artdaq_demo

3.13.00

Generated by Doxygen 1.8.5

Thu Sep 5 2024 11:44:44

Contents

1 Namespace Index **1**

1.1 Namespace List 1

2 Hierarchical Index **3**

2.1 Class Hierarchy 3

3 Class Index **5**

3.1 Class List 5

4 Namespace Documentation **7**

4.1 anonymous_namespace{AsciiSimulator_generator.cc} Namespace Reference 7

4.1.1 Function Documentation 7

4.1.1.1 convertToASCII 7

4.2 artdaq Namespace Reference 7

4.2.1 Detailed Description 8

4.3 demo Namespace Reference 8

4.3.1 Detailed Description 9

5 Class Documentation **11**

5.1 demo::ASCIIDump Class Reference 11

5.1.1 Detailed Description 11

5.1.2 Constructor & Destructor Documentation 11

5.1.2.1 ASCIIDump 11

5.1.3 Member Function Documentation 12

5.1.3.1 analyze 12

5.2 demo::AsciiSimulator Class Reference 12

5.2.1 Detailed Description 12

5.2.2 Constructor & Destructor Documentation 13

5.2.2.1 AsciiSimulator 13

5.3 demo::CheckIntegrity Class Reference 14

5.3.1 Detailed Description 14
5.3.2 Constructor & Destructor Documentation 14
5.3.2.1 CheckIntegrity 14
5.3.3 Member Function Documentation 15
5.3.3.1 analyze 15
5.4 demo::CommandPacket Struct Reference 15
5.4.1 Detailed Description 15
5.5 demo::DemoViewer Class Reference 16
5.5.1 Detailed Description 16
5.5.2 Constructor & Destructor Documentation 16
5.5.2.1 DemoViewer 16
5.6 art::EventReporterOutput Class Reference 17
5.6.1 Detailed Description 17
5.6.2 Constructor & Destructor Documentation 17
5.6.2.1 EventReporterOutput 17
5.7 demo::GetPackageBuildInfo Struct Reference 17
5.7.1 Detailed Description 18
5.7.2 Member Function Documentation 18
5.7.2.1 getPackageBuildInfo 18
5.8 demo::MisbehaviorTest Class Reference 18
5.8.1 Detailed Description 19
5.8.2 Constructor & Destructor Documentation 19
5.8.2.1 MisbehaviorTest 19
5.8.3 Member Function Documentation 19
5.8.3.1 CreateRouteForSequenceID 19
5.8.3.2 CreateRoutingTable 19
5.9 NthEvent Class Reference 20
5.9.1 Detailed Description 20
5.9.2 Constructor & Destructor Documentation 20
5.9.2.1 NthEvent 20
5.9.3 Member Function Documentation 21
5.9.3.1 filter 21
5.10 artdaq::NthEventTransfer Class Reference 21
5.10.1 Detailed Description 22
5.10.2 Constructor & Destructor Documentation 22
5.10.2.1 NthEventTransfer 22
5.10.3 Member Function Documentation 22

5.10.3.1 destination_rank 22
5.10.3.2 isRunning 22
5.10.3.3 receiveFragment 23
5.10.3.4 receiveFragmentData 23
5.10.3.5 receiveFragmentHeader 23
5.10.3.6 source_rank 24
5.10.3.7 transfer_fragment_min_blocking_mode 24
5.10.3.8 transfer_fragment_reliable_mode 24
5.11 demo::RootApplication Class Reference 24
5.11.1 Detailed Description 25
5.11.2 Constructor & Destructor Documentation 25
5.11.2.1 RootApplication 25
5.11.3 Member Function Documentation 25
5.11.3.1 analyze 25
5.12 demo::ToyDump Class Reference 26
5.12.1 Detailed Description 26
5.12.2 Constructor & Destructor Documentation 26
5.12.2.1 ToyDump 26
5.12.3 Member Function Documentation 27
5.12.3.1 analyze 27
5.12.3.2 endSubRun 27
5.13 ToyHardwareInterface Class Reference 27
5.13.1 Detailed Description 28
5.13.2 Member Enumeration Documentation 28
5.13.2.1 DistributionType 28
5.13.3 Constructor & Destructor Documentation 28
5.13.3.1 ToyHardwareInterface 28
5.13.4 Member Function Documentation 29
5.13.4.1 AllocateReadoutBuffer 29
5.13.4.2 BoardType 29
5.13.4.3 FillBuffer 29
5.13.4.4 FreeReadoutBuffer 29
5.13.4.5 NumADCBits 29
5.13.4.6 SerialNumber 30
5.14 demo::ToySimulator Class Reference 30
5.14.1 Detailed Description 30
5.14.2 Constructor & Destructor Documentation 31

5.14.2.1 ToySimulator 31
5.15 demo::UDPReceiver Class Reference 31
5.15.1 Detailed Description 31
5.15.2 Constructor & Destructor Documentation 32
5.15.2.1 UDPReceiver 32
5.16 demo::WFViewer Class Reference 32
5.16.1 Detailed Description 33
5.16.2 Constructor & Destructor Documentation 33
5.16.2.1 WFViewer 33
5.16.3 Member Function Documentation 33
5.16.3.1 analyze 33

Index **34**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

anonymous_namespace{AsciiSimulator_generator.cc} 7
artdaq
    The artdaq namespace 7
demo
    Namespace used to differentiate the artdaq_demo version of GetPackageBuildInfo from other versions present in the system 8

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- CommandableFragmentGenerator
  - demo::AsciiSimulator 12
  - demo::ToySimulator 30
  - demo::UDPReceiver 31
- demo::CommandPacket 15
- EDAnalyzer
  - demo::ASCIIDump 11
  - demo::CheckIntegrity 14
  - demo::DemoViewer 16
  - demo::RootApplication 24
  - demo::ToyDump 26
  - demo::WFViewer 32
- EDFilter
  - NthEvent 20
- demo::GetPackageBuildInfo 17
- OutputModule
  - art::EventReporterOutput 17
- RoutingManagerPolicy
  - demo::MisbehaviorTest 18
- ToyHardwareInterface 27
- TransferInterface
  - artdaq::NthEventTransfer 21

