Table of Contents

Parameters for Modules (Release cycles 6_1_X and later)

Complete:

Goal of this page

This page lists and describes configurable parameters in important EDM modules in release cycle CMSSW_6_1_X and later cycles.

For release cycles CMSSW_3_8_X through CMSSW_6_0_X inclusive, refer to the following link in the TWIKI history:

https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideEDMParametersForModules?re

For releases prior to CMSSW_3_8_0, refer to the following link in the TWIKI history:

https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideEDMParametersForModules?re

edmPluginHelp

One might consider using the executable `edmPluginHelp` instead of this TWIKI. It provides information that complements the information on this TWIKI. It can be used to get information about the configurable parameters of any producer, filter, analyzer, output module, source, ESSource, ESProducer, or service, not just the ones related to the EDM or Framework. For `edmPluginHelp` to work, the function in the C++ code that describes the parameters of the module must have been implemented. The function is named "fillDescriptions". For almost all Framework related modules this function has been implemented. The number of other modules where it has been implemented is increasing.

For example for PoolSource, give the following command within the working area for a specific release:

```
edmPluginHelp -p PoolSource
```

Or for a brief version

```
edmPluginHelp -b -p PoolSource
```

`edmPluginHelp` gets its information from the same source as the configuration validation system. The list of parameters printed by edmPluginHelp will always automatically be consistent with the list of parameters allowed by the validation. With the TWIKI, there is always a possibility that we forgot to update it when a new parameter was added.

There are advantages to the TWIKI. The output of `edmPluginHelp` is intentionally more terse and has less information than this TWIKI. If a mistake is found the TWIKI can be immediately corrected. The `edmPluginHelp` documentation is hard coded in the C++ and cannot be fixed until a new

release is made.

We try to keep both edmPluginHelp and this TWIKI up to date and consistent, but if you notice an error in the output of edmPluginHelp or this TWIKI, post it to the Framework hypernews so it can be fixed.

The options Parameter Set

One can define a parameter set named **options** at the top level that is attached directly to the process object. This parameter set will affect event processing for all modules. All the parameters within "options" are optional as is the parameter set itself. For example, it might look like this in a top level configuration file:

```
process.options = cms.untracked.PSet(
  fileMode = cms.untracked.string('FULLMERGE')
)
```

(A new option will be added in release 10_1_X soon, you will be able to use "edmPluginHelp -t options" or "edmPluginHelp -b -t options" to print out the description of the parameters from the command line.)

name	typeOfParameter	description
numberOfThreads	untracked unsigned int	Defaults to 1. If set to zero, we let TBB use its default which is normally th number of CPUs on the machine. Can be overridden b a command line argument. Available since 7_0_X.
numberOfStreams	untracked unsigned int	By default or if set to 0, the number of streams will b the same as the number of threads. Available since 7_0_X.
numberOfConcurrentRuns	untracked unsigned int	Defaults to 1. Currently if this is set, it must always be set to 1. There are plans to allow higher values in th future.
numberOfConcurrentLuminosityBlocks	untracked unsigned int	Defaults to 1. If set to zero, then uses the number c concurrent runs. Available since 10_1_X.
wantSummary	untracked bool	If true, a trigger summary report is written at the end of the job (to the MessageLogger FwkSummary category, which goes to std::cerr with the MessageLogger defaults). Default is false.
fileMode	untracked string	The two possible values are FULLMERGE and NOMERGE. The

		default mode is FULLMERGE. I NOMERGE mode, all output files are closed and new one opened each time a new input file is opened. This behavio since 3_8_0.
forceEventSetupCacheClearOnNewRun	untracked bool	Default is false.
throwIfIllegalParameter	untracked bool	Use this to disable exceptions for all configuration validation errors related to illegal parameters. Defaults to true Available since 4_4_X.
printDependencies	untracked bool	If true, information about the data dependencies of eac module in the configuration will be output to the log file. Default false. Available since 8_1_X.
sizeOfStackForThreadsInKB	untracked unsigned int	Default of 10 MB is hard coded in cmsRun.cpp. Can be overridden on command line.
The values the next four parameters hold are exception categories. See additional comments below.		
Rethrow	untracked vstring	If an exception with this category is thrown, Event processing stops immediately If caught, exceptions are rethrown so that they rise t the top level, then an error message prints, and the process stops. There is an attempt to call endLuminosityBlock, endRun, close open files, and perfor other cleanup (although each cleanup step is attempted only once and may also fail)
SkipEvent	untracked vstring	If an exception with this category is thrown, it is caught and not rethrown. Processing is immediately stopped for the current path and no other regular paths are executed. TriggerResults is inserted into the Event (in TriggerResults the current path has the status Exception and paths not run have the status Ready). EndPaths are run, because this is necessary to keep th ROOT "Events" tree balanced in the event that fast

