

Idmx-sw Tutorial Course: Basics

Tom Eichlersmith

eichl008@umn.edu

Idmx-sw Basics

University of Minnesota







Outline



Tom Eichlersmith (UMN)

Idmx-sw Basics







Pre-Requisites



May 27, 2021 3/23

Pre-Requisites

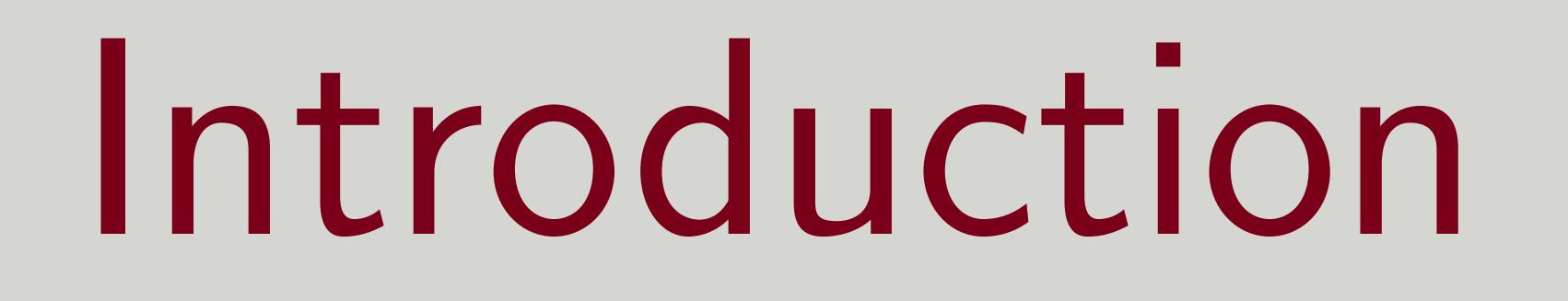
A container runner ¹ installed git version control installed Patience.

¹docker is common for personal computers and singularity is common for shared computing clusters. Tom Eichlersmith (UMN) May 27, 2021 Idmx-sw Basics









May 27, 2021 5/23



Big ldea

Simulate/Reconstruct/Analyze event-by-event

- - passengers to put onto the bus

Event Independence

Each event is **indepedent** of all other events in the software. This isn't necessarily true in real-life (**pileup**), but we have other methods for handling this.

Tom Eichlersmith (UMN)

Particles/Hits/Objects are put onto event bus (so they are passengers) **Processors** look at one event bus at a time in a certain sequence Generally, processors take passengers off the bus to **analyze** and/or **produce**

Process knows the sequence and feeds the event buses to the processors







Compiling Idmx-sw





May 27, 2021 7/23



Compiling Idmx-sw

Demonstration

I'm going to run these commands while sharing my screen. They are here for your future reterence.

1	git clonerecurs
T	grt cronerecurs
2	cd ldmx -sw
3	<pre>source scripts/ldm:</pre>
4	mkdir build
5	cd build
6	ldmx cmake
7	ldmx make install

Tom Eichlersmith (UMN)



sive https://github.com/LDMX-Software/ldmx-sw.git

x-env. sh









Configuration Scripts



May 27, 2021 9/23



Purpose

We have to tell the software **a** lot of things... How many events?

- Where is the data to input (if any)?
- Where should we put the output data (histograms or events)? Which processors to run?
- L. and much more!

Configuration Script

Today : A basic configuration script that runs a simulation of electrons entering the LDMX detector design and then puts those events through some of our reconstruction pipeline.

Tom Eichlersmith (UMN)

Idmx-sw Basics

A python3 script which is run by the software before grabbing the necessary parameters.

May 27, 2021 10 / 23





H NOCESS

The pass name Input and output files Histogram file Maximum number of events Logging frequency

Tom Eichlersmith (UMN)

11 12 13

Idmx-sw Basics

14

1 from LDMX.Framework import ldmxcfg 2 # create my process object 3 p = ldmxcfg.Process('sim') 4 # how many events to process? 5 p.maxEvents = 106 # we want to see every event 7 p.termLogLevel = 08 p.logFrequency = 1 9 # we also only have an output file 10 p.outputFiles = ["myFirstSim_" + str(p.maxEvents) + "_events.root"







Minimum Simulation

1	from LDMX.SimCore
2	<pre>import LDMX.Ecal.E</pre>
3	mySim = simulator.
4	# get the path to
5	mySim.setDetector(
6	# Get a pre-writter
7	from LDMX.SimAppli
8	<pre>mySim.generators =</pre>
9	# add your configu:
10	p.sequence.append(
11	# can still edit p
12	mySim.verbosity =