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

demo::ASCIIDump
An art::EDAnalyzer meant for decoding demo::ASCIIFragment objects 11

demo::AsciiSimulator
Generates ASCIIFragments filled with user-specified ASCII strings 12

demo::CheckIntegrity
Demonstration art::EDAnalyzer which checks that all ToyFragment ADC counts are in the defined range 14

demo::CommandPacket
Struct defining UDP packet used for communicating with data receiver 15

demo::DemoViewer
An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number) 16

art::EventReporterOutput
An art::OutputModule which does nothing, but reports seen events and their fragments. This module is designed for debugging purposes, where writing events into ROOT files or sending events down stream is not necessary 17

demo::GetPackageBuildInfo
Wrapper around the demo::GetPackageBuildInfo::getPackageBuildInfo function 17

demo::MisbehaviorTest
A test RoutingManagerPolicy which does various "bad" things, determined by configuration 18

NthEvent
An art::EDFilter module that passes one out of N events 20

artdaq::NthEventTransfer
Demonstration TransferInterface plugin showing how to discard events Intended for use in the transfer_to_dispatcher case, NOT for primary data stream! 21

demo::RootApplication
Provides a wrapper for displaying ROOT canvases 24

demo::ToyDump
An art::EDAnalyzer module designed to display the data from demo::ToyFragment objects 26

ToyHardwareInterface
JCF, Mar-17-2016: ToyHardwareInterface is meant to mimic a vendor-provided hardware API, usable within the the ToySimulator fragment generator. For purposes of realism, it's a C++03-style API, as opposed to, say, one based in C++11 capable of taking advantage of smart pointers, etc. An important point to make is that it has ownership of the buffer into which it dumps its data - so rather than use new/delete, use its functions AllocateReadoutBuffer/FreeReadoutBuffer 27
demo::ToySimulator
ToySimulator is a simple type of fragment generator intended to be studied by new users of artdaq as an example of how to create such a generator in the "best practices" manner. Derived from artdaq's CommandableFragmentGenerator class, it can be used in a full DAQ simulation, obtaining data from the ToyHardwareInterface class 30
demo::UDPReceiver
An artdaq::CommandableFragmentGenerator which receives data in the form of UDP datagrams . . 31
demo::WFViewer
An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number) 32

Chapter 4

Namespace Documentation

4.1 anonymous_namespace{AsciiSimulator_generator.cc} Namespace Reference

Functions

- template<typename T >
  T convertToASCII (std::string input)

  Convert sizeof(T) characters of a string to a number containing the ASCII representation of that string.

4.1.1 Function Documentation

4.1.1.1 template<typename T > T anonymous_namespace{AsciiSimulator_generator.cc}::convertToASCII (std::string *input*)

Convert sizeof(T) characters of a string to a number containing the ASCII representation of that string.

Template Parameters

T	Output type

Parameters

input	String to convert to ASCII-encoded number

Returns

ASCII-encoded number

Definition at line 26 of file AsciiSimulator_generator.cc.

4.2 artdaq Namespace Reference

The artdaq namespace.

Classes

- class NthEventTransfer

Demonstration TransferInterface plugin showing how to discard events Intended for use in the transfer_to_dispatcher case, NOT for primary data stream!

4.2.1 Detailed Description

The artdaq namespace.

4.3 demo Namespace Reference

Namespace used to differentiate the artdaq_demo version of GetPackageBuildInfo from other versions present in the system.

Classes

- class ASCIIDump

  An art::EDAnalyzer meant for decoding demo::ASCIIFragment objects.
- class CheckIntegrity

  Demonstration art::EDAnalyzer which checks that all ToyFragment ADC counts are in the defined range.
- class DemoViewer

  An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)
- class RootApplication

  Provides a wrapper for displaying ROOT canvases.
- class ToyDump

  An art::EDAnalyzer module designed to display the data from demo::ToyFragment objects.
- class WFViewer

  An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)
- struct GetPackageBuildInfo

  Wrapper around the demo::GetPackageBuildInfo::getPackageBuildInfo function.
- class AsciiSimulator

  Generates ASCIIFragments filled with user-specified ASCII strings.
- class ToySimulator

  ToySimulator is a simple type of fragment generator intended to be studied by new users of artdaq as an example of how to create such a generator in the "best practices" manner. Derived from artdaq's CommandableFragmentGenerator class, it can be used in a full DAQ simulation, obtaining data from the ToyHardwareInterface class.
- struct CommandPacket

  Struct defining UDP packet used for communicating with data receiver.
- class UDPReceiver

  An artdaq::CommandableFragmentGenerator which receives data in the form of UDP datagrams.
- class MisbehaviorTest

  A test RoutingManagerPolicy which does various "bad" things, determined by configuration.

Typedefs

- typedef artdaq::BuildInfo
  <&instanceName,
  artdaqcore::GetPackageBuildInfo,
  artdaqutilities::GetPackageBuildInfo,
  artdaq::GetPackageBuildInfo,
  coredemo::GetPackageBuildInfo,
  demo::GetPackageBuildInfo > ArtdaqDemoBuildInfo

  ArtdaqDemoBuildInfo is a BuildInfo type containing information about artdaq_core, artdaq, artdaq_core_demo and artdaq-_demo builds.
- typedef std::array< uint8_t, 1500 > packetBuffer_t

  An array of 1500 bytes (MTU length)
- typedef std::list< packetBuffer_t > packetBuffer_list_t

  A std::list of packetbuffer_t objects.

Enumerations

- enum CommandType : uint8_t { **Read** = 0, **Write** = 1, **Start_Burst** = 2, **Stop_Burst** = 3 }

  Enumeration describing valid command types.
- enum ReturnCode : uint8_t { **Read** = 0, **First** = 1, **Middle** = 2, **Last** = 3 }

  Enumeration describing status codes that indicate current sender position in the stream.
- enum DataType : uint8_t { **Raw** = 0, **JSON** = 1, **String** = 2 }

  Enumeration describing potential data types.

Variables

- static std::string instanceName = "ArtdaqDemoBuildInfo"

  Instance name for the artdaq_demo version of BuildInfo module.

4.3.1 Detailed Description

Namespace used to differentiate the artdaq_demo version of GetPackageBuildInfo from other versions present in the system.

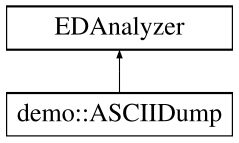
Chapter 5

Class Documentation

5.1 demo::ASCIIDump Class Reference

An art::EDAnalyzer meant for decoding demo::ASCIIFragment objects.

Inheritance diagram for demo::ASCIIDump:



Public Member Functions

- ASCIIDump (fhicl::ParameterSet const &pset)

    ASCIIDump Constructor.
- void analyze (art::Event const &evt) override

    Analyze an event. Called by art for each event in run (based on command line options)

5.1.1 Detailed Description

An art::EDAnalyzer meant for decoding demo::ASCIIFragment objects.

