# Windows App

## Project: SRCalendarSynchronizerApp

## Deploying

If you make any changes to the Windows App, you need to rebuild the DesployCalendarSynchronizer project. To do this, right-click the project, and take the Build or Rebuild option. The file DeployCalendarSynchronizer.msi will be rebuilt. This is the file that you run on the machine where you want to deploy the app … and double-click it to initiate the install. You can find this file at: <project subdirctory>\SRCalendarSynchronizer\DeployCalendarSynchronizer\Debug

## Running

You can run the Windows App in one of three ways. In all three, you have to concern yourself with the config file … making sure that the settings are correct.

When you run the Windows App, the synchronization action takes place (per the parameters indicated in the window). When you close the window, the synchronizing stops. (Note: this action is independent of the Windows Service. That is to say, if the Windows Service is running, it is synchronizing, and keeps on synchronizing even if the Windows App is closed.)

The pertinent config settings are:

* mysqlConnectionString – Connection string for the MySQL database
* SROCalendarConnectionString – Connection string to the SROCalendar database
* SROUpdateConnectionString – Connection string to the SROUpdate database
* WriteLogEntries – whether or not to write entries in the windows event log (Control Panel, Administrative Tools, Event Viewer) Note: when this is set to “true” entries are logged to the Event Viewer. Currently, there are only a few entries … just enough to make sure that the system is running. There’s one for when the scheduler function starts; one for when it stops; and one each time the timer “pops”. It’s easy to add more entries. Let me know if you’d like me to do so; or, you can do it yourself. There’s a method in the Scheduler class (an instance of which is created by either the Windows App or the Window Service) called: writeEventLogEntry(string);

### From within the development environment (the config file is called app.config)

#### Right-click project SRCalendarSynchronizerApp, and click Set As Startup Project

#### F5 to run in debug mode

#### Ctrl-F5 to run not in debug mode

### Double-click SRCalendarSynchronizerApp.exe found in <projectsubdirctory\SRCalendarSynchronizer\SRCalendarSynchronizerApp\bin\Debug (the config file is called SRCalendarSynchronizerApp.exe.config

### Double-click SRCalendarSynchronizerApp.exe found at the place where you’ve deployed the project. Note that when you deploy the app, the config file is also called SRCalendarSynchronizerApp.exe.config.

# Web App

## Project: SRCalendarSynchronizationWebControl

## Deploying

### Right click project SRCalendarSynchronizationWebControl, and click Publish

### Select the target location

### Press Publish

## Running

The config file (both when run in the development environment, as well as when deployed, is called web.config. The pertinent settings are the same as with the Windows App, but without the WriteLogEntries.

I created a Login page. The values for the id and password are found in SROCalender.dbo. SRCalendarSynchronizationWebControl. Currently these values are id: Jason, pwd: Wp#p18mm.

This app provides, in addition to the same controls as the Windows App, the ability to modify these security settings.

### Run inside the development environment

#### Right-click project SRCalendarSynchronizerApp, and click Set As Startup Project.

#### Right-click Default.aspx, and click Set As Start Page

#### F5 to run in debug mode

#### Ctrl-F5 to run not in debug mode

### Deploy it onto a web server, and run it from there.

# Windows Service

The Windows Service and the Windows App perform the exact same function with respect to periodically synchronizing the calendars. The only difference between the two is that the Windows Service doesn’t have any user interface (so, you have to rely on the Web App for that).

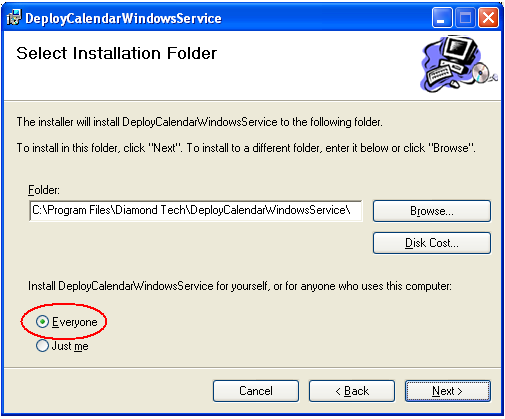
## Deploying

### Project: DeployCalSynchWindowsService

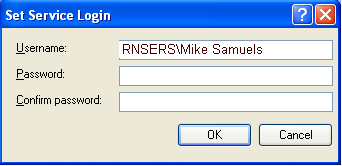
If you make any change to the code, you should rebuild the DeployCalSynchWindowsService project

### In the directory called <.net project>\SRCalendarSynchronizer\DeployCalSynchWindowsService\Debug there is a file called DeployCalSynchWindowsService.msi. Copy this file over to the server where you wish to run the Windows Service, and at that server, double click it; and then follow the installation instructions. Note two things:

#### At the Select Installation Folder panel, click Everyone



#### At the Set Service Login panel, be sure to include the computer name as part of the Username. I keyed in RNSERS\Mike Samuels.



## Running

### When the service is installed, you’ll see it in the Services screen (Control Panel, Administrative Tools, Services). Its status will be empty. To start it manually (or to stop in manually), right click on the service, and click the start (or stop). Its Startup Type will be “automatic”, which means that it will start by itself whenever the computer boots up. You can change that behavior if you want (right-click, properties).