Tom Eichlersmith (UMN)

import simulator EcalGeometry # geometry required by sim simulator('mySim') the installed detector description 'ldmx-det-v12') en generator Lcation import generators as gen [gen.single_4gev_e_upstream_tagger()] ired simulation to the sequence mySim

barameters after adding to sequence









Reconstruction

1	#	1]	mJ	Э С) r	t		С	h	l	р		g	e	0	m
2	im	p	0]	ct		L	D	M	Х		E	С	a			e
3	im	p	0]	ct		L	D	M	Х		H	С	a			H
4	im	p	0]	ct		L	D	M	Х		H	С	a			h
5	#	1]	mJ	Э С) r	t		р	r	0	С	e	S	S	0	r
6	im	p	0]	ct		L	D	M	Х		E	С	a			d
7	im	p	0]	ct		L	D	M	Х		H	С	a			d
8	#	a	dc	3	t	h	e	m		t	0		t	h	e	
9	p.	S	e (Jľ	le	n	С	e		e	X	t	e	n	d	
10				90	; a			d	l	g	l		E	С	a	
11				90	;a			d	l	g	l		E	С	a	
12					;a			d	l	g	l		H	С	a	
13			J		; a			d	l	g	l		H	С	a	
14])												
15	#	V	ie	3 T	T	С	0	n	f	l	g	U	r	a	t	1
16	p.	p	al	JS	5 e											

conditions etrv

- ecal_hardcoded_conditions IcalGeometry
- ncal_hardcoded_conditions templates
- ligi as ecal_digi
- ligi as hcal_digi
- sequence
- LDigiProducer(),
- LRecProducer(),
- LDigiProducer(),
- LRecProducer()

on before actually running















May 27, 2021

Let's Run It!

Demonstration I'm going to run these commands while sharing my screen. They are here for your future reference.

1 ldmx fire basic.py

Tom Eichlersmith (UMN)











Misspelling parameter names Idmx-sw gives each parameter a sensible default, so the process will run if a parameter is not given Misspelling a parameter means that your parameter value is just silently ignored Attempting to access files outside of directories mounted to container Will see "Not found/doesn't exist" errors even though you can see the file Try mounting the directory using ldmx mount Obviously lots more... Many quirks of Idmx-sw that we all need to learn Please put questions in slack and/or reach out to me If you can imagine a C++ solution, put your ideas in the GitHub issue tracker

Common Pitfalls in Config Files









Helpful Tips

- parameters

Overall

Remember that the configuration script is run as a python script *before* the parameters are taken for processing, so you can automate a lot of stuff.

Tom Eichlersmith (UMN)

os.path.realpath to get full path without symlinks to files argparse to configure your run by changing processors or passing input files or changing

before they have been taken for processing Define python functions (maybe even write your own python module) containing settings/processors/sequences/processes that you use a lot

Idmx-sw Basics

Never type out file paths or file names. Input files can be fed on the command line and output files can be formatted based off of input names and parameters.

- print ldmx classes to see what parameters are given to it after your python nonsense but





May 27, 2021 17 / 23

Python Analyses





Python Analyses

Good Starting Point

Easier to write and interact with \checkmark

after-the-fact for your reference.) repository.

Tom Eichlersmith (UMN)

\square Quick start-up and easy to develop \checkmark

Today : I will use a helper python module and write an analysis live! (The file will be shared The helper python module **•** EventTree is uploaded and is also available in the ldmx-analysis









C++ Processors





May 27, 2021



Tom, you just showed us how cool Python analyses are. Why use C++?

Reasons

My General Workflow

Tom Eichlersmith (UMN)

Access to more tools that Python (e.g. Detector ID Decoding) Generally easier to scale to larger samples

① Make sure configuration for generating data sample runs 2 Write short Python analysis to start looking at generated data

