



# Introduction to Flair and basic input

A very basic introduction to perform your first simulation

# A very short introduction

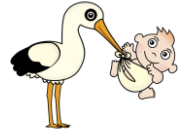
- Fluka's story begun a long time ago (1970s)...
  - ...no graphical interfaces, input and output via text file
- Inputfile can be very long > 50k lines
- Inputfile based on "cards": `.inp` file
- Each card has 1 name, 6 values (called WHATs), 1 string (called SDUM)
- Two examples of cards (the actual meaning is not relevant here):



<b>BEAMPOS</b>	<b>4750.5</b>	<b>130.0</b>	<b>4866.5</b>				<b>NEGATIVE</b>
<b>BEAM</b>	<b>-0.4</b>	<b>0.2</b>	<b>5.0</b>	<b>1.E-4</b>	<b>1.E-4</b>		<b>ELECTRON</b>
↑ Card name	↑ WHAT(1)	↑ WHAT(2)	↑ WHAT(3)	↑ WHAT(4)	↑ WHAT(5)	↑ WHAT(6)	↑ SDUM

# A very short introduction

- In 2006, Flair was born!



Fluka advanced interface

Inputfile creation

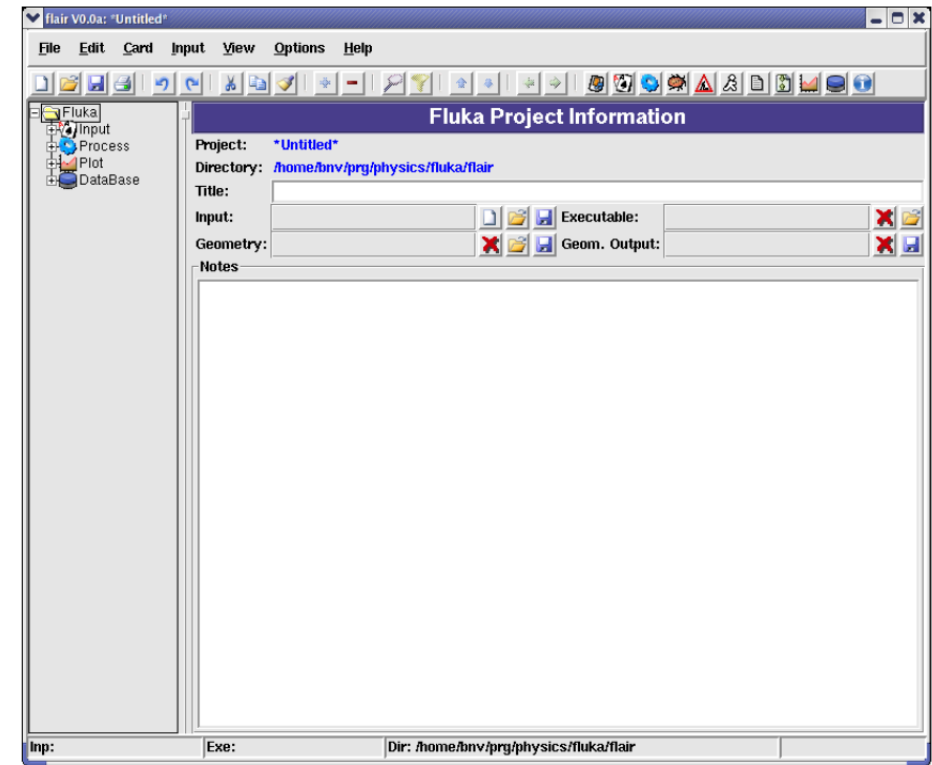
Geometry visualization and construction

Simulation execution

Results visualization

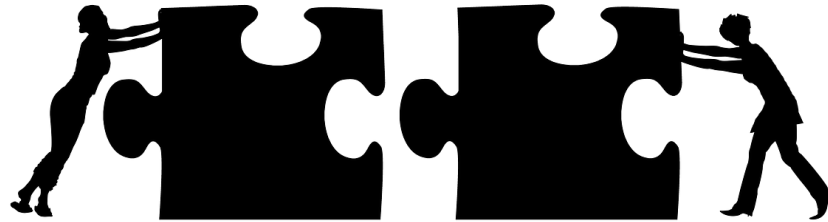
- Flair acts as an intermediate layer between the user and the inputfile
- It allows a user friendly editing of the Fluka input
- Based on a `.flair` file and generates the `.inp` file that is run by Fluka

## Flair ≠ Fluka



# Fluka & Flair

- Although strongly linked, they are two different things (`.inp`  $\neq$  `.flair`)
- Fluka is a Monte Carlo transport code based on text files
- Flair is a graphical interface to Fluka
- They work together but are different