Definition at line 34 of file ASCIIDump_module.cc.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 demo::ASCIIDump::ASCIIDump (fhicl::ParameterSet const & *pset*) `[explicit]`

ASCIIDump Constructor.

Parameters

pset	ParameterSet used for configuring ASCIIDump. Parameter is "raw_data_label", default "daq".

Definition at line 60 of file ASCIIDump_module.cc.

5.1.3 Member Function Documentation

5.1.3.1 void demo::ASCIIDump::analyze (art::Event const & *evt*) `[override]`

Analyze an event. Called by art for each event in run (based on command line options)

Parameters

evt	The art::Event object to dump AsciiFragments from

Definition at line 66 of file ASCIIDump_module.cc.

The documentation for this class was generated from the following file:

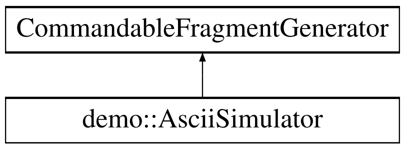
- artdaq_demo/artdaq-demo/ArtModules/ASCIIDump_module.cc

5.2 demo::AsciiSimulator Class Reference

Generates ASCIIFragments filled with user-specified ASCII strings.

`#include <artdaq-demo/Generators/AsciiSimulator.hh>`

Inheritance diagram for demo::AsciiSimulator:



Public Member Functions

- AsciiSimulator (fhicl::ParameterSet const &ps)

    AsciiSimulator Constructor.

5.2.1 Detailed Description

Generates ASCIIFragments filled with user-specified ASCII strings.

AsciiSimulator is a simple type of fragment generator intended to be studied by new users of artdaq as an example of how to create such a generator in the "best practices" manner. Derived from artdaq's CommandableFragmentGenerator class, it can be used in a full DAQ simulation, generating ASCII strings used for data validataion.

Definition at line 25 of file AsciiSimulator.hh.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 demo::AsciiSimulator::AsciiSimulator (fhicl::ParameterSet const & *ps*) `[explicit]`

AsciiSimulator Constructor.

Parameters

ps	fhicl::ParameterSet to configure AsciiSimulator. AsciiSimulator accepts the following configuration parameters: "throttle_usecs", how long to pause at the beginning of each call to getNext_, "string1" and "string2", strings to alternately put into the AsciiFragment

Definition at line 51 of file AsciiSimulator_generator.cc.

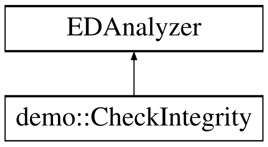
The documentation for this class was generated from the following files:

- artdaq_demo/artdaq-demo/Generators/AsciiSimulator.hh
- artdaq_demo/artdaq-demo/Generators/AsciiSimulator_generator.cc

5.3 demo::CheckIntegrity Class Reference

Demonstration art::EDAnalyzer which checks that all ToyFragment ADC counts are in the defined range.

Inheritance diagram for demo::CheckIntegrity:



Public Member Functions

- CheckIntegrity (fhicl::ParameterSet const &pset)

  CheckIntegrity Constructor.
- ∼CheckIntegrity () override=default

  Default destructor.
- void analyze (art::Event const &evt) override

  Analyze an event. Called by art for each event in run (based on command line options)

5.3.1 Detailed Description

Demonstration art::EDAnalyzer which checks that all ToyFragment ADC counts are in the defined range.

Definition at line 37 of file CheckIntegrity_module.cc.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 demo::CheckIntegrity::CheckIntegrity (fhicl::ParameterSet const & *pset*) `[explicit]`

CheckIntegrity Constructor.

Parameters

pset	ParameterSet used to configure CheckIntegrity

CheckIntegrity has the following paramters: "raw_data_label" (Default: "daq"): The label applied to data (usually "daq") "exception_on_integrity_failure" (Default: false): Whether to throw an exception (abort processing) if an integrity issue is found

Definition at line 72 of file CheckIntegrity_module.cc.

5.3.3 Member Function Documentation

5.3.3.1 void demo::CheckIntegrity::analyze (art::Event const & *evt*) `[override]`

Analyze an event. Called by art for each event in run (based on command line options)

Parameters

evt	The art::Event object containing ToyFragments to check

Definition at line 78 of file CheckIntegrity_module.cc.

The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/CheckIntegrity_module.cc

5.4 demo::CommandPacket Struct Reference

Struct defining UDP packet used for communicating with data receiver.

`#include <artdaq-demo/Generators/UDPReceiver.hh>`

Public Attributes

- CommandType type

  The type of this CommandPacket.
- uint8_t dataSize

  How many words of data are in the packet.
- uint64_t address

  The destination of the CommandPacket.
- uint64_t data [182]

  The data for the CommandPacket.

5.4.1 Detailed Description

Struct defining UDP packet used for communicating with data receiver.

Definition at line 67 of file UDPReceiver.hh.

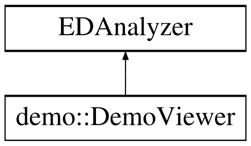
The documentation for this struct was generated from the following file:

- artdaq_demo/artdaq-demo/Generators/UDPReceiver.hh

5.5 demo::DemoViewer Class Reference

An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)

Inheritance diagram for demo::DemoViewer:



Public Member Functions

- DemoViewer (fhicl::ParameterSet const &p)

    DemoViewer Constructor.
- void **analyze** (art::Event const &e) override
- void **beginJob** () override
- void **beginRun** (art::Run const &e) override
- void **endRun** (art::Run const &e) override

5.5.1 Detailed Description

An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)

Definition at line 60 of file DemoViewer_module.cc.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 demo::DemoViewer::DemoViewer (fhicl::ParameterSet const & *p*) [explicit]

DemoViewer Constructor.

Parameters

p	ParameterSet used to configure DemoViewer

```
* DemoViewer accepts the following Parameters:
* "prescale" (REQUIRED): DemoViewer will only redraw historgrams once per this many events
* "num_x_plots": (Default: size_t::MAX_VALUE): Maximum number of columns of plots
* "num_y_plots": (Default: size_t::MAX_VALUE): Maximum number of rows of plots
* "raw_data_label": (Default: "daq"): Label under which artdaq data is stored
* "fragment_ids": (REQUIRED): List of ids to process. Fragment IDs are assigned by BoardReaders.
* "fileName": (Default: artdaqdemo_onmon.root): File name for output, if
* "write_to_file": (Default: false): Whether to write output histograms to "fileName"
*
```

Definition at line 125 of file DemoViewer_module.cc.

