Creating iCalendar Files

Doug Hennig

This is the third of several articles on components of Doug's in-house library. This issue discusses how to create iCalendar files so scheduled items can automatically be added to a calendar.

I needed a way to create iCalendar files for a web site. The idea was that a user could click a link for a scheduled item to automatically add the item to their calendar, either on their computer or their phone.

An iCalendar file is pretty easy to create, as it's just a text file with an ICS extension. Here's an example:

```
BEGIN:VCALENDAR
VERSION:2.0
CALSCALE:GREGORIAN
X-WR-TIMEZONE:America/Phoenix
BEGIN:VEVENT
DTSTART:20121019T170000Z
DTEND:20121019T181500Z
SUMMARY;ENCODING=QUOTED-PRINTABLE:Win32API for
VFP Developers - Doug Hennig
DESCRIPTION;ENCODING=QUOTED-PRINTABLE:The
Windows API (Win32API) contains thousands of
useful functions. However, finding which
function to use when, and how to call it from
a VFP application, can be challenging. This
session discusses how API functions are called
in VFP, where to find information on the API,
and presents lots of useful API functions you
can call in your VFP applications today.
LOCATION;ENCODING=QUOTED-PRINTABLE:Gilbert
UID:SWFOX20121019T170000Z
PRIORITY:3
END:VEVENT
END:VCALENDAR
```

The DTSTART and DTEND elements contain the start and end date and time in UTC (Coordinated Universal Time, http://tinyurl.com/cskd9xu), SUMMARY is the summary or title of the item, DESCRIPTION contains the item's description, LOCATION has location such as a building name or room number, and UID is a unique ID. See http://tinyurl.com/8p55y9x for a more complete specification of the format.

The code to generate the iCalendar text is pretty straightforward: it simply uses text merge to insert the appropriate values into a template string. **Listing 1** shows the code for CreateICSFile.prg, which creates a ICS file when you pass it the following parameters:

- The starting date and time for the item.
- The ending date and time for the item.
- The title.
- The description.
- The location.
- The name and path for the ICS file.

Listing 1. CreateICSFile.prg creates an ICS file from the specified settings.

```
lparameters tdStart, ;
  tdEnd, ;
  tcTitle, ;
  tcDescription, ;
  tcLocation, ;
  tcFile

* Get the time zone offset for UTC.

loTimeZone       = GetTimeZoneInfo()
lnTimeZoneOffset = loTimeZone.TimeZoneOffset

* Generate an ICS file.

lcStart = chrtran(ttoc(tdStart + ;
  lnTimeZoneOffset, 3), '-:', '') + 'Z'
lcEnd   = chrtran(ttoc(tdEnd + ;
  lnTimeZoneOffset, 3), '-:', '') + 'Z'
lcID    = sys(2015)
text to lcContent noshow textmerge pretext 2
BEGIN:VCALENDAR
VERSION:2.0
BEGIN:VEVENT
DTSTART:<<lcStart>>
DTEND:<<lcEnd>>
SUMMARY:<<tcTitle>>
DESCRIPTION:<<tcDescription>>
LOCATION:<<tcLocation>>
UID:<<lcID>>
PRIORITY:3
END:VEVENT
END:VCALENDAR

endtext
strtofile(lcContent, tcFile)
```

The only wrinkle is getting the date/time values in UTC. The best way to do that is to ask Windows for the current time zone information, as I discussed in the March 2016 issue of FoxRockX. **Listing 2** shows the code for GetTimeZoneInfo.prg, which returns an object with TimeZoneDesc (the time zone description) and TimeZoneOffset (the offset from GMT in seconds) properties.

Listing 2. GetTimeZoneInfo.prg asks Windows for the time zone offset from GMT.

```
local loObject, ;
  lcTimeZone, ;
  lnID, ;
  lnStandardOffset, ;
  lnDaylightOffset

* Create an object to hold the time zone
* information.

loObject = createobject('Empty')
addproperty(loObject, 'TimeZoneDesc')
addproperty(loObject, 'TimeZoneOffset')

* Declare the time zone information API
* function and get the time zone
* information.

#define TIME_ZONE_SIZE  172
declare integer GetTimeZoneInformation ;
  in kernel32 ;
  string @lpTimeZoneInformation
lcTimeZone = replicate(chr(0), TIME_ZONE_SIZE)
lnID       = ;
  GetTimeZoneInformation(@lcTimeZone)

* Determine the standard and daylight time
* offset.

lnStandardOffset = ctobin(substr(lcTimeZone, ;
    1, 4), '4RS')
lnDaylightOffset = ctobin(substr(lcTimeZone, ;
  169, 4), '4RS')

* Determine the total offset based on whether
* the computer is on daylight time or not. Get
* the description for the time zone.

if lnID = 2  && daylight time
  loObject.TimeZoneDesc   = ;
    strtran(strconv(substr(lcTimeZone, 89, ;
    64), 6), chr(0), '')
  loObject.TimeZoneOffset = ;
    (lnStandardOffset + lnDaylightOffset) * 60
else   && standard time
  loObject.TimeZoneDesc   = ;
    strtran(strconv(substr(lcTimeZone,  5, ;
    64), 6), chr(0), '')
  loObject.TimeZoneOffset = ;
    lnStandardOffset * 60
endif lnID = 2
return loObject
```

