

**Version**

**1.1**

JEFFERSON LAB

---

Data Acquisition Group

# JEventViewer User's Guide

JEFFERSON LAB DATA ACQUISITION GROUP

# **JEventViewer User's Guide**

---

Carl Timmer  
[timmer@jlab.org](mailto:timmer@jlab.org)

3-Feb-2015

© Thomas Jefferson National Accelerator Facility  
12000 Jefferson Ave  
Newport News, VA 23606  
Phone 757.269.7365 • Fax 757.269.6248

---

# Table of Contents

- 1. Evio Event Viewing..... i**
- 1.1 Features..... ii
  
- 2. File Data Viewing.....4**
- 2.1 Searching.....5
- 2.1.1 By Value .....5
- 2.1.2 By Location.....5
- 2.1.3 By Page .....5
- 2.1.4 By Evio Block Header .....6
- 2.1.5 By Evio Event .....6
- 2.1.6 By Evio Faults.....6

# Section 1

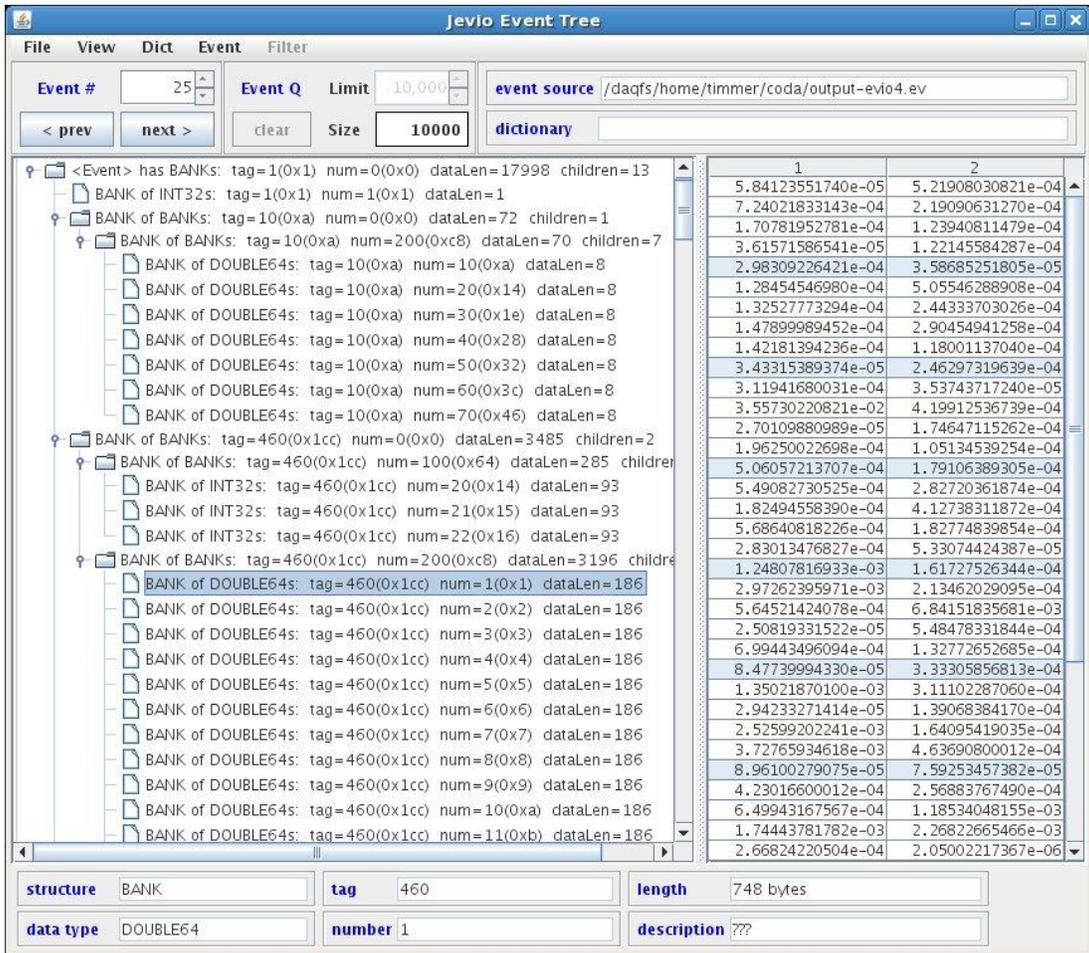
## 1. Evio Event Viewing

This manual describes a graphical user interface for looking at EVIO format files event-by-event, although it can also look at any file as a list of 32 bit integer (words). To run it simply execute:

```
java org.jlab.coda.eventViewer.EventTreeFrame
```

Make sure that the EventTreeFrame class or the jar file JEventViewer-1.x.jar in is your CLASSPATH environment variable. The following is a screen shot of the gui.