		cloning is taking place. Processing then proceeds normally with the next event or whatever comes next. However, see notes 1 and 2 below.
FailPath	untracked vstring	If an exception with this category is thrown, it is caught and not rethrown. The current regular path is recorded as failed in TriggerResults as if a filte had returned false (unless the filter was vetoed, in that case true) and stopped processing on that path in the same position as the module that threw the exception. No more modules are executed on the path. Event processing continues with the next path. If the module is encountered again on a different path for the same event, it will not be rerun and the same exception will be rethrown and handled the same way. However, see notes 1 and 2 below
IgnoreCompletely	untracked vstring	If an exception of this category is thrown, it is caught and not rethrown. An attempt is made to ignore th exception and continue processing the next module i the same path. However, see note 1 below.
canDeleteEarly	untracked vstring	Branch names of products tha the Framework can try to delete before the end of the Event
allowUnscheduled	untracked bool	Obsolete in 9_1_X and later releases. It has no effect but as a matter of cleanup should be deleted from any configurations where it occurs. Unscheduled is alway enabled. In releases prior t 9_1_X the behavior is as follows. If true, the Framework will execute Producer modules which were declared in the configuratio but not scheduled in a path

		and which produce products requested by other modules. Defaults to false.
emptyRunLumiMode	untracked string	Obsolete in 9_3_0 and later releases. It has no effect but as a matter of cleanup should be deleted from any configurations where it occurs. In earlier releases, the three possible values ar **doNotHandleEmptyRunsAndLumis** **handleEmptyRuns**, and **handleEmptyRunsAndLumis** (the default). If **doNotHandleEmptyRunsAndLumis** is specified, there will be no beginRun or endRun transitions for runs containing no events, and no beginLuminosityBlock or endLuminosityBlock transitions for luminosity blocks containing no events. If **handleEmptyRuns** is specified, beginRun or endRu transitions will occur normally for runs with no events, but there will be no beginLuminosityBlock or endLuminosityBlock transitions for luminosity blocks containing no events. If **handleEmptyRunsAndLumis** (the default) is specified, all transitions will occur normally for empty runs or luminosity blocks. If, for example, run transitions are suppressed for empty runs, there will be no beginRun or endRun calls made for empty runs, and therefore no new run products will be produce for empty runs. However, thi parameter does not cause any existing run products from prior processes to be dropped.
makeTriggerResults	untracked bool	Obsolete. Has no effect but as a matter of cleanup shoul be deleted from any configurations where it occurs.

NOTE 1: The behavior after exceptions described above only applies during a call to a user module function while processing an Event (EDProducer::produce, EDFilter::filter, EDAnalyzer::analyze or OutputModule::write). At all other times the behavior is similar to the **Rethrow** behavior described above.

NOTE 2: The behavior described above for SkipEvent and FailPath does not apply if the exception is thrown while processing an event in a module on an EndPath. In this case, SkipEvent and FailPath behave as IgnoreCompletely. This is because all modules on an EndPath are considered independent.

The default for all exception categories is **Rethrow**. (This is correct as of the time this documentation was written, look in FWCore/Framework/src/Actions.cc for the actual defaults for a given release).

The maxEvents Parameter Set

One can define a parameter set named **maxEvents** at the top level that is attached directly to the process object. This parameter set will affect event processing for all modules. Both parameters within "maxEvents" are optional. as is the parameter set itself. For example, it might look like this in a top level configuration file:

```
process.maxEvents = cms.untracked.PSet(
  input = cms.untracked.int32(100),
  output = cms.untracked.int32(20)
)
```

The above parameter set will cause processing to terminate after 100 events are read from the primary input source. It will also cause processing to terminate after every output module in the configuration has output 20 events. Both parameters default to **-1**, which means no limit.