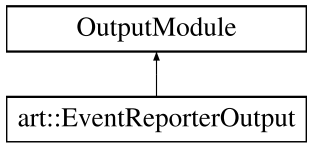
The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/DemoViewer_module.cc

5.6 art::EventReporterOutput Class Reference

An art::OutputModule which does nothing, but reports seen events and their fragments. This module is designed for debugging purposes, where writing events into ROOT files or sending events down stream is not necessary.

Inheritance diagram for art::EventReporterOutput:



Public Member Functions

- EventReporterOutput (fhicl::ParameterSet const &ps)

  EventReporterOutput Constructor.
- ∼EventReporterOutput () override

  EventReporterOutput Destructor.

5.6.1 Detailed Description

An art::OutputModule which does nothing, but reports seen events and their fragments. This module is designed for debugging purposes, where writing events into ROOT files or sending events down stream is not necessary.

Definition at line 60 of file EventReporterOutput_module.cc.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 art::EventReporterOutput::EventReporterOutput (fhicl::ParameterSet const & *ps*) `[explicit]`

EventReporterOutput Constructor.

Parameters

ps	ParameterSet used to configure EventReporterOutput

EventReporterOutput accepts no Parameters beyond those which art::OutputModule takes. See the art::OutputModule documentation for more details on those Parameters.

Definition at line 100 of file EventReporterOutput_module.cc.

The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/EventReporterOutput_module.cc

5.7 demo::GetPackageBuildInfo Struct Reference

Wrapper around the demo::GetPackageBuildInfo::getPackageBuildInfo function.

`#include <artdaq-demo/BuildInfo/GetPackageBuildInfo.hh>`

Static Public Member Functions

- static artdaq::PackageBuildInfo getPackageBuildInfo ()

  Gets the version number and build timestmap for artdaq_demo.

5.7.1 Detailed Description

Wrapper around the demo::GetPackageBuildInfo::getPackageBuildInfo function.

Definition at line 16 of file GetPackageBuildInfo.hh.

5.7.2 Member Function Documentation

5.7.2.1 static artdaq::PackageBuildInfo demo::GetPackageBuildInfo::getPackageBuildInfo ( ) [static]

Gets the version number and build timestmap for artdaq_demo.

Returns

An artdaq::PackageBuildInfo object containing the version number and build timestamp for artdaq_demo

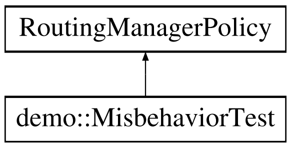
The documentation for this struct was generated from the following file:

- artdaq_demo/artdaq-demo/BuildInfo/GetPackageBuildInfo.hh

5.8 demo::MisbehaviorTest Class Reference

A test RoutingManagerPolicy which does various "bad" things, determined by configuration.

Inheritance diagram for demo::MisbehaviorTest:



Public Member Functions

- MisbehaviorTest (const fhicl::ParameterSet &ps)

  MisbehaviorTest Constructor.
- ∼MisbehaviorTest () override=default

  MisbehaviorTest default Destructor.
- void CreateRoutingTable (artdaq::detail::RoutingPacket &table) override

  Use the current tokens to add entries to the routing table.
- artdaq::detail::RoutingPacketEntry CreateRouteForSequenceID (artdaq::Fragment::sequence_id_t seq, int requesting_rank) override

  Using the existing tokens, determine a route for a given Sequence ID.

5.8.1 Detailed Description

A test RoutingManagerPolicy which does various "bad" things, determined by configuration.

Definition at line 11 of file MisbehaviorTest_policy.cc.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 demo::MisbehaviorTest::MisbehaviorTest (const fhicl::ParameterSet & *ps*) [explicit]

MisbehaviorTest Constructor.

Parameters

ps	ParameterSet used to configure MisbehaviorTest

```
* Note that only one misbehavior can be configured at a time. MisbehaviorTest will work like NoOp_policy when not
* misbehaving MisbehaviorTest accepts the following Parameters: "misbehave_after_n_events" (Default: 1000): The
* threshold after which it will start misbehaving "misbehave_pause_time_ms" (Default: 0): If greater than 0, will
* pause this long before sending out table updates "misbehave_send_confliting_table_data" (Default: false): If
* true, will send a table that contains the same sequence ID being sent to two different EventBuilders
* "misbehave_send_corrupt_table_data" (Default: false): If true, will send a table that contains an entry created
* using rand(), rand() "misbehave_overload_event_builder" (Default: false): If true, will send a large number of
* events to one EventBuilder
```

Definition at line 63 of file MisbehaviorTest_policy.cc.

5.8.3 Member Function Documentation

5.8.3.1 artdaq::detail::RoutingPacketEntry demo::MisbehaviorTest::CreateRouteForSequenceID (artdaq::Fragment::sequence_id_t *seq*, int *requesting_rank*) [override]

Using the existing tokens, determine a route for a given Sequence ID.

Parameters

seq	Sequence ID to route
requesting_rank	Rank requesting the route

Returns

Routing information

Definition at line 130 of file MisbehaviorTest_policy.cc.

5.8.3.2 void demo::MisbehaviorTest::CreateRoutingTable (artdaq::detail::RoutingPacket & *table*) [override]

Use the current tokens to add entries to the routing table.

Parameters

table	Routing Table to append to

Definition at line 81 of file MisbehaviorTest_policy.cc.

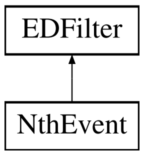
The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/RoutingPolicies/MisbehaviorTest_policy.cc

5.9 NthEvent Class Reference

An art::EDFilter module that passes one out of N events.

Inheritance diagram for NthEvent:



Public Member Functions

- NthEvent (fhicl::ParameterSet const &p)

    Construct the NthEvent Filter.
- NthEvent (NthEvent const &)=delete

    Plugins should not be copied or assigned.
- NthEvent (NthEvent &&)=delete

    Plugins should not be copied or assigned.
- NthEvent & operator= (NthEvent const &)=delete

    Plugins should not be copied or assigned.
- NthEvent & operator= (NthEvent &&)=delete

    Plugins should not be copied or assigned.
- bool filter (art::Event &e) override

    Perform the filtering. NthEvent module passes events where event number % nth == 0.

5.9.1 Detailed Description

An art::EDFilter module that passes one out of N events.

Definition at line 28 of file NthEvent_module.cc.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 NthEvent::NthEvent (fhicl::ParameterSet const & *p*) `[explicit]`

Construct the NthEvent Filter.

