# JEventViewer 2.0 User's Guide

Carl Timmer

Jefferson Lab Experimental Physics Software and Computing Infrastructure group

26-May-2021

© Thomas Jefferson National Accelerator Facility 12000 Jefferson Ave Newport News, VA 23606 Phone 757.269.7100

# Table of Contents

I. EVIO EV	rent Viewing	ii
1.1	Installation	<i>iii</i>
1.2	Prerequisites	iv
1.3	Documentation	iv
1.4	Features	v
2 File Da	to Viewing	7
2. The Da	ia vicwilig	/
2. File Da 2.1	Searching	8
2.1 2.1.1	Searching	8 8
2.1 2.1.1 2.1.2	Searching	8 8 9
2.1 2.1.1 2.1.2 2.1.3	Searching By Value By Location By Page	8 8 9 9
2.1 2.1.1 2.1.2 2.1.3 2.1.4	Searching By Value By Location By Page By Evio Record/Block Header	8 8 9 9 9
2.1 2.1.1 2.1.2 2.1.3 2.1.4 2.1.5	Searching By Value By Location By Page By Evio Record/Block Header By Evio Event	8 8 9 9 9 9 9

### **1. Evio Event Viewing**

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

Event #													
< prev	z S Event	Q Limit	з 🗘	event source	/Users/	timmer/	coda/evioData	Files/co	mpactEvi	oBuild.ev.lz4			
	next > clear	Size	3	dictionary	from evi	o file							
/IO event tree	has BANKs: tag=1(0x1)	num=1(0x1)	dataLen=5153	children=3	Po	sition Ø	+1 0x26cec005	0x7aa	+2 a9bd1e	+3 Øx7e7b9f63	+4 0x55851d84	+5 0x069b36cb	Comments
🔻 🚞 BANK	of BANKs: tag=2(0x2) r	num=2(0x2) d	lataLen=1733	children=8		5	0x2c09ec59	0x77:	1ce26d	0x0f22e99e	0x1961701e	0x110cdaf7	
BA	NK of INT32s: tag=3(0x	3) num=3(0x	3) dataLen=20	3		10	0x03Tac/4T 0x7dbo2c2d	0x4a	67d9bo	0x65a59113 0x52f11eab	0x5DabDC8C	0x21TCE01C 0x51dc1dc0	
BA	NK of CHAR8s: tag=4(0)	x4) num=4(0)	(4) dataLen=5	1 pad=1		20	0x70De3C20	0x220	dfe8ea	0x33T11edD 0x04f1f7cb	0x30C130ed	0x510C10C9	
SE	G5 dataLen=102 pad=2					25	0x39765cdb	0x26	191cd5	0x022e1891	0x2a2494cb	0x2b7c1614	
BA	NK of LONG64s: tag=41	(0x29) num=	41(0x29) data	len=406		30	0x1ef6c21a	0x61	bea8e5	0x71f2f615	0x6cb20a27	0x79557baf	
	IID DC(6) vnos(6) datal	en=203				35	0x4f35009d	0x78	e53f7a	0x018788a9	0x565f77a8	Øx58837c4e	
	ID BCAL datalan-406	205				40	0x601d4c0c	0x4c4	4f3b2e	Øx28737e19	0x710a528c	0x184c10e1	
	ID data an 220					45	0x13f4254a	0x7b	79f116	0x0f8c93eb	0x69469d0b	0x092ea491	
	uid dataLen=329					50	0x0bb9ccb3	0x6aa	a52aff	0x681b199f	0x0a397097	0x6b2da703	
Ha	IID dataLen=17					55	0x13bbdc5c	0x1ba	a5c9b1	0x65069af0	0x7558b01c	0x307a4695	
T BANK	of SEGMENTs: tag=15(0	xf) num=15(0	)xf) dataLen=1	707 childrei		60	0x01a60a55	0x57	148121	0x519bb210	0x12c0e240	0x16535291	
SE	GMENT of INT32s: tag=	9(0x9) dataLe	n=203			65	0x110312ea	0x444	0a1015	0xbf1c01ef	0x/8/233a3	0x6D8845D3	