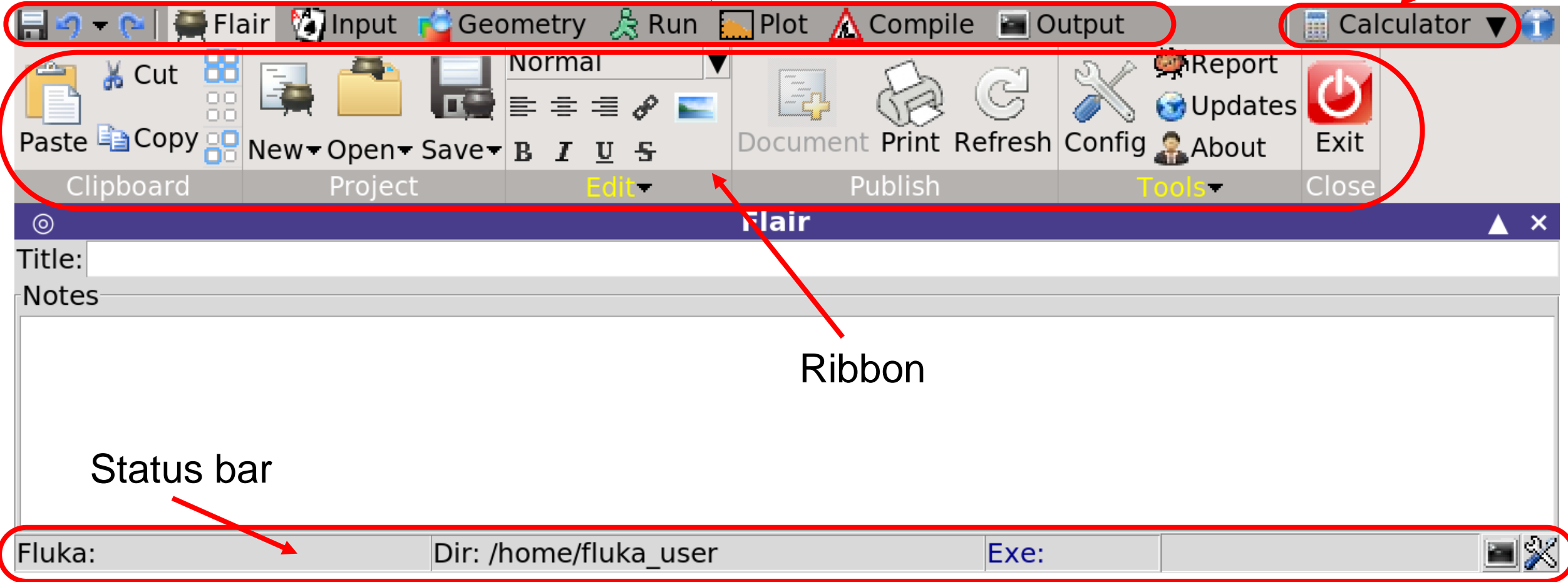


- It is possible to work with Fluka only using text editors (for expert or old users)
- Flair is not just a graphical interface for text editing
- Flair has a lot of features very useful for expert users
- This entire course will be based on Flair

# Starting Flair and basic nomenclature

“Ribbon tab” or “Program tab”

Dynamic tab



# What's for each tab?

The screenshot shows the Flair software interface with the following tabs circled in red: Input, Geometry, Run, Plot, Compile, and Output. Red arrows point from these tabs to the following descriptions:

- Input**: Build input and geometry
- Geometry**: Build geometry and plot results
- Run**: Run and merge results
- Plot**: Plot results
- Compile**: Compile own executable
- Output**: Visualize output files and messages

The interface also shows a menu bar with options like Cut, Copy, Paste, New, Open, Save, Document, Print, Refresh, Config, Report, Updates, About, and Exit. The status bar at the bottom displays 'Fluka: Dir: /home/fluka\_user Exe:'.

# The input as a text file

- Mentioned here just for completeness

```
TITLE
basic template
* Set the defaults for precision simulations
DEFAULTS PRECISIO
* Define the beam characteristics
BEAM
* Define the beam position
BEAMPOS
GEOBEGIN COMBNAME
0 0
* Black body
SPH blkbody 0.0 0.0 0.0 100000.0
* Void sphere
SPH void 0.0 0.0 0.0 10000.0
* Cylindrical target
RCC target 0.0 0.0 0.0 0.0 0.0 10.0 5.0
END
* Black hole
BLKBODY 5 +blkbody -void
* Void around
VOID 5 +void -target
* Target
TARGET 5 +target
END
GEOEND
* .....1.....2.....3.....4.....5.....6.....7...
ASSIGNMA BLCKHOLE BLKBODY
ASSIGNMA VACUUM VOID
ASSIGNMA COPPER TARGET
* Set the random number seed
RANDOMIZ 1.0
* Set the number of primary histories to be simulated in the run
START
STOP
-:--- basic.inp All (26,69) (Fluka)
```

.inp

.flair file includes  
info & instructions  
for the flair project

This course is based  
on the use of flair,  
no further mention  
of these text files

```
# flair project file
Version: 300
Mode: fluka
md5: c8e26fe184526e9282e8555b8fab2455
Input:
TITLE
fully-working template
#define pointless_define_1 10
#define pointless_define_2
*Set the defaults for precision simulations
DEFAULTS PRECISIO
*Define the beam characteristics
BEAM PROTON 0.8
*Define the beam position
BEAMPOS , 0. 0. -1.
GEOBEGIN COMBNAME
*Black body
SPH blkbody 0.0 0.0 0.0 100000.0
*Void sphere
SPH void 0.0 0.0 0.0 10000.0
*Cylindrical target
RCC target 0.0 0.0 0.0 0.0 0.0 10.0 5.0
END
*Black hole
REGION BLKBODY 5
+blkbody -void
*Void around
REGION VOID 5
+void -target
*Target
REGION TARGET 5
+target
END
GEOEND
*.....1.....2.....3.....4.....5.....6.....7...
ASSIGNMA , BLCKHOLE BLKBODY
ASSIGNMA , VACUUM VOID
ASSIGNMA , COPPER TARGET
USRBIN allpart 10 ALL-PART -21 6. 6. 11. -6. -6. -2. 120. 120. 130.
USRBIN edep 10 ENERGY -22 6. 6. 11. -6. -6. -2. 120. 120. 130.
*Set the random number seed
RANDOMIZ , 1.0
*Set the number of primary histories to be simulated in the run
START , 10000.
STOP
EndInput

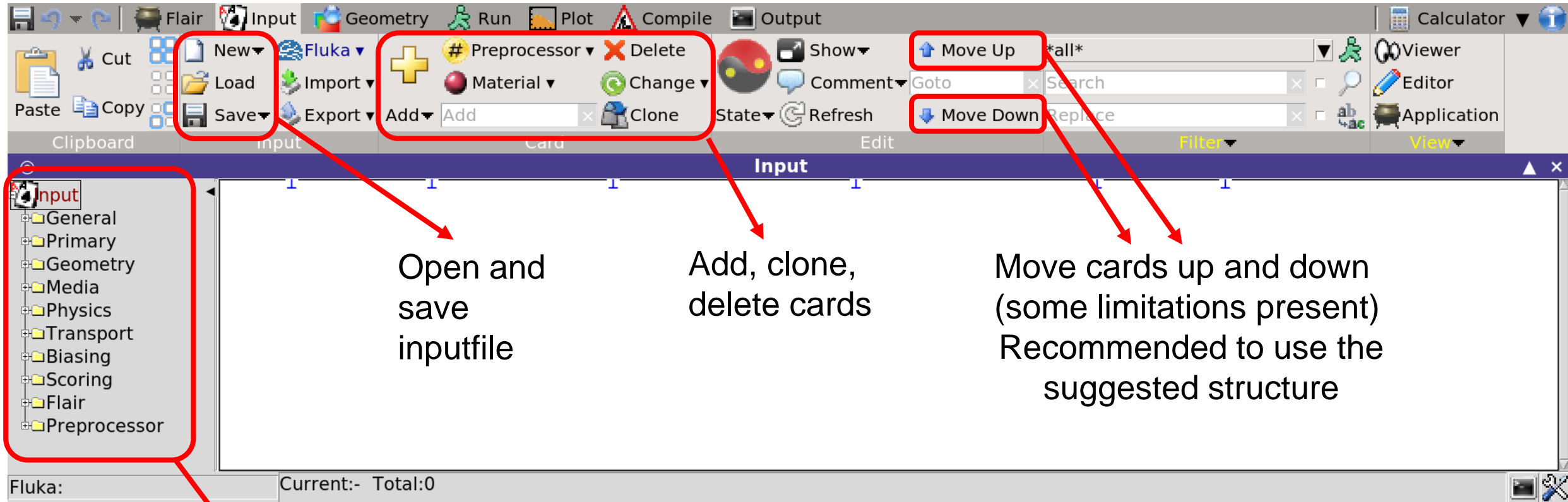
Page: Plot

# Run information
Run: <default>
End
Run: test/test
Define: pointless_define_2=10
Start: 1000
StartRun: 1598620157
End
Run: small_prod/small
Define: pointless_define_2=10
Start: 1000
Last: 1
```

