Flight Data Viewer

Version 1.1 Documentation

Eric Burlingame

Winter 2010

Flight Data Viewer Version 1.1

Free, Open-Source Data Viewer for MS Flight Simulator

This software makes use of FSUPIC for its data retrieval. Also, because it is written in Visual C# it relies on Microsoft's .NET Framework for its interface and compiler. .NET is very common and is already installed on most to all windows machines.

Terms of Usage

You are free to change, edit, and improve the source code of the software as you please. You are free to distribute the original as well as an improved edition as long as you credit the original author, myself.

FSUIPC SDK Usage

I would like to give full credit to Scott McCrory, Bob Scott who wrote the C# class, and Pete Dowson who developed FSUIPC. Flight Data Viewer makes use of the System Developers Kit for FSUIPC. Thank you very much!

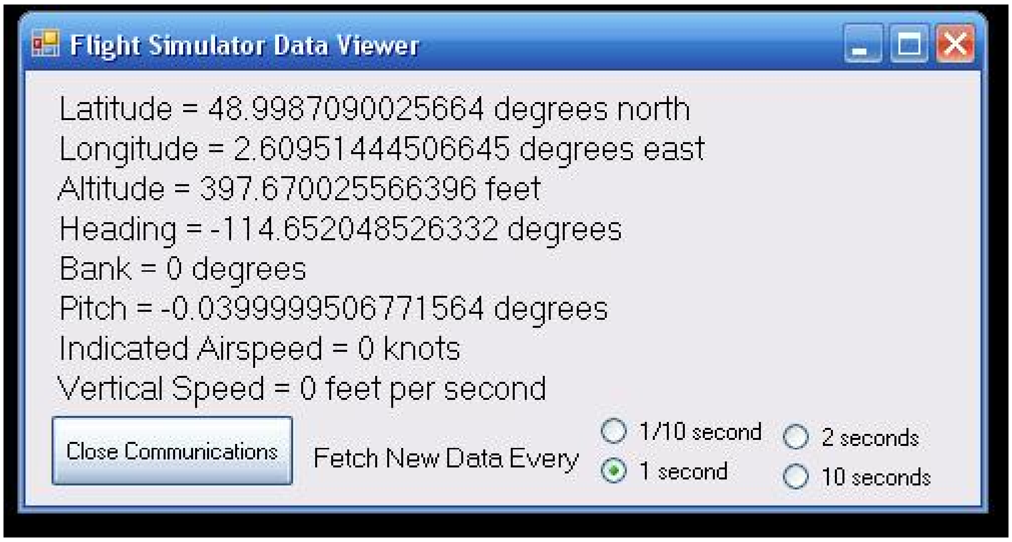
Flight Simulator Version Compatibility

Flight Data Viewer is compatible with FS2002, FS2004, and FSX. I have only tested FS2004 and FSX but FS2002 and FS98 should work as well.

Source Code Notes

The Main form is the only windows form in the application. It relies on the FSData Class and the fsuipc Class. FSData contains subroutines that gathers data from the fsuipc class and returns it. There is a subroutine for each of the major fields in FSData along with the subroutine GetAircraftDataArray(). This opens the FSUIPC port once, retrieves all the data and returns it in a double array. This is more efficient than calling each subroutine individually.

Screenshot



Flight Simulator Data Viewer

Latitude = 48.9987090025664 degrees north
Longitude = 2.60951444506645 degrees east
Altitude = 397.670025566396 feet
Heading = -114.652048526332 degrees
Bank = 0 degrees
Pitch = -0.0399999506771564 degrees
Indicated Airspeed = 0 knots
Vertical Speed = 0 feet per second

Fetch New Data Every: 1/10 second [✓ 1 second] 2 seconds 10 seconds

Code Breakdown of Main Form

Line	Source Code	Breakdown Comments
1	using System;	
2	using System.Collections.Generic;	
3	using System.ComponentModel;	
4	using System.Data;	
5	using System.Drawing;	
6	using System.Linq;	
7	using System.Text;	
8	using System.Windows.Forms;	Standard using systems, nothing special.
9		
10	namespace FlightDataViewer	
11	{	
12	public partial class Main : Form	
13	{	
14	double[] CurrentData = new double[9];	This will hold the array of data that will be retrieved from FSUIPC. (see array fields *A1*)
15	FSData FSData = new FSData();	Creates a new instance of the Class FSData which communicates with the fsuipc class.
16	public Main()	
17	{	
18	InitializeComponent();	
19	}	
20		
21	private void cmdOpen_Click(object sender, EventArgs e)	Open communications button event handler.
22	{	