SE	GMENT of CHAR8s: tag=	10(0xa) data	Len=51 pad=	1		75	0x71da59ad	0x/10 0x68/	d68629	0x35655675	0x2f0b4a00	0x13017700	
SE <sup>4</sup>	GMENT of SHORT16s: ta	g=11(0xb) da	ataLen=102 pa	ad=2		80	0x15239bd2	0x63	257a31	0x19fc0d75	0x2962ea6f	0x3e9b9286	
SE	GMENT of LONG64s: tag	=41(0x29) d	ataLen=406			85	0x7b60acfe	0x220	0f2835	0x60da42e3	0x6b7f7d57	0x2c8f72be	
SE	GMENT of FLOAT32s: tag	g=12(0xc) da	taLen=203			90	0x05009a5d	0x526	60f0d3	0x499b895a	0x1de70705	0x6911fe01	
SE	GMENT of DOUBLE64s: t	ag=13(0xd) d	lataLen=406			95	0x5c023cd5	Øx630	dbfda3	0x29e5ff67	0x21df8735	0x3c1d497c	
C SE	GMENT of CHARSTAR8s	tag=14(0xe)	datalen=329	0		100	0x46623b47	0×10	094b77	0x03b5040c	0x1a5a135c	0x5119bd58	
	of TACSECMENTs: tag=	16(0×10) pup	$= 16(0 \times 10)$ d:	tal en = 1707		105	0x250134cb	0×056	62dc9c	0x2f1e7108	0x78cd18ea	0x41665ef6	
	CECMENT of INT224	10(0×10) Iuli	datal an - 202	atalen=1707		110	0x709312f7	0x498	831316	0x36bafd13	Øx2eeec78b	0x609e4332	
	COSEGMENT OF INT 325: 1	ag=17(0x11)	dataLen=203			115	0x73c88418	Øx66a	ab32e6	0x5422c61f	0x4fe0a361	0x764c28a3	
	GSEGMENT OF CHAR8S:	tag = 18(0x12)	dataLen=51			120	0x53839d77	0x5es	920c47	0x73810398	0x78fd3f4c	0x325d39d9	
_ TA	GSEGMENT of SHORT16	: tag=19(0x1	<ol> <li>dataLen=1</li> </ol>	02		125	0x2088a595	0x26	0DCGED	0x035100a5	0x4c191122	0x21431/9e	
TA	GSEGMENT of LONG64s:	tag=41(0x29	) dataLen=40	6		130	0x110c000e	0x10	a2a2b4	0x15/5/604 0x7f731e31	0x1ad7C110	0x3e5C4035	
TA	GSEGMENT of FLOAT32s	: tag=20(0x1	<ol> <li>dataLen=20</li> </ol>	)3		140	0x7b57f126	0x32	f57fe2	0x2d9373de	0x1d498a82	0x4068e8ce	
T/	GSEGMENT of DOUBLE64	s: tag=21(0x	15) dataLen=4	106		145	0x2f43b49f	0x41	5695e9	0x4d291ffc	0x3bda22c0	0x32ece3fa	
T/	GSEGMENT of CHARSTAR	R8s: tag=22(0	x16) dataLen=	=329		150	0x748c4087	0x08	18b59e	0x42765210	0x5f2f8f89	0x092213ce	
						155	0x730a59fe	0x75	e343d2	0x14ae92de	0x42db49d0	0x5c1ef5ae	
						160	0x6236df1a	0x4ea	a6238f	0x1425b6a2	0x4d1b4b43	0x0a0c24fb	
						165	0x3ca0d12e	Øx596	6ec858	0x7c5cf751	0x2118308b	0x5aac8550	
						170	0x62f88706	0x5c4	4f706c	0x353f36dc	0x480f8389	0x0d2e94c0	
						175	0x09610728	0×062	298703	Øx1c7d696d	Øx277b2d9a	0x5677e619	
						180	0x4b588a15	0x/b	cc2e90	0x699c27a7	0x3/e/11d3	0x28bbc324	
						100	0x20292485	0X37	7853f1	0X4Ce308ea	exilcoct45	exozcatez ex1f42bf4e	
						195	0x72fe557e	0x3a	26570d	0x2cd1a120	0x0b46a375	0x190cea23	
						200	0x74ae8edb	0x2b3	3e9229	0x54591cb2	0.00400373	0719000023	
rsion	6	structure	TAGSEGM	ENT	tag	17			length	812 bytes			
51011	•	Junacture							iciigui	SIZ Dytes			