.flair

# Input tab – 1: general info

- Standard looking “Windows” tab



Open and save inputfile

Add, clone, delete cards

Move cards up and down (some limitations present)  
Recommended to use the suggested structure

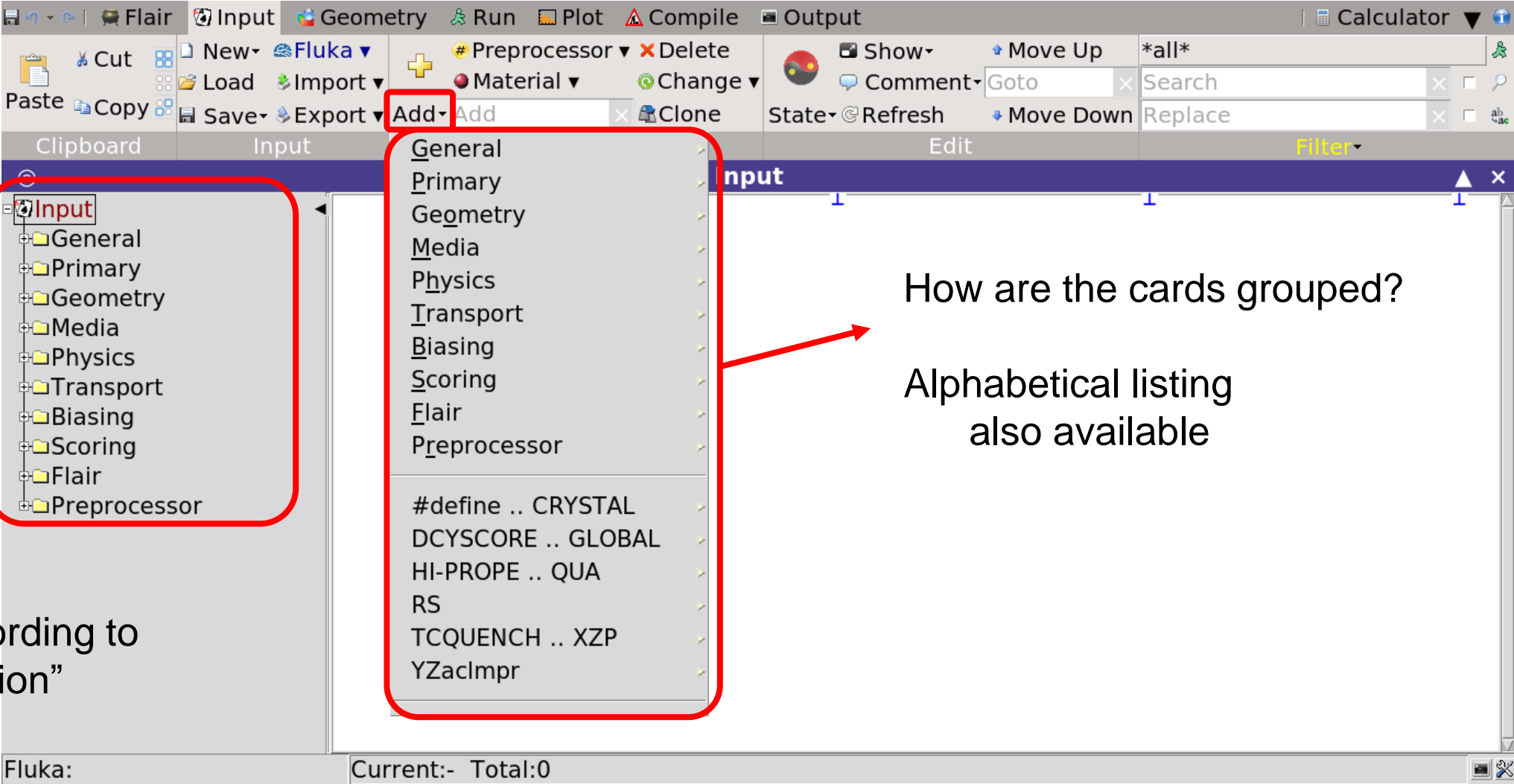
Inputfile tree

Cards grouped according to their “field of action”



