c:\sas\curl.exe -ksLo c:\temp\gcal-resp.txt
                          --header "Authorization: GoogleLogin auth=&AUTH."
                          --url "&URL."
                          && c:\sas\tidy.exe -xml -indent -wrap 256 -quiet -m c:\temp\gcal-resp.txt;

        /* Delete the prior response file and get a new one */
        x 'del c:\temp\gcal-resp.txt';
        options noxwait;
        %sysexec &GCAL_CALL.;

        /* Read the XML into a dataset */
        filename response 'c:\temp\gcal-resp.txt';
        data gcal_calendar_data;
                format line $2048.;
                infile response lrecl=2048 DLM='0A'x;
                input line;
        run;

        /* Fix the XML and make it human readable */
```

```
data gcal_calendar_data(where=(id) drop=lineT);
	set gcal_calendar_data;
	format calendar_name calendar_id calendar_url calendar_desc $256. id $10.;

	/* We have a new calendar entry, increment the rank variable */
	if prxmatch('/<entry>/',line) then rank + 1;

	/* The IF blocks which follow handle XML parsing */
	if prxmatch("/<title type='text'>/",line) then do;
		line = prxchange("s/<title type='text'>//",-1,line);
		line = prxchange('s/<\/title>//',-1,line);
		id = 'title';
		output;
	end;
	if prxmatch ('/content type/',line) and
	  line ne "<content type='text' />" then do;
		line = prxchange('s/^.*text.>//',-1,line);
		line = prxchange('s/<\/content>.*$//',-1,line);
		id = 'desc';
		output;
	end;
	if prxmatch('/endTime/',line) then do;
		lineT = line;
		line = prxchange('s/^.*endTime=?//',-1,line);
		line = prxchange('s/.startTime.*$//',-1,line);
		line = prxchange("s/'//",-1,line);
		id = 'end';
		output;
	end;
	if prxmatch('/startTime/',lineT) then do;
		line = lineT;
		line = prxchange('s/^.*startTime=?//',-1,line);
		line = prxchange("s/'|>|\///",-1,line);
		id = 'start';
		output;
	end;
	if prxmatch('/<published>/',line) then do;
		line = prxchange('s/<published>//',-1,line);
		line = prxchange('s/<\/published>//',-1,line);
		id = 'published';
		output;
	end;
	if prxmatch('/<updated>/',line) then do;
		line = prxchange('s/<updated>//',-1,line);
		line = prxchange('s/<\/updated>//',-1,line);
		id = 'updated';
		output;
	end;
	if prxmatch('/<name>/',line) then do;
		line = prxchange('s/<name>//',-1,line);
		line = prxchange('s/<\/name>//',-1,line);
		id = 'name';
		output;
	end;
	if prxmatch('/<email>/',line) then do;
		line = prxchange('s/<email>//',-1,line);
		line = prxchange('s/<\/email>//',-1,line);
		id = 'email';
		output;
	end;
	if prxmatch('/<id>/',line) then do;
		line = prxchange('s/<id>//',-1,line);
		line = prxchange('s/<\/id>//',-1,line);
		id = 'id';
		output;
	end;
```

```
		retain rank;
	run;

	/* Put each calendar entry on a single line */
	proc transpose data=gcal_calendar_data let
			out=gcal_calendar_xpose(drop=_NAME_ rank);
		var line;
		id id;
		by rank;
	run;

	/* Reorder dataset variables */
	proc sql;
		%if "&OUTPUT" ne "" %then %do;
			create table gcal_calendar_&OUTPUT. as %end;
		%if "&OUTPUT" eq "" %then %do;
			create table gcal_calendar as %end;
		select distinct title
				, start
				, end
				, published
				, updated
				, id
				, name
				, *
		from gcal_calendar_xpose
		order by start, end
	;
	quit;

%mend gcal_getEvents;

/* Create a new calendar */
%macro gcal_createCalendar(name=,desc=,tmz=,hide=,color=,location=);

	%let GCAL_POST_CALL = c:\sas\curl.exe –ksL
	--header "Authorization: GoogleLogin auth=&AUTH."
	 -X POST
	--url https://www.google.com/calendar/feeds/default/owncalendars/full
	 -H "Content-Type: application/json"
	--data @c:\temp\payload;

	data _null_;
		file 'c:\temp\payload';

		put '{';
		put '  "data": {';
		%if "&DESC." ne "" %then %do;
			put '  "details": "' "&DESC." '",'; %end;
		%if "&TMZ." ne "" %then %do;
			put '  "timezone": "' "&TMZ." '",'; %end;
		%if "&HIDE." ne "" %then %do;
			put '  "hidden": "' "&HIDE." '",'; %end;
		%if "&COLOR." ne "" %then %do;
			put '  "color": "' "&COLOR." '",'; %end;
		%if "&LOCATION." ne "" %then %do;
			put '  "location": "' "&LOCATION." '"'; %end;
		put '  "title": "' "&NAME." '",';
		put ' }';
		put '}';
	run;

	options noxwait;
	%sysexec &GCAL_POST_CALL.;
%mend gcal_createCalendar;
```

```
/* Create a new calendar event */
%macro gcal_createEvent(url=,name=,desc=,location=,start_dt=,end_dt=);
	%let GCAL_POST_CALL = c:\sas\curl.exe -ksL
				--header "Authorization: GoogleLogin auth=&AUTH."
				 -X POST
				--url "&URL." -H "Content-Type: application/json"
				--data @c:\temp\payload;

	data create_event;
		file 'c:\temp\payload';
		put '{';
		put '  "data": {';
		%if "&DESC." ne "" %then %do;
			put '  "details": "' "&DESC." '",'; %end;
		put '  "transparency": "opaque",';
		put '  "status": "confirmed",';
		%if "&LOCATION." ne "" %then %do;
			put '  "location": "' "&LOCATION." '",'; %end;
		put '  "title": "' "&NAME." '",';
		put '  "when": [';
		put '    {';
		put '      "start": "' "&START_DT." '",';
		put '      "end": "' "&END_DT." '"';
		put '    }';
		put '  ]';
		put ' }';
		put '}';
	run;

	%sysexec &GCAL_POST_CALL.;

	x 'del c:\temp\payload';
%mend gcal_createEvent;

/* Delete calendar */
%macro gcal_deleteCalendar(id);

	%let GCAL_POST_CALL = c:\sas\curl.exe -ksL
				--header "Authorization: GoogleLogin auth=&AUTH."
				 -X DELETE
				--url "&ID.";

	options noxwait;
	%sysexec &GCAL_POST_CALL.;

%mend gcal_deleteCalendar;
```