#### Figure 1.1: Event-viewing gui

This version of the JEventViewer is compatible with all evio formats, including the latest, evio 6.0. To run it, using Java 8 or later, simply execute:

java org.jlab.coda.eventViewer.EventTreeFrame

Make sure that the jar files, JEventViewer-2.0.jar and all the other jars in java/jars directory, are your CLASSPATH environment variable. The alternative to that is executing the provided script:

scripts/jeviodump

Note that the script is for CODA users and sets the classpath to:

```
$CODA/common/jar
```

In other words, make sure that your environmental variable CODA is defined and all the jar files in java/jars are in that directory as well.

#### 1.1 Installation

The code can be downloaded from its github site:

git clone <a href="https://github.com/JeffersonLab/JEventViewer.git">https://github.com/JeffersonLab/JEventViewer.git</a>

The default branch is "2.0" but one can insure that by calling:

git checkout 2.0

There's the jar file **JEventViewer-2.0.jar** in the **java/jars/java8** directory, already pre-built with Java 8, so one does not need to build it. There's another one in **java/jars/java15** directory built with Java 15.

However, to build it simply do:

ant jar

Other options can be seen by calling:

ant help

The output of this command is:

help:		
[echo]	Usage: ant [ant	options] <target1> [target2   target3  ]</target1>
[echo]	targets:	
[echo]	help	- print out usage
[echo]	env	- print out build file variables' values
[echo]	compile	- compile java files
[echo]	clean	- remove class files
[echo]	cleanall	- remove all generated files
[echo]	jar	- compile and create jar file
[echo]	install	- create jar file and install into 'prefix'
[echo]		if given on command line by -Dprefix=dir',
[echo]		else install into CODA if defined
[echo]	uninstall	- remove jar file previously installed into 'prefix'
[echo]		if given on command line by -Dprefix=dir',
[echo]		else installed into CODA if defined
[echo]	all	- clean, compile and create jar file
[echo]	javadoc	<ul> <li>create javadoc documentation</li> </ul>
[echo]	developdo	c - create javadoc documentation for developer

[echo]	undoc	-	remove	all	javado	c documentation
[echo]	prepare	-	create	nece	essary	directories

Although this is fairly self-explanatory, executing ant is the same as ant compile. That will compile all the java. All compiled code is placed in the generated ./build directory. If the user wants a jar file, execute ant jar to place the resulting file in the ./build/lib directory. The java command in the user's path will be the one used to do the compilation.

#### 1.2 Prerequisites

The other jar files necessary to compile JEventViewer-2.0.jar are in the java/jars directory. They are compiled with Java 8:

- cMsg-5.2.jar
- disruptor-3.4.3.jar
- et-16.4.jar
- jevio-6.0.jar
- lz4-java.1.8.0.jar

In addition, there are 2 subdirectories:

- java/jars/java8, which contains all such jars compiled with Java 8, and
- java/jars/java15 which contains all jars compiled with Java 15.

If a jar file is not available in Java 15 use the Java 8 version. To generate these jar files, go to their respective github sites and follow the directions there:

https://github.com/JeffersonLab/cMsg
https://github.com/JeffersonLab/disruptor
https://github.com/JeffersonLab/et
https://github.com/JeffersonLab/evio
https://github.com/lz4/lz4-java

#### 1.3 Documentation

Basically, you are now reading the only user documentation in either a pdf or word doc. In the repository, it's located in the **doc/users\_guide** directory. There is javadoc that can be

generated (ant javadoc or ant developdoc) but would only be useful to a developer or one trying to modify the source code.