# Input tab – 2: inputfile tree and card grouping

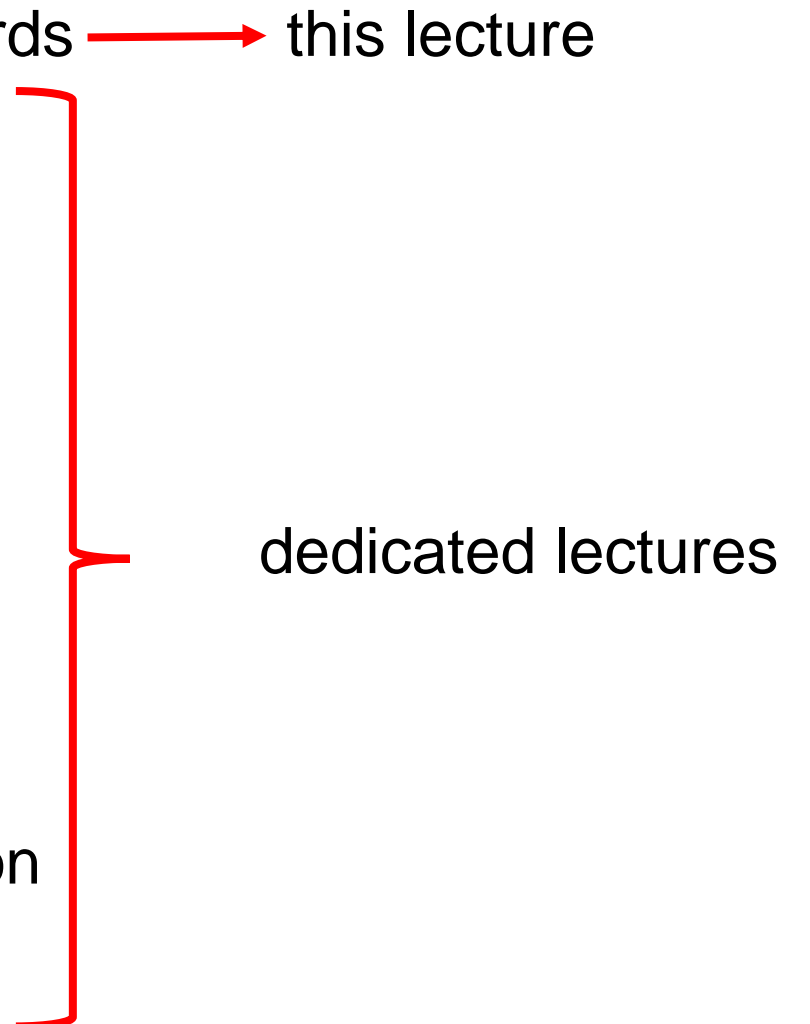
- Inputfile tree and card grouping



Inputfile tree  
Cards grouped according to their "field of action"

How are the cards grouped?  
Alphabetical listing also available

# Input tab – 3: inputfile tree and card grouping

- General: defaults selection and other general cards → this lecture
  - Primary: definition of the particle source
  - Geometry: definition of the geometry
  - Media: definition and assignment of “materials”
  - Physics: control specific physics processes
  - Transport: control specific transport details
  - Biasing: definition of biasing
  - Scoring: definition of estimators
  - Flair: definition of flair add-ons for visualization
  - Preprocessor: definition of preprocessor instructions
- dedicated lectures
- 

# Input tab – 4: General cards

**TITLE**

**START**

**STOP**

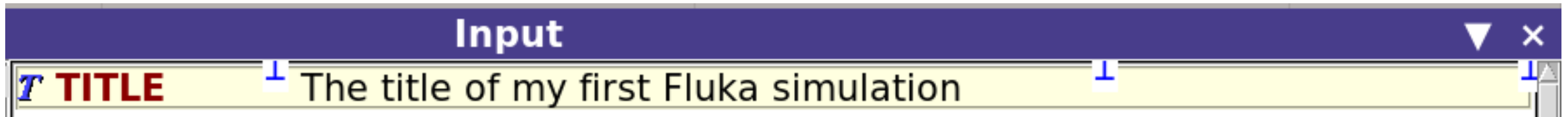
**RANDOMIZe**

**DEFAULTS**

---

## TITLE

- Not a mandatory card
- Allows to assign a title to the simulations
- The title is printed in the output files



# Input tab – 5: General cards

TITLE

START

STOP

RANDOMIZe

DEFAULTS

## START

- Actually listed among the “Primary” cards
- Allows to set the number of primary particles to be simulated
- Allows to set other parameters for advanced use

Set the number of primary histories to be simulated in the run

 **START**

No.: 10000.

Core: ▼

Time:

Report: default ▼

# Input tab – 6: General cards

**TITLE**

**START**

**STOP**

**RANDOMIZe**

**DEFAULTS**

---

## STOP

- Stop the execution of the program
- Not really mandatory (program stops at the end of the input)
- Can become handy for debugging purposes



**STOP**

# Input tab – 7: General cards

**TITLE**

**START**

**STOP**

**RANDOMIZe**

**DEFAULTS**

---

## RANDOMIZ

- Allows to initialize different random sequences
- For debugging purposes, the “random seed” must be the same
- Different “random seeds” are required in order to differentiate histories
- Flair takes care of the “random seeds” when spawning (see later)

Set the random number seed

 **RANDOMIZ**

Unit: 01 ▼

Seed: 123

# Input tab – 8: General cards

**TITLE**

**START**

**STOP**

**RANDOMIZe**

**DEFAULTS**

---

## DEFAULTS

- Allows to select the physics defaults (list of predefined defaults available)
- Physics defaults can be overridden with specific cards
- Can be preceded only by the **TITLE** and **GLOBAL** cards
- Given the progress over time in computer power, it is a reasonable approach to:
  - always select the most detailed physics defaults: **PRECISIO**
  - depending on the needs of the problem, override specific defaults

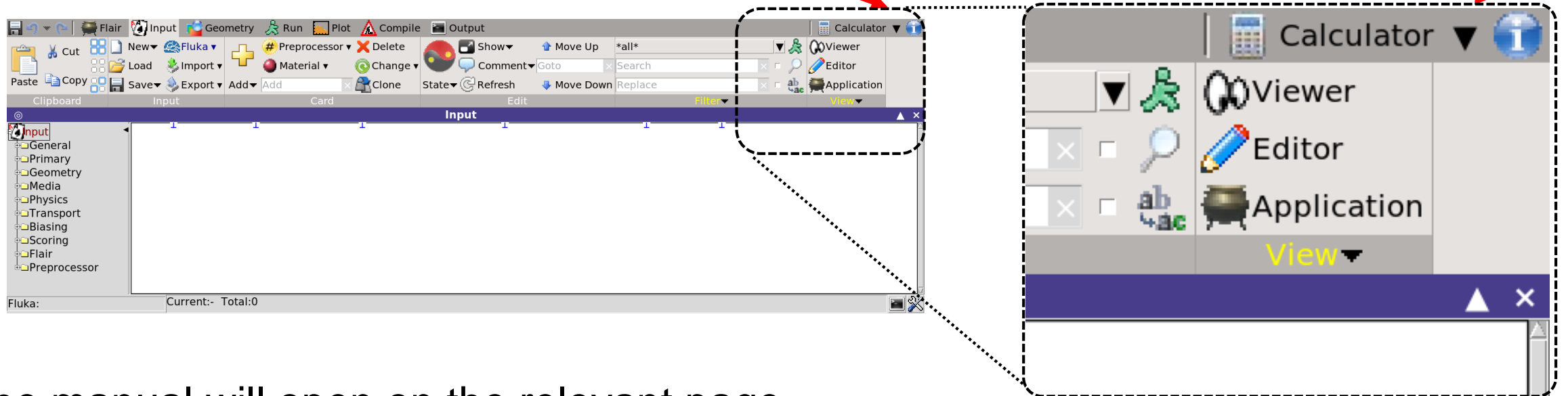
Set the defaults for precision simulations