Parameters

p	fhicl::ParameterSet for configuring the filter. Parameter "nth", mod of events to pass

Definition at line 63 of file NthEvent_module.cc.

5.9.3 Member Function Documentation

5.9.3.1 bool NthEvent::filter (art::Event & *e*) [inline], [override]

Perform the filtering. NthEvent module passes events where event number % nth == 0.

Parameters

e	Event to filter

Returns

Whether the event passes the filter

Definition at line 68 of file NthEvent_module.cc.

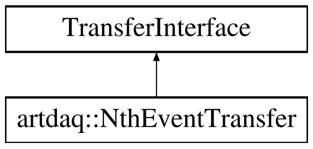
The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/NthEvent_module.cc

5.10 artdaq::NthEventTransfer Class Reference

Demonstration TransferInterface plugin showing how to discard events Intended for use in the transfer_to_dispatcher case, NOT for primary data stream!

Inheritance diagram for artdaq::NthEventTransfer:



Public Member Functions

- NthEventTransfer (fhicl::ParameterSet const &ps, artdaq::TransferInterface::Role role)

    NthEventTransfer Constructor.
- TransferInterface::CopyStatus transfer_fragment_min_blocking_mode (artdaq::Fragment const &fragment, size_t send_timeout_usec) override

    Transfer a Fragment to the destination. May not necessarily be reliable, but will not block longer than send_timeout_usec.
- TransferInterface::CopyStatus transfer_fragment_reliable_mode (artdaq::Fragment &&fragment) override

    Copy a fragment, using the reliable channel. moveFragment assumes ownership of the fragment.
- int receiveFragment (artdaq::Fragment &fragment, size_t receiveTimeout) override

    Receive a fragment from the transfer plugin.
- int receiveFragmentHeader (detail::RawFragmentHeader &header, size_t receiveTimeout) override

Receive a Fragment Header from the transport mechanism.
- int receiveFragmentData (RawDataType ∗destination, size_t wordCount) override

  Receive the body of a Fragment to the given destination pointer.
- int source_rank () const override

  Get the source rank from the physical transfer.
- int destination_rank () const override

  Get the destination rank from the physical transfer.
- bool isRunning () override

  Determine whether the TransferInterface plugin is able to send/receive data.
- void flush_buffers () override

  Flush any in-flight data. This should be used by the receiver after the receive loop has ended.

5.10.1 Detailed Description

Demonstration TransferInterface plugin showing how to discard events Intended for use in the transfer_to_dispatcher case, NOT for primary data stream!

Definition at line 27 of file NthEvent_transfer.cc.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 artdaq::NthEventTransfer::NthEventTransfer (fhicl::ParameterSet const & *ps*, artdaq::TransferInterface::Role *role*)

NthEventTransfer Constructor.

Parameters

ps	fhicl::ParameterSet used to configure TransferInterface. Contains "nth", the interval at which events will be transferred, and "physical_transfer_plugin", a table configuring the Transfer-Interface plugin used for those transfers
role	Either kSend or kReceive, see TransferInterface constructor

Definition at line 121 of file NthEvent_transfer.cc.

5.10.3 Member Function Documentation

5.10.3.1 int artdaq::NthEventTransfer::destination_rank ( ) const [inline], [override]

Get the destination rank from the physical transfer.

Returns

The destination rank from the physical transfer

Definition at line 99 of file NthEvent_transfer.cc.

5.10.3.2 bool artdaq::NthEventTransfer::isRunning ( ) [inline], [override]

Determine whether the TransferInterface plugin is able to send/receive data.

Returns

True if the TransferInterface plugin is currently able to send/receive data

Definition at line 105 of file NthEvent_transfer.cc.

5.10.3.3 int artdaq::NthEventTransfer::receiveFragment (artdaq::Fragment & *fragment,* size_t *receiveTimeout*) `[inline]`, `[override]`

Receive a fragment from the transfer plugin.

Parameters

fragment	Reference to output Fragment object
receiveTimeout	Timeout before aborting receive

Returns

Rank of sender or RECV_TIMEOUT

Definition at line 61 of file NthEvent_transfer.cc.

5.10.3.4 int artdaq::NthEventTransfer::receiveFragmentData (RawDataType ∗ *destination,* size_t *wordCount*) `[inline]`, `[override]`

Receive the body of a Fragment to the given destination pointer.

Parameters

destination	Pointer to memory region where Fragment data should be stored
wordCount	Number of words of Fragment data to receive

Returns

The rank the Fragment was received from (should be source_rank), or RECV_TIMEOUT

Definition at line 84 of file NthEvent_transfer.cc.

5.10.3.5 int artdaq::NthEventTransfer::receiveFragmentHeader (detail::RawFragmentHeader & *header,* size_t *receiveTimeout*) `[inline]`, `[override]`

Receive a Fragment Header from the transport mechanism.

Parameters

out	*header*	Received Fragment Header
	receiveTimeout	Timeout for receive

Returns

The rank the Fragment was received from (should be source_rank), or RECV_TIMEOUT

Definition at line 73 of file NthEvent_transfer.cc.

5.10.3.6 int artdaq::NthEventTransfer::source_rank ( ) const [inline], [override]

Get the source rank from the physical transfer.

Returns

The source rank from the physical transfer

Definition at line 93 of file NthEvent_transfer.cc.

5.10.3.7 TransferInterface::CopyStatus artdaq::NthEventTransfer::transfer_fragment_min_blocking_mode (artdaq::Fragment const & *fragment*, size_t *send_timeout_usec*) [override]

Transfer a Fragment to the destination. May not necessarily be reliable, but will not block longer than send_timeout_usec.

Parameters

fragment	Fragment to transfer
send_timeout_-usec	Timeout for send, in microseconds

Returns

CopyStatus detailing result of transfer

Definition at line 147 of file NthEvent_transfer.cc.

5.10.3.8 TransferInterface::CopyStatus artdaq::NthEventTransfer::transfer_fragment_reliable_mode (artdaq::Fragment && *fragment*) [override]

Copy a fragment, using the reliable channel. moveFragment assumes ownership of the fragment.

Parameters

fragment	Fragment to copy

Returns

CopyStatus (either kSuccess, kTimeout, kErrorNotRequiringException or an exception)

Definition at line 160 of file NthEvent_transfer.cc.

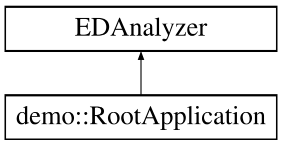
The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/TransferPlugins/NthEvent_transfer.cc

5.11 demo::RootApplication Class Reference

Provides a wrapper for displaying ROOT canvases.