#### 1.4 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
- View the contents of any file as 32 bit hex integers
- Search for values, positions, evio records/blocks, evio events, or evio errors

In the figure above, 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 data type and not another container type. 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 2) and allows the user to navigate to the desired event.

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 3). 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 such as its structure type, data type, tag, num, length in bytes, description, evio version, and the type of data compression if any.

**Warning about performance**: for large files, make sure they are local to the machine that's running this program since it uses memory mapping to look at file data. You do not want the performance hit you'll take for viewing files which are served over the network!

## 2. File Data Viewing

File View Dict Open Event File ^C Open Dictionary View File Bytes Export File to XML Quit	Event Filter 0 Event Q Lim clear Size	it 100 C	The obtai "File" <b>Figur</b>	followin ned by s menu c e 2.1: Da	g figure selecting of the in <b>ata-view</b> ev bytes	e is a s g the "V itial scre <b>/ing gui</b>	creen siew File een sho	shot of a e Bytes" o own previe	file's dat ption of th ously.
File									
/Users/timmer/coda/evioDataFiles/compactEvioBuild.ev									
	2 Ar some	Word Position	+1	+2	+3	+4	+5	Comments	Color Key-
File	e Info	0	0x4556494f	0x00000001	0x0000000e	0x00000002	0x00000000	File Header	
File ID	0x4556494f	10	0x00000000	0x00000f4c4	0×00000000	0×000000000	0x0000009b	File neader	File
File type	EVIO_FILE	15	0×00000001	0x0000000e	0×00000001	0×00000004	0x00800006		Header
Header words	1	20	0x00000000	0x00000000	0x0000022e	0x000000000	0x00000000 0x3c786d6c		User header
Record count	2	30	Øx44696374	0x3e0a2020	Øx3c62616e	Øx6b206e61	Øx6d653d22		
Index array bytes	0	35	0x48616c6c	0x44222020 0x362d3822	0x20202020 0x20207470	0x20202020 0x70653d22	0x20202074		Record Normal-
Evio version	6	40	0x22203e0a	0x2020202020	0x20203c64	0x65736372	0x69707469		Header
Has dictionary	true	50	0x6f6e2066	0x6f726d61	0x743d224e	0x65772046	Øx6f726d61		Index array
Has first event	false	55	0x7422203e 0x3c2f6465	0x68616c6c	0x5f645f74 0x7074696f	0x61675172	0x616e6765 0x20202020		User header
Has trailer & index	true	65	0x203c6261	0x6e6b206e	0x616d653d	0x22444328	0x25742922		
User header bytes	620	70	0x20202020	0x20202020	Øx7461673d	0x22362220	0x6e756d3d		Record with error
User register	0x0	75	0x22342220 0x66206e61	0x3e0a2020 0x6d653d22	0x20202020 0x78706f73	0x20202020 0x28256e29	0x3c6c6561 0x22202074		Event with error
Trailer position	62660	85	Øx61673d22	0x3622206e	Øx756d3d22	0x3522202f	0x3e0a2020		Evio struct error
User int 1	0x0	90	0x20202020	0x20202020	Øx3c62616e	0x6b206e61	Øx6d653d22		