One can also customize the output limits to individual output modules. For example:

```
process.maxEvents = cms.untracked.PSet(
    input = cms.untracked.int32(100),
    output = cms.untracked.PSet(
       outputModule1 = cms.untracked.int32(20),
       outputModule2 = cms.untracked.int32(10)
    )
)
```

The above parameter set will cause processing to terminate after 100 events are read from the primary input source. It will also cause processing to terminate the output module with a module label of `outputModule1` has output 20 events and the output module with a module label of `outputModule12` has output 10 events.

Note that when processing reaches the point where the Framework says it has reached the output limit, all events already being processed will be completed. For example if you were processing 50 events concurrently, you

could get anywhere from 0 to 49 additional events written to output after the output limit is reached. This behavior only applies to the output parameter, not the input parameter.

The maxLuminosityBlocks Parameter Set

One can define a parameter set named **maxLuminosityBlocks** at the top level that is attached directly to the process object. This parameter set will affect event processing for all modules. For example, it might look like this in a top level configuration file:

```
process.maxLuminosityBlocks = cms.untracked.PSet(
  input = cms.untracked.int32(10)
)
```

The above parameter set will cause processing to terminate after 10 luminosity blocks are read from the primary input source.

Input Sources

Producer Sources (e.g., Generator Sources, Test Beam sources)

name	type	description
firstRun	untracked uint32	first run number to generate. Defaults to 1.
firstLuminosityBlock	untracked uint32	first luminosity block number to generate. Defaults to 1.
firstEvent	untracked uint32	first event number to generate. Defaults to 1.
numberEventsInRun	untracked uint32	number of events to generate per run. Defaults to maxEvents.
numberEventsInLuminosityBlock	untracked uint32	number of events to generate per lumi block. Defaults to maxEvents.
firstTime	untracked uint64	value before first event, in nanoseconds. Defaults to 1.
timeBetweenEvents	untracked uint64	increment before each event, in nanoseconds. Defaults to 1.
eventCreationDelay	untracked uint32	real-time delay between events, in microseconds. Defaults to 0.
processingMode	untracked string	"Runs" (i.e. don't process lumis or events), "RunsAndLumis" (i.e. don't process events), or "RunsLumisAndEvents". defaults to "RunsLumisAndEvents"
fileNames	untracked vstring	file names, URL's or other input entities, if needed. Defaults to no input files. Not used by EmptySource or most Generator Sources.

Pool Source

name	type	description
fileNames	untracked vstring	physical or logical file names. Required
processingMode	untracked string	"Runs" (i.e. don't process lumis or events) "RunsAndLumis" (i.e. don't process events), o "RunsLumisAndEvents". defaults to "RunsLumisAndEvents"
secondaryFileNames	untracked vstring	secondary physical or logical file names. defaults to empty
needSecondaryFileNames	untracked bool	throw if secondaryFileNames is no specified or empty. defaults to false
branchesMustMatch	untracked string	match requirement for branches for multiple input files. "permissive or "strict". defaults to "permissive"
setRunNumber	untracked uint32	overrides run number in file (usable only with generated data). default to 0 (no override)
skipBadFiles	untracked bool	skip missing or inaccessible files. defaults to false
skipEvents	untracked uint32	Number of initial events to skip. Does not skip Runs or LuminosityBlocks Does not count as skippe events, events that woul have been skipped for some other reason. Defaults to 0
eventsToSkip	untracked VEventRange	Vector of EventRanges to skip. Skips the Events i those EventRanges. Defaults to empty vector
lumisToSkip	untracked VLuminosityBlockRange	Vector of LuminosityBlockRanges to skip. Skips the LuminosityBlocks and the Events in those LuminosityBlockRanges. Defaults to empty vector If the vector is not empty, then Run's are also skipped if they do