Inheritance diagram for demo::RootApplication:



Public Member Functions

- RootApplication (fhicl::ParameterSet const &p)

    RootApplication Constructor.
- ∼RootApplication () override

    RootApplication Destructor.
- void beginJob () override

    Called by art at the beginning of the job. RootApplication will create a window unless one already exists and force_new == false.
- void analyze (art::Event const &e) override

    Called by art for each event.
- void endJob () override

    Called by art at the end of the job. RootApplication will close the findow if dont_quit == false.

5.11.1 Detailed Description

Provides a wrapper for displaying ROOT canvases.

Definition at line 26 of file RootApplication_module.cc.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 demo::RootApplication::RootApplication (fhicl::ParameterSet const & *p*) `[explicit]`

RootApplication Constructor.

Parameters

p	ParameterSet for configuring RootApplication

RootApplication accepts the following Paramters: "force_new" (Default: true): Always create a new window "dont_quit" (Default: false): Keep window open after art exits

Definition at line 74 of file RootApplication_module.cc.

5.11.3 Member Function Documentation

5.11.3.1 void demo::RootApplication::analyze (art::Event const & *e*) `[override]`

Called by art for each event.

Parameters

e	The art::Event object

RootApplication checks for ROOT system events, it does not touch the art::Event

Definition at line 82 of file RootApplication_module.cc.

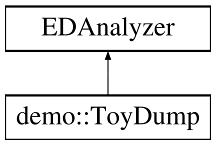
The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/RootApplication_module.cc

5.12 demo::ToyDump Class Reference

An art::EDAnalyzer module designed to display the data from demo::ToyFragment objects.

Inheritance diagram for demo::ToyDump:



Public Member Functions

- ToyDump (fhicl::ParameterSet const &pset)

    ToyDump Constructor.
- ∼ToyDump () override

    ToyDump Destructor.
- void analyze (art::Event const &evt) override

    Analyze an event. Called by art for each event in run (based on command line options)
- void endSubRun (art::SubRun const &sr) override

    Print summary information from a SubRun.

5.12.1 Detailed Description

An art::EDAnalyzer module designed to display the data from demo::ToyFragment objects.

Definition at line 38 of file ToyDump_module.cc.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 demo::ToyDump::ToyDump (fhicl::ParameterSet const & *pset*) `[explicit]`

ToyDump Constructor.

Parameters

pset	ParamterSet used to configure ToyDump

```
* ToyDump accepts the following Parameters:
* "raw_data_label" (Default: "daq"): The label used to identify artdaq data
* "num_adcs_to_print" (Default: 10): How many ADCs to print to screen from each ToyFragment (-1 to disable, 0 for
* all) "num_adcs_to_write" (Default: 0): How many ADCs to write to file from each ToyFragment (-1 to disable, 0 f
* all) "output_file_name" (Default: "out.bin"): File to write ADC values to "binary_mode" (Default: true): Whethe
* to write output in binary (true) or as tab-delimited ASCII text (false) "columns_to_display_on_screen" (Default
* 10): How many ADC values to print in each row when writing to stdout
```

Definition at line 91 of file ToyDump_module.cc.

5.12.3 Member Function Documentation

5.12.3.1 void demo::ToyDump::analyze (art::Event const & *evt*) [override]

Analyze an event. Called by art for each event in run (based on command line options)

Parameters

evt	The art::Event object to dump ToyFragments from

Definition at line 103 of file ToyDump_module.cc.

5.12.3.2 void demo::ToyDump::endSubRun (art::SubRun const & *sr*) [override]

Print summary information from a SubRun.

Parameters

sr	Subrun object

Definition at line 259 of file ToyDump_module.cc.

The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/ToyDump_module.cc

5.13 ToyHardwareInterface Class Reference

JCF, Mar-17-2016: ToyHardwareInterface is meant to mimic a vendor-provided hardware API, usable within the the Toy-Simulator fragment generator. For purposes of realism, it's a C++03-style API, as opposed to, say, one based in C++11 capable of taking advantage of smart pointers, etc. An important point to make is that it has ownership of the buffer into which it dumps its data - so rather than use new/delete, use its functions AllocateReadoutBuffer/FreeReadoutBuffer.

`#include <artdaq-demo/Generators/ToyHardwareInterface/ToyHardwareInterface.hh>`

Public Types

- enum DistributionType {
  DistributionType::uniform, DistributionType::gaussian, DistributionType::monotonic, DistributionType::uninitialized, **uninit2** }

  Allow for the selection of output distribution.

Public Member Functions

- ToyHardwareInterface (fhicl::ParameterSet const &ps)

  Construct and configure ToyHardwareInterface.
- void StartDatataking ()

  "StartDatataking" is meant to mimic actions one would take when telling the hardware to start sending data - the uploading of values to registers, etc.
- void StopDatataking ()

  Performs shutdown actions.
- void FillBuffer (char ∗buffer, size_t ∗bytes_read)

  Use configured generator to fill a buffer with data.
- void AllocateReadoutBuffer (char ∗∗buffer)

  Request a buffer from the hardware.
- void FreeReadoutBuffer (const char ∗buffer)

  Release the given buffer to the hardware.
- int SerialNumber () const

  Gets the serial number of the simulated hardware.
- int NumADCBits () const

  Get the number of ADC bits used in generating data.
- int BoardType () const

  Return the "board type" of the simulated hardware.

5.13.1 Detailed Description

JCF, Mar-17-2016: ToyHardwareInterface is meant to mimic a vendor-provided hardware API, usable within the the Toy-Simulator fragment generator. For purposes of realism, it's a C++03-style API, as opposed to, say, one based in C++11 capable of taking advantage of smart pointers, etc. An important point to make is that it has ownership of the buffer into which it dumps its data - so rather than use new/delete, use its functions AllocateReadoutBuffer/FreeReadoutBuffer.

Definition at line 22 of file ToyHardwareInterface.hh.

5.13.2 Member Enumeration Documentation

5.13.2.1 enum **ToyHardwareInterface::DistributionType** `[strong]`

Allow for the selection of output distribution.

Enumerator

uniform A uniform distribution.

gaussian A Gaussian distribution.

monotonic A monotonically-increasing distribution.

uninitialized A use-after-free expliot distribution.

Definition at line 83 of file ToyHardwareInterface.hh.

5.13.3 Constructor & Destructor Documentation

5.13.3.1 ToyHardwareInterface::ToyHardwareInterface (fhicl::ParameterSet const & *ps*) `[explicit]`

Construct and configure ToyHardwareInterface.