23	`if (cmdOpen.Text == "Open Communications")`	If statement that makes the button act a switch. If the label is "Open Communication, it will prefrom the open function and vise-versa.
24	`{`	
25	`timer1_Tick(sender, e);`	Calls the timer routine that updates the labels and gets new data.
26	`timer1.Enabled = true`	Enables the timer that regularly retrives and updates the data from FSData.
27	`cmdOpen.Text = "Close Communications";`	Sets the button label to "Close Communications" that switches its function.
28	`}`	
29	`Else`	
30	`{`	
31	`timer1.Enabled = false;`	Stops the update timer.
32	`cmdOpen.Text = "Open Communications";`	Sets the label to open.
33	`}`	
34	`}`	
35		
36	`private void timer1_Tick(object sender, EventArgs e)`	Update timer tick event handler
37	`{`	
38	`CurrentData = FSData.GetAircraftDataArray();`	Calls a function that retrieves an array from the FSData class that contains the current data collected from fsuipc and copies into the CurrentData array.
39		
40	`lblLat.Text = string.Format("Latitude = {0} degrees north", CurrentData[1].ToString());`	Sets the Longitude label from the data in CurrentData array.
41	`lblLong.Text = string.Format("Longitude = {0} degrees west", CurrentData[2].ToString());`	Sets the Latitude label from the data in CurrentData array.
42	`lblAlt.Text = string.Format("Altitude = {0} feet", CurrentData[3].ToString());`	Sets the Alitude label from the data in CurrentData array.
43	`lblhead.Text = string.Format("Heading = {0} degrees", CurrentData[4].ToString());`	Sets the Heading label from the data in CurrentData array.
44	`lblbank.Text = string.Format("Bank = {0} degrees", CurrentData[5].ToString());`	Sets the bank label from the data in CurrentData array.
45	`lblpitch.Text = string.Format("Pitch = {0} degrees", CurrentData[6].ToString());`	Sets the pitch label from the data in CurrentData array.
46	`lblspeed.Text = string.Format("Indicated Airspeed = {0} knots", CurrentData[7].ToString());`	Sets the speed label from the data in CurrentData array.
47	`lblvertspeed.Text = string.Format("Vertical Speed = {0} feet per second", CurrentData[8].ToString());`	Sets the vertical speed label from the data in CurrentData array.
48		
49	`if (CurrentData[1] < 0)`	This If statement determines whether the latitude is positive or negative and prints the value into latitude label with either south or north.
50	`{`	
51	`lblLat.Text = string.Format("Latitude = {0} degrees south", (CurrentData[1]*-1).ToString());`	
52	`}`	
53	`else`	

54	`{`	
55	`lblLat.Text = string.Format("Latitude = {0} degrees north", (CurrentData[1]).ToString());`	
56	`}`	
57		
58	`if (CurrentData[2] < 0)`	This If statement determines whether the latitude is positive or negative and prints the value into longitude label with either east or west.
59	`{`	
60	`lblLong.Text = string.Format("Longitude = {0} degrees west", (CurrentData[2]*-1).ToString());`	
61	`}`	
62	`else`	
63	`{`	
64	`lblLong.Text = string.Format("Longitude = {0} degrees east", (CurrentData[2]).ToString());`	
65	`}`	
66	`}`	
67		
68	`private void Form1_Load(object sender, EventArgs e)`	
69	`{`	
70		Form_load event handler, empty
71	`}`	
72		
73	`private void optHunderth_CheckedChanged(object sender, EventArgs e)`	
74	`{`	
75	`timer1.Interval = 100;`	
76	`}`	
77		
78	`private void optOne_CheckedChanged(object sender, EventArgs e)`	
79	`{`	
80	`timer1.Interval = 1000;`	
81	`}`	
82		
83	`private void optTen_CheckedChanged(object sender, EventArgs e)`	
84	`{`	
85	`timer1.Interval = 10000;`	
86	`}`	
87		
88	`private void optThousanth_CheckedChanged(object sender, EventArgs e)`	These event handlers are for the radio buttons on the form and each change the update timer's interval property accordingly.
89	`{`	
90	`timer1.Interval = 2000;`	
91	`}`	
92	`}`	
93	`}`	
94		

Appendix

A1: CurrentData[] Fields:

0. ID - Not used in this version
1. Latitude - Latitude of aircraft in degrees
2. Longitude - Longitude of aircraft in degrees
3. Altitude - Altitude of aircraft in feet
4. Heading - in degrees (magnetic north)
5. Bank - degrees positive right, negative left
6. Pitch - degrees positive nose down, negative nose up
7. Airspeed - indicated Airspeed in knots
8. Vertical Speed - in feet per second (positive up, negative down)