 **DEFAULTS**

: PRECISIO ▼

# The manual

- Can be accessed using F1 button
- Can be accessed clicking on the “info” button

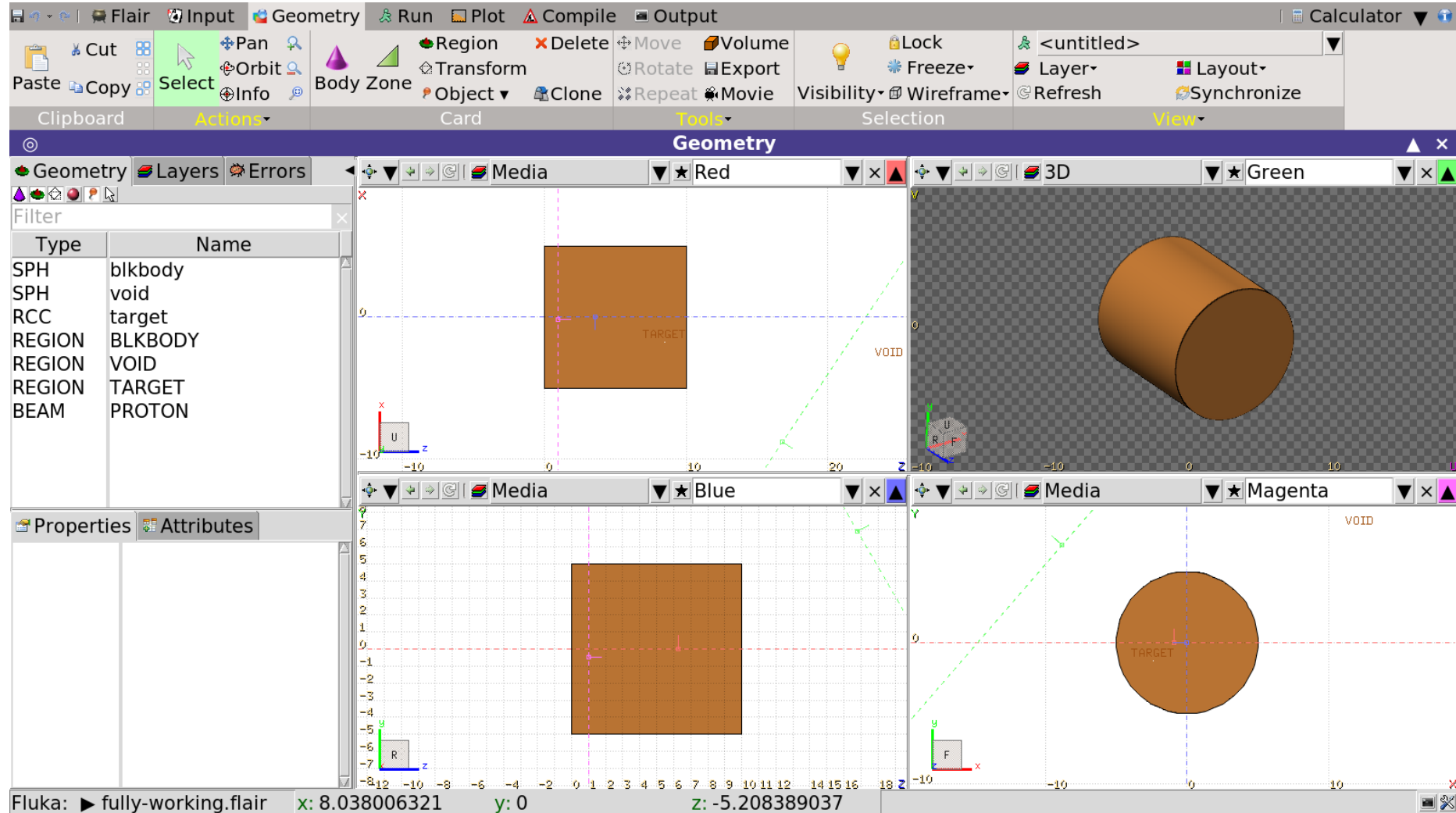


- The manual will open on the relevant page
- The manual is also available on the Fluka web page [www.fluka.cern](http://www.fluka.cern)