Parameters

ps	fhicl::ParameterSet with configuration options for ToyHardwareInterface

Definition at line 23 of file ToyHardwareInterface.cc.

5.13.4 Member Function Documentation

5.13.4.1 void ToyHardwareInterface::AllocateReadoutBuffer (char ** *buffer*)

Request a buffer from the hardware.

Parameters

buffer	(output) Pointer to buffer

Definition at line 250 of file ToyHardwareInterface.cc.

5.13.4.2 int ToyHardwareInterface::BoardType ( ) const

Return the "board type" of the simulated hardware.

Returns

A vendor-provided integer identifying the board type

Definition at line 258 of file ToyHardwareInterface.cc.

5.13.4.3 void ToyHardwareInterface::FillBuffer (char * *buffer,* size_t * *bytes_read*)

Use configured generator to fill a buffer with data.

Parameters

buffer	Buffer to fill
bytes_read	Number of bytes to fill

Definition at line 126 of file ToyHardwareInterface.cc.

5.13.4.4 void ToyHardwareInterface::FreeReadoutBuffer (const char * *buffer*)

Release the given buffer to the hardware.

Parameters

buffer	Buffer to release

Definition at line 256 of file ToyHardwareInterface.cc.

5.13.4.5 int ToyHardwareInterface::NumADCBits ( ) const

Get the number of ADC bits used in generating data.

Returns

The number of ADC bits used

Definition at line 302 of file ToyHardwareInterface.cc.

5.13.4.6 int ToyHardwareInterface::SerialNumber ( ) const

Gets the serial number of the simulated hardware.

Returns

Serial number of the simulated hardware

Definition at line 318 of file ToyHardwareInterface.cc.

The documentation for this class was generated from the following files:

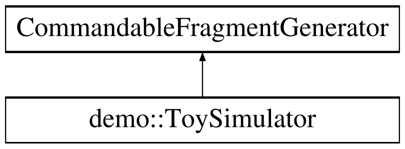
- artdaq_demo/artdaq-demo/Generators/ToyHardwareInterface/ToyHardwareInterface.hh
- artdaq_demo/artdaq-demo/Generators/ToyHardwareInterface/ToyHardwareInterface.cc

5.14 demo::ToySimulator Class Reference

ToySimulator is a simple type of fragment generator intended to be studied by new users of artdaq as an example of how to create such a generator in the "best practices" manner. Derived from artdaq's CommandableFragmentGenerator class, it can be used in a full DAQ simulation, obtaining data from the ToyHardwareInterface class.

`#include <artdaq-demo/Generators/ToySimulator.hh>`

Inheritance diagram for demo::ToySimulator:



Public Member Functions

- ToySimulator (fhicl::ParameterSet const &ps)

  ToySimulator Constructor.
- virtual ∼ToySimulator ()

  Shutdown the ToySimulator.

5.14.1 Detailed Description

ToySimulator is a simple type of fragment generator intended to be studied by new users of artdaq as an example of how to create such a generator in the "best practices" manner. Derived from artdaq's CommandableFragmentGenerator class, it can be used in a full DAQ simulation, obtaining data from the ToyHardwareInterface class.

ToySimulator is designed to simulate values coming in from one of two types of digitizer boards, one called "TOY1" and the other called "TOY2"; the only difference between the two boards is the # of bits in the ADC values they send. These values are declared as FragmentType enum's in artdaq-demo's artdaq-core-demo/Overlays/FragmentType.hh header.

Definition at line 35 of file ToySimulator.hh.

5.14.2 Constructor & Destructor Documentation

5.14.2.1 demo::ToySimulator::ToySimulator (fhicl::ParameterSet const & *ps*) [explicit]

ToySimulator Constructor.

Parameters

ps	ParameterSet used to configure ToySimulator

The ToySimulator FragmentGenerator accepts the following configuration paramters: "timestamp_scale_factor" (Default: 1): How much to increment the timestamp Fragment Header field for each event "distribution_type" (REQUIRED): Which type of distribution to use when generating data. See ToyHardwareInterface for more information "rollover_subrun_-interval" (Default: 0): If this ToySimulator has fragment_id 0, will cause the system to rollover subruns every N events. 0 (default) disables.

Definition at line 25 of file ToySimulator_generator.cc.

The documentation for this class was generated from the following files:

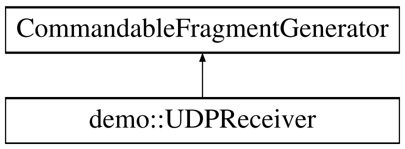
- artdaq_demo/artdaq-demo/Generators/ToySimulator.hh
- artdaq_demo/artdaq-demo/Generators/ToySimulator_generator.cc

5.15 demo::UDPReceiver Class Reference

An artdaq::CommandableFragmentGenerator which receives data in the form of UDP datagrams.

`#include <artdaq-demo/Generators/UDPReceiver.hh>`

Inheritance diagram for demo::UDPReceiver:



Public Member Functions

- UDPReceiver (fhicl::ParameterSet const &ps)

  UDPReceiver Constructor.

5.15.1 Detailed Description

An artdaq::CommandableFragmentGenerator which receives data in the form of UDP datagrams.

Definition at line 81 of file UDPReceiver.hh.

5.15.2 Constructor & Destructor Documentation

5.15.2.1 demo::UDPReceiver::UDPReceiver (fhicl::ParameterSet const & *ps*) [explicit]

UDPReceiver Constructor.

Parameters

ps	ParameterSet used to configure UDPReceiver

```
* UDPRecevier accepts the following Parameters:
* "port" (Default: 6343): The port on which to receive UDP data
* "ip" (Default: 127.0.0.1): The Address to bind to ("0.0.0.0" listens on all addresses)
* "send_CAPTAN_commands" (Default: false): Whether to send CommandPackets to start and stop the data flow
* "raw_output_enabled" (Default: false): Whether to write UDP data to disk as well as to EventBuilders
* "raw_output_path" (Default: "/tmp"): Directory to save raw output file (UDPReceiver-[ip]:[port].bin)
*
```

Definition at line 21 of file UDPReceiver_generator.cc.