Figure 1.1: Event-viewing gui



## EVIO EVENT VIEWING

### 1.1 Features

Here's a quick list of the main features:

- Valid event sources are files, cMsg messages, and ET buffers
- Fast compare ability for data from different events
- When receiving events through cMsg or ET, they can be filtered based on their CODA event type (physics, control, etc.) and trigger type if physics event
- View integer data as hex or decimal
- Select dictionary from event source or from separate file containing dictionary
- View the dictionary being used
- Export any evio file in xml format
- Add or remove data columns
- View the contents of any file as 32 bit hex integers and search for values, evio structures, or evio errors

Starting with the middle of the gui first, the left side shows a tree structure diagram of the whole, single evio event being viewed. Notice that the type of each evio structure is given (bank, segment, tagsegment), along with the type of data it contains, tag, num, size, and # of children. Tag and num are shown in decimal and hex. If a dictionary is being used, the dictionary name is displayed instead of the corresponding structure type, data type, tag, and num values.

The right side, on the other hand, shows the data of any selected bank, segment, or tagsegment that contains a primitive data type. The number of columns can be set in the "View" menu. Integers can be displayed in hex or decimal.

A fast compare feature is able to compare data from different events. If the current event is changed while viewing the data of its selected structure, and if the new event has a structure with the same hierarchy of tags that the previous selection had, it too is automatically selected. This facilitates comparing the same structure in each successive event by simply hitting the "next" event button.

A dictionary can be loaded from a separate xml format file, or it can come embedded in an evio format file or buffer (cMsg, ET). The viewer allows the user to switch, in the "Dict" menu, between the different dictionaries if more than one is available. Any dictionary being used can be displayed instead of the data.

Selecting an ET system or a cMsg server as an event source, in the "Event" menu, brings up other menus to allow the proper connections to be created and maintained. The only assumptions made are that in a cMsg message, the evio data is contained in the byteArray field. Any dictionary is first looked for in the evio data and if none is found, it is looked for in a String payload item called "dictionary".

The box in the upper left (under the row of menu buttons), "Event #", shows the event currently selected (in this case 25) and allows the user to navigate to the desired event.

## EVIO EVENT VIEWING

The box to its right, “Event Q”, shows different things depending on if the data source is a file, cMsg message, or ET event. For files it shows the total number of events (in this case 10,000). For cMsg messages and ET events, on the other hand, events are continually arriving. In this case, “Size” shows the number of events currently in an internal queue. “Limit” allows the user to set the size of this internal queue, while “Clear” will remove all events currently in the queue. Once this queue is full, nothing else is added. The “Event #” controls can be used to switch between events in the queue.

Switching between the different event sources can be done in the “Event” menu item. When selecting a cMsg or ET source, the “Filter” menu is enabled. With this menu, the user can choose to look at control, partially-built physics, physics events, or any combination as well as the selecting the run type of interest.

Notice that above the data, there are boxes containing the event and dictionary sources. Beneath the data are boxes containing information about the selected data structure.

**Section**  
**2**

## 2. File Data Viewing

The following figure is a screen shot of a particular file's data obtained by selecting the "View File Bytes" option of the "File" menu of the initial screen shown previously.