		not contain any Lumi's that will be processed.
eventsToProcess	untracked VEventRange	Event ranges to process. Other events are not processed. Within a specific input file, onl Runs and LuminosityBlock associated with events that are both in ranges on the list and in that input file are processed Default is an empty vector and this paramete is ignored.
lumisToProcess	untracked VLuminosityBlockRange	LuminosityBlock ranges t process. Other luminosit blocks, and the events i them, are not processed. Default is an empty vector and this paramete is ignored. If the vecto is not empty, then Run's are also skipped if they do not contain any Lumi' that will be processed.
noEventSort	untracked bool	If true and constraints allow it, then events, runs, and lumis are processed in the order they appear in the TTree's. If false, they are sorted based on process history, run number, lumi number, and event number. One constraint is that everything in an input file associated with the same run number and process history must be processed contiguously. The other constraint is that everything in an input file associated with the same process history, run number and lumi number must be processed contiguously. Default is true. (see below for more details).
duplicateCheckMode	untracked string	Four possible values: "noDuplicateCheck", "checkEachFile", "checkEachRealDataFile",

		and "checkAllFilesOpened". In the first mode, no duplicate checking is done. In the other modes when a duplicate Event is found it is skipped. An event is duplicate if it has the same process history, run number, lumi number, and event number as an event already encountered. In mode "checkEachFile", each event is compared with the other events in the same input file. In the "checkEachRealDataFile", this is done only for real data files, nothing is done for simulation. In the final mode, events are compared with all other events in all the input files. The default is "checkAllFilesOpened"
firstRun	untracked uint32	Runs, LuminosityBlocks, and Events associated with run numbers less than firstRun are not processed. Defaults to 1
firstLuminosityBlock	untracked uint32	LuminosityBlocks and Events associated with firstRun and with luminosity block number less than firstLuminosityBlock are not processed. Defaults to 1.
firstEvent	untracked uint32	Events associated with firstRun and firstLuminosityBlock and also with event number less than firstEvent are not processed. Defaults to 1. There is a special case. If firstLuminosityBlock is 0, then Events associated with firstRun and with event number less than firstEvent are not processed.
inputCommands	untracked vstring	

		Used to drop selected branches on input. Defaults to "keep *"
dropDescendantsOfDroppedBranches	untracked bool	If a branch is dropped o input, drop other branches derived from this branch. Defaults to true.

See Example 6 in SWGuidePoolInputSources for an example of how to use **inputCommands** to implement drop on input.

noEventSort The processing order is determined as follows. If **noEventSort** is true, processing is ordered by the following criteria listed in order of precedence: input file, ProcessHistory/RunNumber ordered by first appearance in current file, ProcessHistory/RunNumber/LumiNumber ordered by first appearance in current file, and entry number in Events TTree. If **noEventSort** is false, then processing is ordered by the following criteria listed in order of precedence: input file, process history in order of appearance in the process, run number, luminosity number, event number, and Event TTree entry order.

branchesMustMatch See SWGuideMultipleFiles for more information

See also:

SWGuidePoolInputSources - How to specify a POOL/ROOT file as a source of events.

SWGuideMultipleFiles - How to read multiple POOL/ROOT files in one job.

Output Modules

PoolOutputModule

name	type	description
fileName	untracked string	output file or URL. Required
catalog	untracked string	output catalog name. Default: file:PoolFileCatalog.xml
logicalFileName	untracked string	logical file name. Defaults to empty string.
compressionLevel	untracked int32	ROOT compression level (0-9). Default: 7.
compressionAlgorithm	untracked string	ROOT compression algorithm. Allowed values "ZLIB" (default) or "LZMA". You probably want a compression level less than 7 with the "LZMA" algorithm
basketSize	untracked int32	default ROOT buffer size in bytes. Default: 16384.
splitLevel	untracked int32	default ROOT split level (0-99). Default: 99.
overrideInputFileSplitLevels		

	untracked bool	for copied branches, override ROOT split levels and basket sizes in input files. Default: false
maxSize	untracked int32	Max file size before starting new file, in kilobytes. Default: ~2TB.
fastCloning	untracked bool	If false, fast copying is disabled. Default: true.
sortBaskets	untracked string	"sortbasketsbyoffset", "sortbasketsbyentry" or "sortbasketsbybranch". Default: "sortbasketsbyoffset". Passed directly to ROOT when the Events tree is fast cloned.
dropMetaData	untracked string	"NONE", "DROPPED", "PRIOR", or "ALL" Default: "NONE".
outputCommands	untracked vstring	Used to drop selected branches. Defaults to "keep *"
SelectEvents	untracked PSet	Used to select only events satisfying certain triggers. See Paths and trigger bits -

The **outputCommands** and **SelectEvents** parameters are supported for all output modules, not just **PoolOutputModule**.