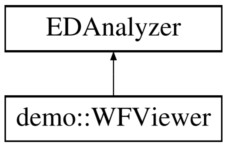
The documentation for this class was generated from the following files:

- artdaq_demo/artdaq-demo/Generators/UDPReceiver.hh
- artdaq_demo/artdaq-demo/Generators/UDPReceiver_generator.cc

5.16 demo::WFViewer Class Reference

An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)

Inheritance diagram for demo::WFViewer:



Public Member Functions

- WFViewer (fhicl::ParameterSet const &p)

  WFViewer Constructor.
- ∼WFViewer () override

  WFViewer Destructor.
- void analyze (art::Event const &e) override

  Analyze an event. Called by art for each event in run (based on command line options)
- void beginRun (art::Run const &) override

  Art calls this function at the beginning of the run. Used for set-up of ROOT histogram objects and to open the output file if one is specified.

5.16.1 Detailed Description

An example art analysis module which plots events both as histograms and event snapshots (plot of ADC value vs ADC number)

Definition at line 44 of file WFViewer_module.cc.

5.16.2 Constructor & Destructor Documentation

5.16.2.1 demo::WFViewer::WFViewer (fhicl::ParameterSet const & *p*) [explicit]

WFViewer Constructor.

Parameters

p	ParameterSet used to configure WFViewer

```
* WFViewer accepts the following Parameters:
* "prescale" (REQUIRED): WFViewer will only redraw historgrams once per this many events
* "digital_sum_only" (Default: false): Only create the histogram, not the event snapshot
* "num_x_plots": (Default: size_t::MAX_VALUE): Maximum number of columns of plots
* "num_y_plots": (Default: size_t::MAX_VALUE): Maximum number of rows of plots
* "raw_data_label": (Default: "daq"): Label under which artdaq data is stored
* "fragment_ids": (REQUIRED): List of ids to process. Fragment IDs are assigned by BoardReaders.
* "fileName": (Default: artdaqdemo_onmon.root): File name for output, if
* "write_to_file": (Default: false): Whether to write output histograms to "fileName"
*
```

Definition at line 117 of file WFViewer_module.cc.

5.16.3 Member Function Documentation

5.16.3.1 void demo::WFViewer::analyze (art::Event const & *e*) [override]

Analyze an event. Called by art for each event in run (based on command line options)

Parameters

e	The art::Event object to process, and display if it passes the prescale

Definition at line 268 of file WFViewer_module.cc.

The documentation for this class was generated from the following file:

- artdaq_demo/artdaq-demo/ArtModules/WFViewer_module.cc

Index

ASCIIDump
    demo::ASCIIDump, 11
AllocateReadoutBuffer
    ToyHardwareInterface, 29
analyze
    demo::ASCIIDump, 12
    demo::CheckIntegrity, 15
    demo::RootApplication, 25
    demo::ToyDump, 27
    demo::WFViewer, 33
anonymous_namespace{AsciiSimulator_generator.cc}, 7
    convertToASCII, 7
art::EventReporterOutput, 17
    EventReporterOutput, 17
artdaq, 7
artdaq::NthEventTransfer, 21
    destination_rank, 22
    isRunning, 22
    NthEventTransfer, 22
    receiveFragment, 23
    receiveFragmentData, 23
    receiveFragmentHeader, 23
    source_rank, 23
    transfer_fragment_min_blocking_mode, 24
    transfer_fragment_reliable_mode, 24
AsciiSimulator
    demo::AsciiSimulator, 13

BoardType
    ToyHardwareInterface, 29

CheckIntegrity
    demo::CheckIntegrity, 14
convertToASCII
    anonymous_namespace{AsciiSimulator_generator.-cc}, 7
CreateRouteForSequenceID
    demo::MisbehaviorTest, 19
CreateRoutingTable
    demo::MisbehaviorTest, 19

demo, 8
demo::ASCIIDump, 11
    ASCIIDump, 11
    analyze, 12
demo::AsciiSimulator, 12
    AsciiSimulator, 13
demo::CheckIntegrity, 14
    analyze, 15
    CheckIntegrity, 14
demo::CommandPacket, 15
demo::DemoViewer, 16
    DemoViewer, 16
demo::GetPackageBuildInfo, 17
    getPackageBuildInfo, 18
demo::MisbehaviorTest, 18
    CreateRouteForSequenceID, 19
    CreateRoutingTable, 19
    MisbehaviorTest, 19
demo::RootApplication, 24
    analyze, 25
    RootApplication, 25
demo::ToyDump, 26
    analyze, 27
    endSubRun, 27
    ToyDump, 26
demo::ToySimulator, 30
    ToySimulator, 31
demo::UDPReceiver, 31
    UDPReceiver, 32
demo::WFViewer, 32
    analyze, 33
    WFViewer, 33
DemoViewer
    demo::DemoViewer, 16
destination_rank
    artdaq::NthEventTransfer, 22
DistributionType
    ToyHardwareInterface, 28

endSubRun
    demo::ToyDump, 27
EventReporterOutput
    art::EventReporterOutput, 17

FillBuffer
    ToyHardwareInterface, 29
filter
    NthEvent, 21
FreeReadoutBuffer
    ToyHardwareInterface, 29

gaussian
    ToyHardwareInterface, 28
getPackageBuildInfo
    demo::GetPackageBuildInfo, 18

isRunning
    artdaq::NthEventTransfer, 22

MisbehaviorTest
    demo::MisbehaviorTest, 19
monotonic
    ToyHardwareInterface, 28

NthEvent, 20
    filter, 21
    NthEvent, 20
    NthEvent, 20
NthEventTransfer
    artdaq::NthEventTransfer, 22
NumADCBits
    ToyHardwareInterface, 29

receiveFragment
    artdaq::NthEventTransfer, 23
receiveFragmentData
    artdaq::NthEventTransfer, 23
receiveFragmentHeader
    artdaq::NthEventTransfer, 23
RootApplication
    demo::RootApplication, 25

SerialNumber
    ToyHardwareInterface, 30
source_rank
    artdaq::NthEventTransfer, 23

ToyHardwareInterface
    gaussian, 28
    monotonic, 28
    uniform, 28
    uninitialized, 28
ToyDump
    demo::ToyDump, 26
ToyHardwareInterface, 27
    AllocateReadoutBuffer, 29
    BoardType, 29
    DistributionType, 28
    FillBuffer, 29
    FreeReadoutBuffer, 29
    NumADCBits, 29
    SerialNumber, 30
    ToyHardwareInterface, 28
    ToyHardwareInterface, 28
ToySimulator
    demo::ToySimulator, 31
transfer_fragment_min_blocking_mode
    artdaq::NthEventTransfer, 24
transfer_fragment_reliable_mode
    artdaq::NthEventTransfer, 24

UDPReceiver
    demo::UDPReceiver, 32
uniform
    ToyHardwareInterface, 28
uninitialized
    ToyHardwareInterface, 28

WFViewer
    demo::WFViewer, 33