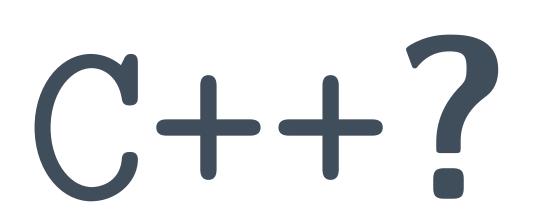
Idmx-sw Basics

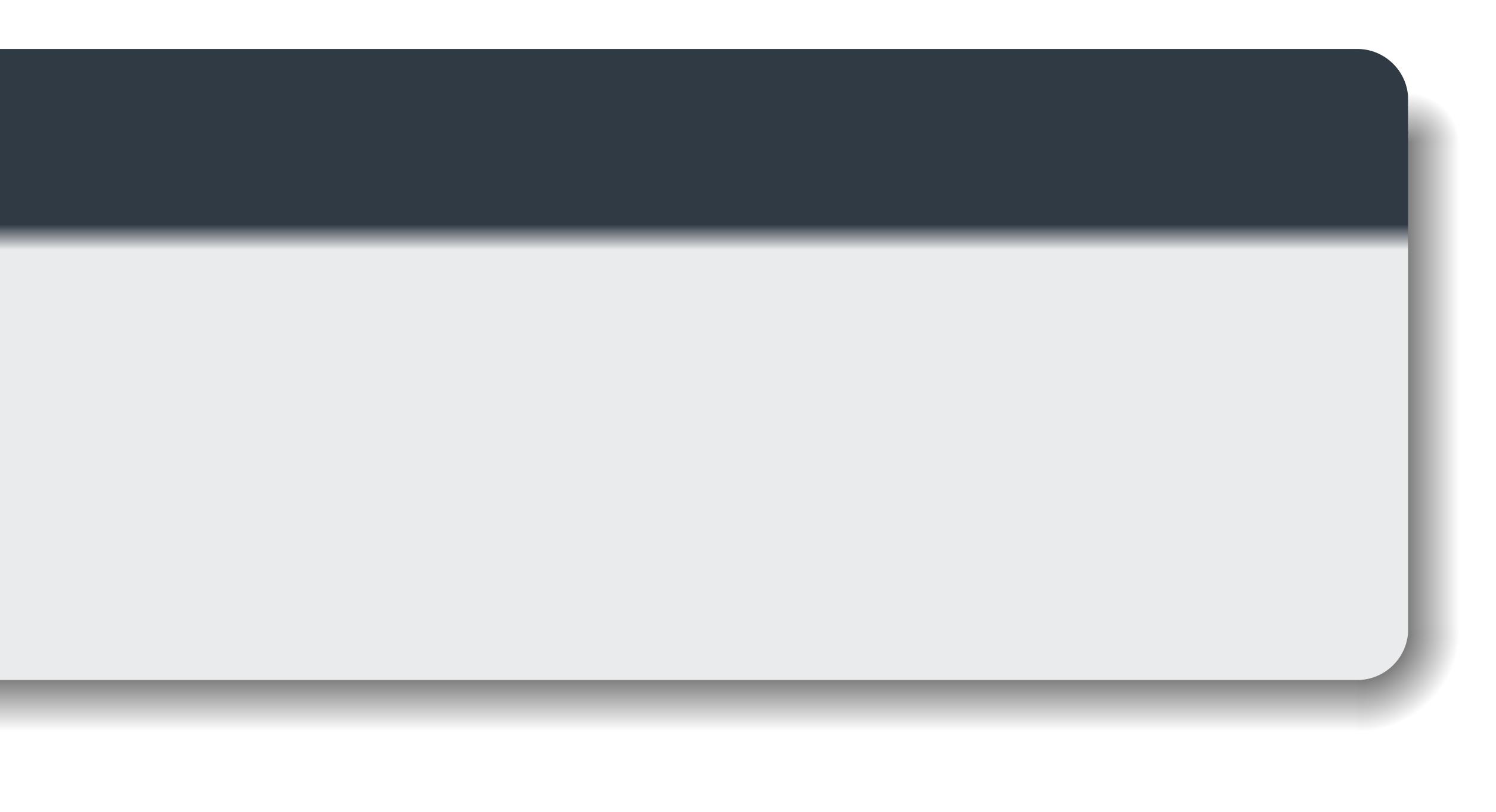
3 Switch to C++ when Python analysis is "getting out of hand" or I need a C++-only tool













Starting a C++ Analysis

General Outline

To Build (in README of ldmx-analysis)

1	mkdir	buil	d.,	cd	bu
2	ldmx	cmake			
3	ldmx	make	ins	stal	

Tom Eichlersmith (UMN)

Clone the ldmx-analysis repo and make your own branch (e.g. myusername-dev) Copy an existing analyzer's header (in include/Analysis) and source (in src/Analysis) to your own files in the same directories. Clean out the other analyzer code and start writing your own! Start make histograms/ntuplizing/skimming!

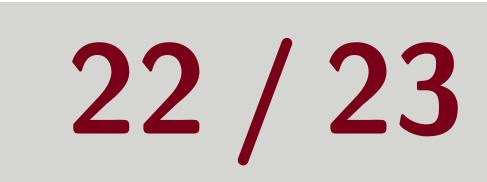
ild; #make a build directory











Analysis

Features to Look At:

- Vourself

Tom Eichlersmith (UMN)

Lots of examples in DQM module and ldmx-analysis repo.

Helpful for creating *lots* of histograms EcalPN analyzer in DQM module

NtupleManager: Create tree of once per event values in histogram file Helpful for higher-level physics variables when you don't know which you want to compare yet

ECalVetoAnalyzer in ldmx-analysis

setStorageHint: Skim events based on criteria in your processor Helpful for picking out specific (maybe frustrating?) events to look at in more detail

TriggerProcessor in Recon module

Process::keep: drop or keep event objects to save space in your output files

Can be done only in python configuration

HistogramPool: Manage histograms by name instead of managing histogram pointers







CMS Contractions of the test of the test of the test of test o