User Int 2	UXU	95	0x79706T73 0x756d3d22	0x28256e29 0x3622202f	0x22202074 0x3e0a2020	0x616/3022 0x20202020	0x3622206e		Event normal
Sear	rch By	105	0x6e6b203e	0x0a202020	0x2020203c	Øx62616e6b	0x206e616d		Word value
O Word Value		110	0x653d2242	0x43414c22	0x20202020	0x20207461	Øx673d2237		Current selection
O Word Position		115	0x6e616d65	0x20202020	0x256e2922	0x20203060	0x3d223722		
Page Scrolling		125	0x206e756d	0x3d22312d	0x3322202f	0x3e0a2020	0x20202020		
Evio Record		130	0x3c2f6261	0x6e6b203e	0x0a20203c	0x2f62616e	0x6b203e0a		
O Evio Event		135	0x45473522	0x20746167	0x3d223522	0x203e0a20	0x20202020		
O Evio Fault		145	0x20203c64	Øx65736372	0x69707469	0x6f6e2066	0x6f726d61		
0		150	0x743d224t	0x6c642046	0x61726d61	0x7422203e	0x74616720		
Sear	ch For	160	0x69707469	0x6f6e3e0a	0x20203c2f	0x64696374	0x456e7472		
0.01-0100		165	0x793e0a3c	0x2f786d6c	0x44696374	0x3e0a0000	0x00002856		
Uxcuda0100		170	0×00000001 0×00000000	0x0000000e 0xc0da0100	0x00000002 0x0000a118	0×00000008	0×00000006	Record Header	
		180	0×00000000	0×00000000	0×00000000	0x0000508c	0x0000508c		
Search	Controls	185	0x00001422	0x00011001	0x000006c6	0x00021002	0x000000cc		
<	>	190	0x1ba99155	0x66051442	0x2a112f81	0x3e29b7a5	0x6601833a		
Start Scan	Stop	200	0x16def006	0x5c6bd049	0x061dbfff	0x60dbe21a	0x302705ca		
Start Start		205	0x5da933b7	0x6db1a342	0x32c217ca	0x79f2a5e2	0x7d90e09d		
D	one	210	0x1f5010e7	0x06cc6861	0x38d21427	0x5c8015d4	0x51d209fd		
Reco	rd Info	220	0x7d96cd6e	Øx671e8a72	0x50252e0a	Øx1b1e4532	Øx53a74221		
Total words	10326	225	0x00852986	0x0e32a7cd	0x1ca38e5a	0x4a82b8cb	0x3bcf5a38		
Record #	1	235	0x11c9dbd6	0x2cee1759	0x7705f5c4	0x555759b7	0x4eccd9d0		
Header words	14	240	0x557685e5	0x7ce61446	0x066b76fd	0x67d6d90f	Øx010503b9		
Event count	2	245	0x181b2bb5	0x6bf72675	0x37f90764	Øx63eb252a Øx3de76aac	0x4e6d86e6		
Index array bytes	8	255	0x343d0aa6	0x6ea14630	0x575a15dc	0x00c9f153	0x05429d3a		
Version	6	260	0x0be0106e	0x3f62c149	0x208e5adc	Øx6931da35	Øx549e1398		
Has dictionary	false	265	0x15ef9fc7 0x73f7e7c7	0x0c31313c 0x1c580cb1	0x5ffb516b 0x0e225802	0x25d68603 0x7181a348	0x6444de2b 0x24a82bcf		
Is last	0	275	0x038e8fdb	0x211d33e9	0x3da50ff3	0x7e2ce4aa	0x73b7d8bc		
Uncompressed bytes	41240	280	0x614a7e0d	0x63133eff	0x20bb33ac	0x10845f28	0x007a54c1		
Compression type	None	285	0x1e9a4541 0x0f9a3b72	0x10/77428 0x34dc65f3	0x568114f4 0x6f7fc32b	0x36ae905a 0x34db0cb4	0x3df7318b 0x6f1558df		
Compressed words	0	295	0x6dab37f9	0x0e023989	0x186fd98d	0x15b8d5c2	0x7cadeadb		
User register 1	0x0	300	0x072af9c6	0x415d94ed	0x7dd7f570	Øx4495586c	Øx12b4c439		
User register 2	0x0	305	0x544135a5 0x0cc44c7a	0x09be353c	0x42c/d481 0x50911b02	0x32851333 0x4796e472	0x/01/ac8b 0x6f952c7f		
		315	0x4150a985	0x3342ca12	0x54fec22f	0x44b3009e	0x6eec0c1c		

There are occasions when one wants to examine the raw bytes in a file. This tool will allow one to do just that. It is capable of viewing any file's data, although it's designed specifically to look at evio versions 4 and 6 format data.