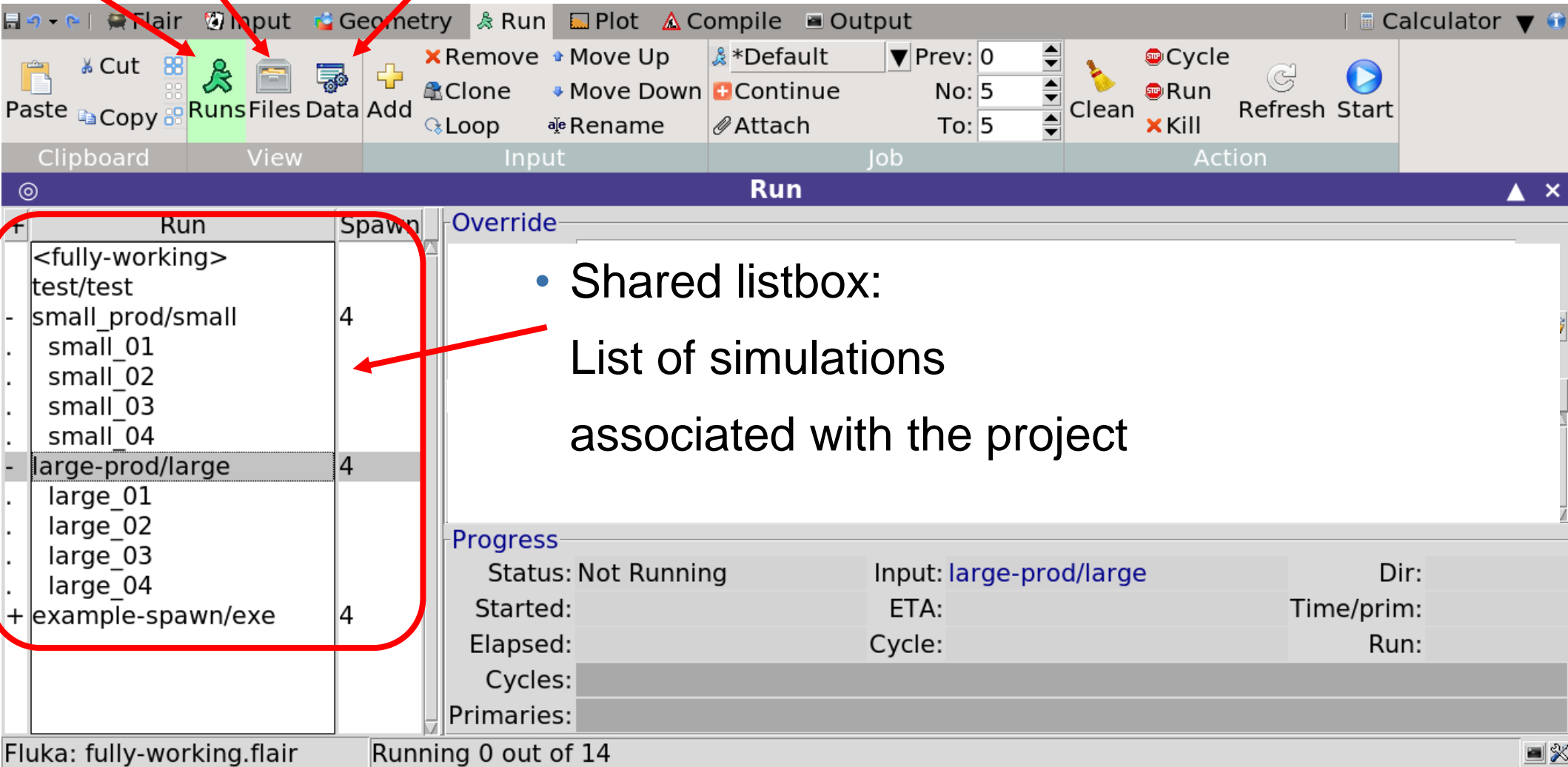
# Geometry tab

- Visualize and edit geometry
- Plot results
- Dedicated lectures



# Run tab

- 3 views: “Runs”, “Files”, and “Data”



Clipboard View Input Job Action

Run	Spawn	Override
<fully-working>		
test/test		
- small_prod/small	4	
. small_01		
. small_02		
. small_03		
. small_04		
- large-prod/large	4	
. large_01		
. large_02		
. large_03		
. large_04		
+ example-spawn/exe	4	

Progress

Status: Not Running	Input: large-prod/large	Dir:
Started:	ETA:	Time/prim:
Elapsed:	Cycle:	Run:
Cycles:		
Primaries:		

Fluka: fully-working.flair Running 0 out of 14

- Shared listbox:  
List of simulations  
associated with the project

# Run tab – Runs view – 1

- Management of the various simulations
- Basic inputfile of the Flair project
- Different simulations associated with the Flair project

The screenshot displays the Flair software interface, specifically the Run tab and the Runs view. The interface includes a menu bar with options like Flair, Input, Geometry, Run, Plot, and Compi. A context menu is open over the 'Runs' view, showing options such as Add, Remove, Move Up, Move Down, Clone, Loop, and Rename. The 'Runs' view is a table with columns for Run, Spawn, and Override. The 'Run' column contains entries like '<fully-working>', 'test/test', 'small\_prod/small', 'small\_01', 'small\_02', 'small\_03', 'small\_04', 'large-prod/large', 'large\_01', 'large\_02', 'large\_03', 'large\_04', and 'example-spawn/exe'. The 'Spawn' column shows values of 4 for several entries. The 'Override' column shows 'pointless\_define\_1' and 'pointless\_define\_2' with checkboxes. The status bar at the bottom indicates 'Fluka: fully-working.flair' and 'Running 0 out of 14'.

Run	Spawn	Override
<fully-working>		
test/test		
- small_prod/small	4	
. small_01		
. small_02		
. small_03		
. small_04		
- large-prod/large	4	
. large_01		
. large_02		
. large_03		
. large_04		
+ example-spawn/exe	4	

Fluka: fully-working.flair      Running 0 out of 14

# Run tab – Runs view – 2

- Override of inputs

- Number of primaries

- Executable

- #define

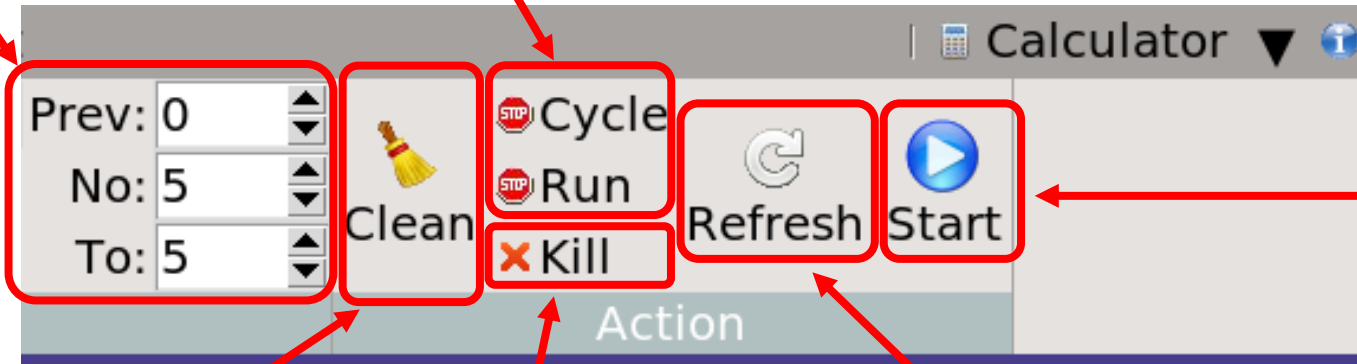
Dedicated lecture

Defines:		Default Defines
	Name	Value
<input type="checkbox"/>	pointless_define_1	
<input checked="" type="checkbox"/>	pointless_define_2	10

