SessionTime - log Windows session events and display session durations and locked periods

[**Ivan Golović**](https://www.codeproject.com/script/Membership/View.aspx?mid=11176433)

19 Aug 2019 [MIT license](https://opensource.org/licenses/MIT)

This application consists of Windows service that logs Windows session events and WPF application that parses resulting log to display session durations and locked/unlocked periods within sessions for selected user.

## **Introduction**

This article describes a possible approach to logging session events on Windows operating system. It also parses event data to display session durations and locked/unlocked periods. Following session events are logged: logon, logoff, lock, unlock.

Source code might be useful to developers who need working source code to help them with their own Windows session logging solution. Windows service and app might be useful if you, say, want to monitor how much time you spend on the computer or at which time you logged in and started working.

The application consists of three parts:

* SessionTimeMonitor - Windows service that logs events
* SessionTimeViewer - Windows application that displays parsed session durations and locked/unlocked periods within sessions
* SessionTimeSetup - Installer/uninstaller for Windows service and application

### **SessionTimeMonitor**

Main part is a Windows service called SessionTimeMonitor - it handles the OnSessionChange event which is triggered each time when a session change occurs. When event is triggered, record is added by the service to the XML file on the disk. Record contains information about the event, such as datetime, type of event, etc.

### **SessionTimeViewer**

SessionTimeViewer application opens and parses XML file that contains data records written by the SessionTimeMonitor. It displays list of sessions with their start and end datetime along with the locked/unlocked periods within a single session. SessionTimeViewer is implemented as a WPF application.

## **Background**

I needed this application to determine how much time I spend on the computer and I wanted to have a customizable application. An alternative approach was to write an application that parses Windows event logs but I concluded that parsing Windows event logs and pairing logon/logoff and lock/unlock events was quite cumbersome.

## **Using the Code**

### **Development Tools**

Initial version of Windows service and WPF application were developed using .NET Framework 4.0 and Visual Studio 2010.

### **How It Works**

The following illustration shows how this approach works:

SessionTimeMonitor (Windows service)

-> writes to - >

XML data file

-> is read and parsed by - >

SessionTimeViewer (Windows application)

### **Handling the OnSessionChange Event**

The main part of the solution is a Windows service that logs session events to the XML data file, this is done through overriding the OnSessionChange event.

Log is kept in the form of XML file stored on disk.

### **XML Data Record**

Each logged event is represented by SessionTrackingParams XML element. Such element is added to log when the service starts (during the service's OnStart event) for each existing session or when session changes (when OnSessionChange event occurs) only for session that changed. Such events are logged in SessionTimeMonitor service and serialized to the list stored in XML data file.

**Displaying Session Duration and Locked/Unlocked Periods**

Parsing of XML data file is performed by SessionTimeViewer which is a Windows WPF application that shows following types of data:

* sessions - as periods between logon and logoff event within same service run marked by ServiceRunGuid
* unlocked periods - as periods within same session between:
  + logon and logoff
  + logon and lock
  + unlock and logoff
  + unlock and lock
* locked periods - as periods within same session between:
  + lock and unlock
  + lock and logoff

This way, it is possible to see when and how long certain user has been logged on and when locked or unlocked periods happened.

## **Points of Interest**

* SessionTimeMonitor utilizes the Cassia library which is "a .NET library for accessing the native Windows Terminal Services API (now the Remote Desktop Services API)" ([https://code.goog le.com/p/cassia/](https://code.google.com/p/cassia/), 2016-01-06), it uses its GetSessions method to obtain session data
* SessionTimeViewer displays only session for which the logoff event exists, this means that current session along with its locked/unlocked periods is not displayed until logoff
* When computer starts up, it is possible that SessionTimeMonitor:
  + starts before user logon - in such case, user logon will be logged through OnSessionChange event
  + starts after the user logon - in such case, all existing sessions are logged through service's OnStart event
* SessionTimeMonitor and SessionTimeViewer are using config files to set path of XML data file, so those paths are configurable
* Service and application have been tested on Windows 7 (32 bit), Windows 8.1 (64 bit), Windows 10 (64-bit)
* For initial version of application prerequisite on client computer is .NET Framework 4.0

## **History**

* 1.0.0.0 - 2016-06-01 - Initial version
* 1.0.0.1 - 2016-06-01 - Bugfix for version 1.0.0.0, previous version wasn't logging logoff event correctly due to last-minute untested change, sorry to all the folks who downloaded version 1.0.0.0
* 1.0.0.2 – various bugfixes, user selection, Wix setup introduced

## **License**

Copyright 2019. Ivan Golović

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.