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 up to 1GB worth of data at one time. For larger files, the next or previous 1GB are loaded when required while scanning through it. On the immediate right of the data is a slider which indicates where the current view is in relation to the part of the file that is currently memory mapped (up to 1GB). On the far right is a color key to highlighted cells.

The figure above is showing an evio 6 format file. All such files have a file header shown in blue. The light blue is the main header of 14 words. Although there is no index in this case, there is a dictionary which is stored in the file header's so-called user header. This is seen highlighted in dark blue. In the "File Info" box on the top left, all values in the file header appear in a table.

When searching for record headers, each one shows up highlighted in green. The light green is the main header of 14 words. The mandatory index of events shows up in medium green. Although not seen above, since it isn't used in evio, any associated user header is shown in dark green. When a record header is found, it's data is shown in the "Record Info" box on the left.

When searching for events, the first 2 words of each are highlighted in cyan. When an event is found, it's data is shown in the "Event Info" box on the left (not seen in the figure above).

#### 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 or by blocks of 40 pages
- 4. Jumping from one evio record/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 extended 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 gold. 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 "<" and ">" buttons which hop through the file page (or view) by page. It also actives the "<<" and ">>" buttons immediately underneath which move through the file in 40 pages at a click.

#### 2.1.4 By Evio Record/Block Header

For evio version 4 files: look for an evio format block header by selecting the "Evio Block" button. The program first looks for the magic # (0xc0da0100) of an evio block header. If found, it checks that the header length is 8 words. If so, it highlights all 8 words in green. All the information contained in that header is also displayed on the left in a panel called "Block Info.

For evio version 6 files: look for an evio format block header by selecting the "Evio Record" button. The program first looks for the magic # (0xc0da0100) of an evio record header. If found, it checks that the header length is 14 words. If so, it highlights all 14 words in light green. It highlights the index part of the header in medium green, and the user header part in the darkest green. All the information contained in that header is also displayed on the left in a panel called "Record Info" which can be seen in the figure above.

#### 2.1.5 By Evio Event

Look for an evio event (top level evio bank) by selecting the "Evio Event" button. This is less straightforward than looking for record/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 a position before any events. Then hit the forward button. It is smart enough to hop over any file/record/block headers encountered and uses the length found in the event's header to be able to find the next one when the forward button is clicked again. The first two words (or header) of each event found in this way is highlighted in cyan and the header information is displayed on the left in a panel called "Event Info" (see figure below).

Even	it info ———
Length	4
Tag	0xffd1
Num	0
Туре	BANK
Data type	UINT32
Padding	0
Bank Type	Prestart event

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. Remember that the word immediately after a record/block header is the first word of an event. Hint: selecting the first word of any bank structure (top level or not) will display all of its information

A quick note on the bank type. In CODA online, some tags are reserved for specific purposes. If a selected event has such a reserved tag, its purpose will be shown as the "Bank Type".

#### 2.1.6 By Evio Faults

Look for faults or errors in the evio format by selecting the "Evio Fault" button. Simply hit the "Start Scan" button and this program scans the file from beginning to end (or as far as it can parse) and lists all blocks containing errors in a panel on the left called "Evio Errors" (which can be seen in figure 2.3 below).

The algorithm used to find these errors tries to parse as much of the file as possible. For example, if a block header length does not equal the sum of the lengths of all the events it contains, then the block header length is assumed for the moment to be correct and the event lengths in error. It tries to continue by scanning the next block and stops if it encounters an unrecoverable error or makes it to the end of the file.

Errors that are caught include bad/inconsistent values in a block/event header, wrong endianness of the displayed data, length of block header not consistent with length of contained events, and not enough data to read block/event (usually a bad length), and too large of an event count in a header. The search can go into events themselves to find lower level evio errors.

For an evio version 6 file, it will find inconsistencies between compression type and header values of compression word length and uncompressed data length. Any conflict between the index length and the number of events in a record will be flagged. Of course, if a file contains compressed data, evio events will not be scanned.

To print out suspicious record numbers or record header sizes, one must set the debug flag by hand in the scanFileForErrors() method of the EvioScanner(V6).java file.