Figure 2.1: Data-viewing gui

Word Position	+1	+2	+3	+4	+5	Comments
0	0x0000000d	0x00000000	0x00000008	0x00000001	0x00000000	
5	0x00000004	0x00000000	0xc0da0100	0x00000004	0xffd10100	Block Header
10	0x54816af1	0x000006b4	0x00000001	0x0000e3bd	0x00000001	
15	0x00000008	0x00000003	0x00000000	0x00000004	0x00000000	
20	0xc0da0100	0x00000004	0xffd20100	0x54816afd	0x00000000	Block Header
25	0x00000000	0x000092eb	0xff701001	0x000000a3	0xff232033	
30	0x000a0006	0x00000000	0x00000001	0x00000000	0xb94ba825	
35	0x000006b4	0x00000001	0x00850001	0x00010000	0x36010002	
40	0xb94ba825	0x00000000	0x35010002	0xb94ba825	0x00000000	
45	0x38010002	0xb94ba825	0x00000000	0x37010002	0xb94ba825	
50	0x00000000	0x40010002	0xb94ba825	0x00000000	0x34010002	
55	0xb94ba825	0x00000000	0x3e010002	0xb94ba825	0x00000000	
60	0x33010002	0xb94ba825	0x00000000	0x3f010002	0xb94ba825	
65	0x00000000	0x3a010002	0xb94ba825	0x00000000	0x39010002	
70	0xb94ba825	0x00000000	0x3d010002	0xb94ba825	0x00000000	
75	0x3b010002	0xb94ba825	0x00000000	0x3c010002	0xb94ba825	
80	0x00000000	0x47010002	0xb94ba825	0x00000000	0x4e010002	
85	0xb94ba825	0x00000000	0x4d010002	0xb94ba825	0x00000000	
90	0x52010002	0xb94ba825	0x00000000	0x01010002	0xb94ba826	
95	0x00000000	0x49010002	0xb94ba825	0x00000000	0x1c010002	
100	0xb94ba825	0x00000000	0x5e010002	0xb94ba825	0x00000000	
105	0x4b010002	0xb94ba825	0x00000000	0x5f010002	0xb94ba825	
110	0x00000000	0x19010002	0xb94ba825	0x00000000	0x1a010002	
115	0xb94ba825	0x00000000	0x1b010002	0xb94ba825	0x00000000	
120	0x0f010002	0xb94ba825	0x00000000	0x0e010002	0xb94ba825	
125	0x00000000	0x0d010002	0xb94ba825	0x00000000	0x0c010002	
130	0xb94ba825	0x00000000	0x0b010002	0xb94ba825	0x00000000	
135	0x14010002	0xb94ba825	0x00000000	0x15010002	0xb94ba825	
140	0x00000000	0x16010002	0xb94ba825	0x00000000	0x13010002	
145	0xb94ba825	0x00000000	0x12010002	0xb94ba825	0x00000000	
150	0x11010002	0xb94ba825	0x00000000	0x10010002	0xb94ba825	
155	0x00000000	0x20010002	0xb94ba825	0x00000000	0x21010002	
160	0xb94ba825	0x00000000	0x22010002	0xb94ba825	0x00000000	
165	0x23010002	0xb94ba825	0x00000000	0x24010002	0xb94ba825	
170	0x00000000	0x29010002	0xb94ba825	0x00000000	0x25010002	
175	0xb94ba825	0x00000000	0x2a010002	0xb94ba825	0x00000000	
180	0x26010002	0xb94ba825	0x00000000	0x28010002	0xb94ba825	
185	0x00000000	0x27010002	0xb94ba825	0x00000000	0x1f010002	
190	0xb94ba825	0x00000000	0x00000077	0x00361001	0x00000075	
195	0x001a0101	0x80d00101	0x90c00001	0x98297503	0x00000017	
200	0xc4015007	0xbc8b5544	0xbc8b5a83	0x88c00008	0xf8c00000	
205	0xf8c00000	0x81100101	0x91000001	0x98297503	0x00000017	
210	0xc4015007	0x89000006	0xf9000000	0xf9000000	0x81500101	
215	0x91400001	0x98297503	0x00000017	0xc4015007	0xbc3a35902	
220	0xf94000ed	0x89400008	0x81900101	0x91800001	0x98297503	
225	0x00000017	0xc4015007	0x89800006	0xf9800000	0xf9800000	
230	0x81d00101	0x91c00001	0x98297503	0x00000017	0xc4015007	
235	0xbca7550d	0xbca7599d	0x89c00008	0x82100101	0x92000001	
240	0x98297503	0x00000017	0xc4015007	0xbca455bc	0xbca456ec	
245	0xbca45bd3	0xfa0000ed	0x8a00000a	0xfa000000	0xfa000000	
250	0x82500101	0x92400001	0x98297503	0x00000017	0xc4015007	
255	0xbc8b595b	0xbc8b5ba7	0xbc8b6584	0xbc8b6a0a	0x8a40000a	
260	0xfa000000	0xfa000000	0x82900101	0x92800001	0x98297503	
265	0x00000017	0xc4015007	0x8a800006	0xfa800000	0xfa800000	
270	0x83500101	0x93400001	0x98297503	0x00000017	0xc4015007	
275	0xbca7550f	0xbca45507	0xbca45a76	0xfb4000ed	0x8b40000a	

## FILE DATA VIEWING

There are occasions when one wants to examine the raw bytes in a file. This tool will allow one to do just that. Each cell of the table contains 32 bits worth of data displayed in hex. Data can be switched between big and little endian under the “File” menu. The table contains 200MB worth of data at one time. For larger files, the next or previous 200MB are loaded when required while scanning through it. On the very far right of the gui is a slider which indicates where the current view is in relation to the total length of the file.