# Run tab – Runs view – 3

- Cycles control: how many cycles to run, starting from which cycle

- Cleanly stop the cycles/runs currently running



- Start a simulation

- Remove files from a previous simulation

- Refresh the progress field

- Kill the current simulations

# Run tab – Runs view – 4

The screenshot shows a window titled "Run" with a tree view on the left and a detailed view on the right. The tree view lists runs under categories: <fully-working>, test/test, small\_prod/small, large-prod/large, and example-spawn/exe. The detailed view for the selected run "small\_01" shows the following information:

Status: <b>Running</b>	Input: <b>small_prod/small_01</b>	Dir: <b>fluka_29267</b>
Started: <b>2020.08.28 17:16:55</b>	ETA: <b>2020.08.28 17:17:19</b>	Time/prim: <b>5.66 ms</b>
Elapsed: <b>906 ms</b>	Cycle: <b>4.76 s</b>	Run: <b>22.7 s</b>
Cycles:	Current: <b>2 [5]</b>	Completed: <b>20%</b>
Primaries:	Current: <b>161 [1000]</b>	Completed: <b>16%</b>

• Inputfile actually running

• Temporary directory

• Status

• Estimated end of the simulation

• Time per primary

• Overall job progress bar

• Cycle progress bar

• Time since the start of the cycle

• Time until the end of the cycle

• Time until the end of the simulation

# Run tab – Runs view – 5

- At the end of the simulations...

The screenshot shows a software interface with a 'Run' tab. On the left, there is a tree view of runs. The 'Run' column lists various simulation runs, and the 'Spawn' column shows the number of spawned processes. A red arrow points from the 'Status' column header to the 'Status' field in the detailed view.

Run	Spawn
<fully-working> test/test	
- small_prod/small	4
small_01	
small_02	
small_03	
small_04	
- large-prod/large	4
large_01	
large_02	
large_03	
large_04	
+ example-spawn/exe	4

• **Status**

**Override**

- **WARNING:**  
“Finished OK” means OK from the computing point of view, there is no guarantee that the output of the simulation is physically meaningful!

**Progress**

Status: Finished OK	Input: small_prod/small_01	Dir:
Started:	ETA:	Time/prim:
Elapsed:	Cycle:	Run:
Cycles:		
Primaries:		

# After running – 1

- Content of the working directory

```
fluka_user:/home/fluka_user$  
fluka_user:/home/fluka_user$ ls  
fully-working.flair  fully-working.inp  my.exe  small_prod  tutorial.flair  
fluka_user:/home/fluka_user$  
fluka_user:/home/fluka_user$ .
```

- Content of the working sub-directory

```
fluka_user:/home/fluka_user$ cd small_prod/  
fluka_user:/home/fluka_user/small_prod$ ls  
ransmall_01001  ransmall_04005      small_01004_fort.21  small_02003_fort.21  small_03002_fort.21  small_04001_fort.21  
ransmall_01002  ransmall_04006      small_01004_fort.22  small_02003_fort.22  small_03002_fort.22  small_04001_fort.22  
ransmall_01003  small_01.inp        small_01005.err      small_02004.err      small_03003.err      small_04002.err  
ransmall_01004  small_01.out        small_01005.log      small_02004.log      small_03003.log      small_04002.log  
ransmall_01005  small_01001.err     small_01005.out      small_02004.out      small_03003.out      small_04002.out  
ransmall_01006  small_01001.log     small_01005_fort.21  small_02004_fort.21  small_03003_fort.21  small_04002_fort.21  
ransmall_02001  small_01001.out     small_01005_fort.22  small_02004_fort.22  small_03003_fort.22  small_04002_fort.22  
ransmall_02002  small_01001_fort.21  small_02.inp        small_02005.err      small_03004.err      small_04003.err  
ransmall_02003  small_01001_fort.22  small_02.out        small_02005.log      small_03004.log      small_04003.log  
ransmall_02004  small_01002.err     small_02001.err      small_02005.out      small_03004.out      small_04003.out  
ransmall_02005  small_01002.log     small_02001.log      small_02005_fort.21  small_03004_fort.21  small_04003_fort.21  
ransmall_02006  small_01002.out     small_02001.out      small_02005_fort.22  small_03004_fort.22  small_04003_fort.22  
ransmall_03001  small_01002_fort.21  small_02001_fort.21  small_03.inp        small_03005.err      small_04004.err  
ransmall_03002  small_01002_fort.22  small_02001_fort.22  small_03.out        small_03005.log      small_04004.log  
ransmall_03003  small_01003.err     small_02002.err      small_03001.err      small_03005.out      small_04004.out  
ransmall_03004  small_01003.log     small_02002.log      small_03001.log      small_03005_fort.21  small_04004_fort.21  
ransmall_03005  small_01003.out     small_02002.out      small_03001.out      small_03005_fort.22  small_04004_fort.22  
ransmall_03006  small_01003_fort.21  small_02002_fort.21  small_03001_fort.21  small_04.inp        small_04005.err  
ransmall_04001  small_01003_fort.22  small_02002_fort.22  small_03001_fort.22  small_04.out        small_04005.log  
ransmall_04002  small_01004.err     small_02003.err      small_03002.err      small_04001.err      small_04005.out  
ransmall_04003  small_01004.log     small_02003.log      small_03002.log      small_04001.log      small_04005_fort.21  
ransmall_04004  small_01004.out     small_02003.out      small_03002.out      small_04001.out      small_04005_fort.22  
fluka_user:/home/fluka_user/small_prod$ .
```



# After running – 2

- Content of the working directory

```
fluka_user:/home/fluka_user$  
fluka_user:/home/fluka_user$ ls  
fully-working.flair  fully-working.inp  my.exe  small_prod  tutorial.flair  
fluka_user:/home/fluka_user$  
fluka_user:/home/fluka_user$ .
```