For a more detailed explanation of **outputCommands**, see SWGuideSelectingBranchesForOutput - How to configure output modules to write specific branches

If **dropMetaData** is "ALL", all per product per event metadata will be dropped. If **dropMetaData** is "PRIOR", all per product per event metadata for products produced in prior processes will be dropped. If **dropMetaData** is "DROPPED", all per product per event metadata for dropped products produced in prior processes will be dropped.

compressionAlgorithm was not supported in releases before CMSSW_5_0_0_pre1.

The **maxSize** parameter is checked only when a primary input file is closed. So, events from the same input file will always go into the same output file, regardless of the value of **maxSize**.

The **catalog** and **logicalFileName** parameters are passed to the job report, and are otherwise unused.

See also Paths and trigger bits - How to configure output modules to write specific events

EDProducers

EDAnalyzers

EventContentAnalyzer

This module prints to standard out a list of all products within the Event. A 'verbose' option will actually print the contents of the requested products.

name	type	description
indentation	untracked string	string prepended to output (defaults to "++")
verbose	untracked bool	when true it prints the contents of Event products. Default is false.
verboseForModuleLabels	untracked vstring	prints the contents of the products made by those modules. If 'verbose=true' and this is empty, print contents of all products. Default is empty.
verboseIndentation	untracked string	string used to indent verbose information (defaults to " ")

EventSetupRecordDataGetter

This module can be configured to get any data from any EventSetup Record.

name	type	description
verbose	untracked bool	when true it prints a statement for each successful get. Default is false.
toGet	VPSet	holds records and data to get

Parameters expected in *toGet* parameter

name	type	description
record	string	name of the record holding the data
data	vstring	names of the data types to get from the record. If a label is needed, give the type followed by '/' and then ending with the label.

EDFilters

ESSources

EmptyESSource

This module creates empty EventSetup Records using the IOVs provided.

name	type	description
recordName	string	name of the Record which should be created
iovIsRunNotTime	bool	?
firstValid	vuint32	a list of the run numbers for the start of each IOV transition. Set to '1' if you want valid for all time

Services

Tracer

Prints 'tracing' information to standard out which tells you what cmsRun is doing.

Use this command to get information about allowed parameters for this and other services.

```
edmPluginHelp -p Tracer
```

or

```
edmPluginHelp -b -p Tracer
```

Review Status

Reviewer/Editor and Date (copy from screen)	Comments
WilliamTanenbaum - 09 Nov 2015	added new parameter "printDependencies"
WilliamTanenbaum - 21 Aug 2013	removed obsolete parameter "parameterSetsMustMatch"
WilliamTanenbaum - 13 Nov 2012	Generated and External sources replaced by Producer sources
WilliamTanenbaum - 21 Aug 2012	add maxEvents and maxLuminosityBlocks parameter sets
WilliamTanenbaum - 19 Dec 2011	updating
WilliamTanenbaum - 31 Jan 2011	updating
WilliamTanenbaum - 22 jun 2009	updating
WilliamTanenbaum - 05 jun 2009	updating
WilliamTanenbaum - 30 Apr 2009	updating
WilliamTanenbaum - 09 Apr 2009	updating
WilliamTanenbaum - 20 Jan 2009	updating
WilliamTanenbaum - 15 Jan 2007	page last content editor
JennyWilliams - 31 Jan 2007	editing to include in SWGuide

Responsible: WilliamTanenbaum
Last reviewed by: Reviewer

This topic: CMSPublic > SWGuideEDMParametersForModules
Topic revision: r74 - 2019-04-05 - WDavidDagenhart