### 2.1 Searching

In order to facilitate finding the data of interest, there are a number of different ways to hunt through it. The control panel on the left has “Search By” radio buttons allowing one to select whether to search by:

1. Looking for a given value
2. Jumping to a given position in the file
3. Scrolling page by page
4. Jumping from one evio block header to the next
5. Jumping from one evio event to the next
6. Scanning the whole file for evio faults or errors

#### 2.1.1 By Value

Look for a given value by selecting the “Word Value” radio button, typing the value into the “Search For” widget, and then hit the forward or backward search button under “Search Controls”. The “Stop” button will be activated since searching a large file (say 20GB) may take serious time. If a search is stopped, the view position stays where it was when the search was started. If stopped, starting another search starts from the same location. A progress bar is there to estimate how much of the file has been searched.

When a value is found, it is highlighted in red. Hit the search button again to find the next or previous value. Highlights can be cleared under the “File” menu.

#### 2.1.2 By Location

Look at a given location in the file by selecting the “Word Position” button, typing the position into the “Search For” widget, and then hitting the “Go” button. The view jumps to the given location and the value is selected (but not highlighted). The first position starts at 1, not 0. You can read the position from the table by taking the number in the far left column and adding the number of the heading at the very top of the column.

#### 2.1.3 By Page

The “Page Scrolling” button activates the forward and backward buttons which hop through the file view by view.

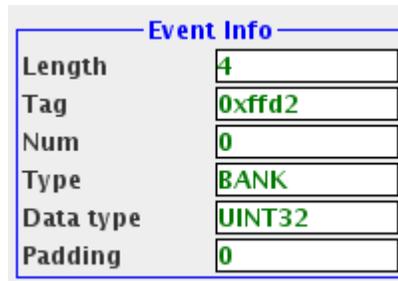
### 2.1.4 By Evio Block Header

Look for an evio format block header by selecting the “Evio Block” button. The program first looks for the last word – the magic # 0xc0da0100 - of an evio block header. If found, it checks that the previous word is 0 and the one before that has the least significant 8 bits equal to 4 (the evio version). If so, it highlights all 8 words in orange. All the information contained in that header is also displayed on the left in a panel called “Block Info” which can be seen in the figure above.

This type of searching is useful when dealing with corrupted data files as it can home in on an area of the file whose format can be checked and can give information about what data should come following. Hint, if no block header is found in a scan, switch the data endianness and try another one.

### 2.1.5 By Evio Event

Look for an evio event by selecting the “Evio Event” button. This is less straightforward than looking for block headers since there is no universal signature to look for. There are two ways to do the search. The first way is start the search immediately upon loading the file’s data or to first select the “0” position (far left column before any data). Then hit the forward button. It is smart enough to hop over any block header encountered and uses the length found in the event’s header to be able to find the next when the forward button is clicked again. The first word of each event found in this way is highlighted in blue and the header information is displayed on the left in a panel called “Event Info” (see figure below).



Event Info	
Length	4
Tag	0xffd2
Num	0
Type	BANK
Data type	UINT32
Padding	0

2.2 Event information panel

The second way to search is to select the known first word of an event with the mouse. Hit the forward button to find subsequent events. Hint, the word immediately after a block header is the first word of an event.

### 2.1.6 By Evio Faults

Look for faults or errors in the evio format by selecting the “Evio Fault” button. This program scans the file from beginning to end and lists any errors in a panel to the left called “Evio Errors” (which can be seen in figure 2.1). The algorithm used to find these errors assumes that if a block header length does not equal the sum of the lengths of all the events it contains, then the block header is correct and the events are in error. It tries to continue scanning for the next error and stops if it encounters an unrecoverable error or makes it to the end of the file.

## **FILE DATA VIEWING**

The errors come in two varieties: block and event. All blocks in which there are problems are listed as are any events in which errors are found. These errors are listed as buttons. Click one and it hops to the location of the error with either the first block or event word selected. Its corresponding error message is also displayed at the top of the gui (not pictured).