Originally, I just used the following code to get the start date/time and something similar for the ending date/time:

```
ttoc(tdStart + lnTimeZoneOffset, 3) + 'Z'
```

This gave values formatted like this:

```
2012-10-20T21:00:00Z
```

While this worked just fine with Microsoft Outlook, I found a problem when trying to add an item to the calendar on my iPhone. I found a blog post by Joe Bradford that suggested the problem might be the wrong MIME type for ICS files. Checking the IIS settings confirmed I had the same issue as Joe. The easiest way to change the settings for me (since I didn't have access to the IIS settings directly) was to create a web.config file that changed the MIME setting and add it to the root folder of the web site:

```
<configuration>
  <system.webServer>
    <staticContent>
      <remove fileExtension=".ics" />
      <mimeMap fileExtension=".ics"
        mimeType="text/calendar" />
    </staticContent>
  </system.webServer>
</configuration>
```

However, that still didn't resolve the problem; while I didn't get an error, the item simply didn't show up in the calendar. Just for grins, I tried to import an ICS file into Google Calendar; while the item did appear in the calendar, it was in the wrong timeslot (almost a year off). I manually created a calendar item and exported it, and noticed a difference in how the date/time values were formatted: the same as mine but without dashes or colons:

```
DTSTART:20121020T210000Z
DTEND:20121020T221500Z
```

As a result, the code in CreateICSFile.prg now strips out the extra characters inserted by TTOC().

TestCreateICSFile.prg creates an ICS file for a sample appointment from 9:00 a.m. to 10:00 a.m. local time (for the Central Daylight Time zone in North America, that's 1400 to 1500 UTC) on August 1, 2017. It has the following code:

```
ltStart = datetime(2017, 8, 1,  9, 0, 0)
ltEnd   = datetime(2017, 8, 1, 10, 0, 0)
CreateICSFile(ltStart, ltEnd, ;
  'Meeting with Brad', 'Meeting with ' + ;
  'Brad to discuss proposal. Remember ' + ;
  'to bring new customer forms.', ;
  'Boardroom', 'MyMeeting.ics')
```

Figure 1 shows what the appointment looks like when you double-click the ICS file to add it to Microsoft Outlook and **Figure 2** shows how it appears when you import it into Google Calendar.

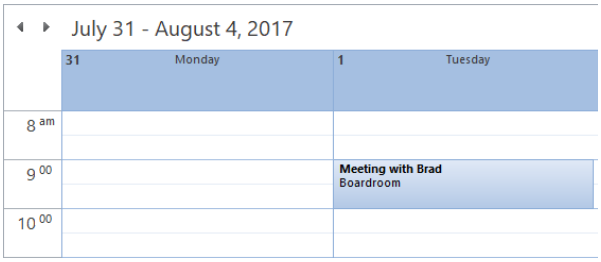

Figure 1. The sample appointment as it appears in Microsoft Outlook.

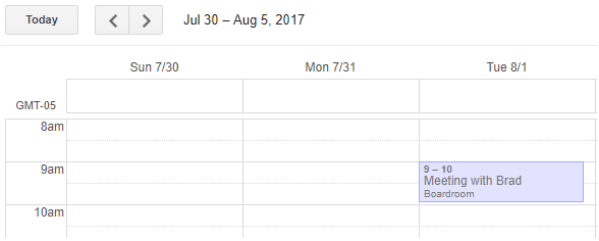

Figure 2. The sample appointment as it appears in Google Calendar.

Summary

Creating iCalendar files is simple. If your application supports scheduled items, such as appointments, meetings, or events, you can easily add the ability to automatically add them to the user's calendar using CreateICSFile.prg.

Doug Hennig is a partner with Stonefield Software Inc. He is the author of the award-winning Stonefield Database Toolkit (SDT); the award-winning Stonefield Query; the MemberData Editor, Anchor Editor, and CursorAdapter and DataEnvironment builders that come with Microsoft Visual FoxPro; and the My namespace and updated Upsizing Wizard in Sedna.

Doug is co-author of "VFPX: Open Source Treasure for the VFP Developer," "Making Sense of Sedna and SP2," the "What's New in Visual FoxPro" series (the latest being "What's New in Nine"), "Visual FoxPro Best Practices For The Next Ten Years," and "The Hacker's Guide to Visual FoxPro 7.0." He was the technical editor of "The Hacker's Guide to Visual FoxPro 6.0" and "The Fundamentals." All of these books are from Hentzenwerke Publishing (http://www.hentzenwerke.com). He wrote over 100 articles in 10 years for FoxTalk and has written numerous articles in FoxPro Advisor, Advisor Guide to Visual FoxPro, and CoDe. He currently writes for FoxRockX (http://www.foxrockx.com).

Doug spoke at every Microsoft FoxPro Developers Conference (DevCon) starting in 1997 and at user groups and developer conferences all over the world. He is one of the organizers of the annual Southwest Fox conference (http://www.swfox.net). He is one of the administrators for the VFPX VFP community extensions Web site (http://vfpx.codeplex.com). He was a Microsoft Most Valuable Professional (MVP) from 1996 to 2011. Doug was awarded the 2006 FoxPro Community Lifetime Achievement Award (http://tinyurl.com/ygnk73h).