Each block in which there is a problem is listed as a button. Click one and it hops to the beginning of that block which will be highlighted in red. Within that block, the ">" and "<" buttons move from event to event. If an event has an error, it is the last event to be accessible through the search buttons and will be highlighted in purple. If the event containing the error has an internal bank or structure with an error, it can also be

accessed through the search buttons and will be highlighted in orange. A corresponding error message (or messages) is displayed at the top of the gui in red text.

Below, a small file with evio format errors has been scanned. It reveals errors in 2 records. The first record is selected showing, in red, a header with an uncompressed data length of 0 even though there is no compression. It also shows the header saying it contains 3 events but there are entries in the index for only 2. Finally, it found an error in the first event, signified by its header in purple. The error is in a sub-structure, highlighted in orange. In this case a little investigation shows that the second bank header word shows padding of 2 for a data type of 32 bit unsigned int, when it should be 0.

File			1011001001001001010	V 09103					
		/llcore /timme	r/coda/aviaDataEi	ilac (HandCreated)	16 m				
/ Users/timmer/cood/evioUstariles/manucreateuvo.ev									
Record. No con	npression, but com	pien i= 0 or und	complien wrong,	index bytes (6) !=	4 event-count (12)	, Event #0 na	senor		
	Word Position	+1 0×4556494f	+2	+3	+4	+5	Comments	Color Key	
File Info	5	0×10000406	0×00000000	0xc0da0100	0x08070605	0x04030201	File Header	Til.	
File ID 0x45564941	10	0×00000000	0x0000015c	0×00000000	0×00000000	0x0000007c		Header	
File type EVIO_FILE	15	0×00000002	0x000008c	0×00000003	0×00000111	0x00000222		Index error	
File split # 1	20	0x00000333	0x0000001f	0×00000001	0×0000000e	0×00000003		Index array	
Header words 56	25	0x00000008	0x000000000	0x00000000C	0x00000100	0x000000000		User header	
Record count 3	35	0x0000001c	0×00000014	0×00000001	0×00000002	0x00000003		- Record Normal-	
Index array bytes 16	40	0×00000006	0x00018e01	0×00000004	0×00028102	0×00000001		Header	
Evio version 6	45	0×00000001	0×00000001	0×00000004	0×00030103	0×00000002		Index array	
Has dictionary false	50	0x00000002	0x00000002	0x00000023	0x00000002	0x00000000e		lindex array	
Has first event false	60	0x00000000	0×000000000	0×000000001	0x000000002	0x00000003		User neader	
Has trailer & index true	65	0×00000004	0×00000014	0×00000014	0×00000014	0×00000111			
User header bytes 12	70	0×00000222	0x00000333	0×00000004	0×00018501	0x00010002		Deserd with error	
User register 0x807060504030201	75	0x00030004	0x00056666	0×00000004	0x0001c701	0x01020304		Record with error	
Trailer position 348	80	0X05060708 0x00000005	0x09333333 0x00000005	0×00000004	0×00010101	0x000000005		Event with error	
User int 1 0x0	90	0×000000000	0×000000010	0x30000206	0×00000000	0xc0da0100		Evio struct error	
User int 2 0x0	95	0×00000000	0×00000000	0×00000011	0×00000022	0×00000033			
Search Ry	100	0×00000044	0×0000007c	0×00000002	0×0000008c	0×00000003		Event normal	
Owentstelen								Word value	
Word Value								Current selection	
Word Position									
Page Scrolling									
Evio Record									
O Duin Dunnt									
🖸 Evio Fault									
Court For									
Search For									
0xc0da0100									
Evio Errors									
O Block 1									
O Block 2									
Search Controls									
Search Controls									
Start Scan Stop									
David									
Done									
Record Info									
Total words 31									
Pacord # 1									
Header words 14									
Event count 3									
ladev array bytes									
Marsian C									
Version 6									
Has dictionary false									
Is last false									
User header bytes 12									
Uncompressed bytes 0									
Compression type None									
Compressed words 0									
User register 1 0x0									
User register 2 0x0									

#### Figure 2.3: Error Scanning