- Content of the working sub-directory

```
fluka_user:/home/fluka_user$ cd small_prod/  
fluka_user:/home/fluka_user/small_prod$ ls  
ransmall_01001  ransmall_04005  small_01004_fort.21  small_02003_fort.21  small_03002_fort.21  small_04001_fort.21  
ransmall_01002  ransmall_04006  small_01004_fort.22  small_02003_fort.22  small_03002_fort.22  small_04001_fort.22  
ransmall_01003  small_01.inp  small_01005.err  small_02004.  err  
ransmall_01004  small_01.out  small_01005.log  small_02004.  log  
ransmall_01005  small_01001.err  small_01005.out  small_02004.  out  
ransmall_01006  small_01001.log  small_01005_fort.21  small_02004.  fort.21  
ransmall_02001  small_01001.out  small_01005_fort.22  small_02004.  fort.22  
ransmall_02002  small_01001_fort.21  small_02005.err  small_03004.err  small_04003.err  
ransmall_02003  small_01001_fort.22  small_02005.log  small_03004.log  small_04003.log  
ransmall_02004  small_01002.err  small_02001.err  small_02005.out  small_03004.out  small_04003.out  
ransmall_02005  small_01002.log  small_02001.log  small_02005_fort.21  small_03004_fort.21  small_04003_fort.21  
ransmall_02006  small_01002.out  small_02001.out  small_02005_fort.22  small_03004_fort.22  small_04003_fort.22  
ransmall_03001  small_01002_fort.21  small_02001_fort.21  small_03005.err  small_04004.err  
ransmall_03002  small_01002_fort.22  small_02001_fort.22  small_03005.log  small_04004.log  
ransmall_03003  small_01003.err  small_02002.err  small_03001.err  small_03005.out  small_04004.out  
ransmall_03004  small_01003.log  small_02002.log  small_03001.log  small_03005_fort.21  small_04004_fort.21  
ransmall_03005  small_01003.out  small_02002.out  small_03001.out  small_03005_fort.22  small_04004_fort.22  
ransmall_03006  small_01003_fort.21  small_02002_fort.21  small_03001_fort.21  small_04005.err  
ransmall_04001  small_01003_fort.22  small_02002_fort.22  small_03001_fort.22  small_04005.log  
ransmall_04002  small_01004.err  small_02003.err  small_03002.err  small_04001.err  small_04005.out  
ransmall_04003  small_01004.log  small_02003.log  small_03002.log  small_04001.log  small_04005_fort.21  
ransmall_04004  small_01004.out  small_02003.out  small_03002.out  small_04001.out  small_04005_fort.22  
fluka_user:/home/fluka_user/small_prod$ .
```

.inp and .out files specific of each spawn

small\_01.inp  
small\_01.out

small\_02.inp  
small\_02.out

small\_03.inp  
small\_03.out

small\_04.inp  
small\_04.out

# Run tab – Files view – 1

- Generated files accessible via the Files view

Run	Spawn	Cycles	File	Type	Size	Date
<fully-working>		001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
test/test		002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
- small_prod/small	4	003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
. small_01		004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
. small_02		005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
. small_03		006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
. small_04		compile				
- large-prod/large	4	data				
. large_01		input				
. large_02		plot				
. large_03		temporary				
. large_04						
+ example-spawn/exe	4					

Fluka: fully-working.flair | Files: 6 Total Size: 15104969

# Run tab – Files view – 2

- File per each cycle:
  - one (1) .out file
  - one (1) .log file
  - one (1) .err file
  - one (1) random seed file
  - one (1) scoring file per each logical unit scoring used

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				

# Run tab – Files view – 3

- Naming convention for file names; the filename contains:
  - the name of the run, e.g.: `small`
  - The spawn identifier, e.g.: `01`
  - The cycle identifier, e.g.: `001`
  - The file type identifier, e.g.: `.err` , `fort.21` , `ran`

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				

# Run tab – Files view – 4

- Naming convention for file names; the filename contains:
  - the name of the run, e.g.: `small`
  - The spawn identifier, e.g.: `01`
  - The cycle identifier, e.g.: `001`
  - The file type identifier, e.g.: `.err` , `fort.21` , `ran`
- In this example 6 files were generated:

`small_01001.err`

`small_01001.log`

`small_01001.out`

`ransmall_01001`

`small_01001_fort.21`

`small_01001_fort.22`

# Run tab – Files view – 4

- Naming convention for file names; the filename contains:
  - the name of the run, e.g.: `small`
  - The spawn identifier, e.g.: `01`
  - The cycle identifier, e.g.: `001`
  - The file type identifier, e.g.: `.err` , `fort.21` , `ran`

- In this example 6 files were generated:

`small_01001.err`

`small_01001_fort.21`

`small_01001.log`

`small_01001_fort.22`

`small_01001.out`

`ransmall_01001`

← renaming is planned

# Run tab – Files view – 5

- Spawn 1 Cycle 1

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				

- Spawn 1 Cycle 5

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01005	-file-	1651	2020.08.28 17:17
002	small_prod/small_01005.err	Error	22470	2020.08.28 17:17
003	small_prod/small_01005.log	Log	0	2020.08.28 17:17
004	small_prod/small_01005.out	Output	104249	2020.08.28 17:17
005	small_prod/small_01005_fort.21	21	7488238	2020.08.28 17:17
006	small_prod/small_01005_fort.22	22	7488238	2020.08.28 17:17
compile				
data				
input				
plot				
temporary				

# Run tab – Files view – 6

- Spawn 1 Cycle 1

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				

- Spawn 2 Cycle 1

Cycles	File	Type	Size	Date
001	small_prod/ransmall_02001	-file-	1651	2020.08.28 17:16
002	small_prod/small_02001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_02001.log	Log	0	2020.08.28 17:16
004	small_prod/small_02001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_02001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_02001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				



# Run tab – Files view – 7

- Spawn 1 Cycle 1

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01001	-file-	1651	2020.08.28 17:16
002	small_prod/small_01001.err	Error	22470	2020.08.28 17:16
003	small_prod/small_01001.log	Log	0	2020.08.28 17:16
004	small_prod/small_01001.out	Output	104372	2020.08.28 17:16
005	small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16
006	small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16
compile				
data				
input				
plot				
temporary				

- Spawn 1 Cycle 6

Cycles	File	Type	Size	Date
001	small_prod/ransmall_01006	-file-	1651	2020.08.28 17:17
002				
003				
004				
005				
006				
compile				
data				
input				
plot				
temporary				

- Random file for the next cycle is generated

# Run tab – Data view – 1

- All the generated files need to be merged to be analyzed

```
small_01001_fort.21  
small_01002_fort.21  
small_01003_fort.21  
small_01004_fort.21`  
small_01005_fort.21
```

```
small_02001_fort.21  
small_02002_fort.21  
small_02003_fort.21  
small_02004_fort.21  
small_02005_fort.21
```

```
small_03001_fort.21  
small_03002_fort.21  
small_03003_fort.21  
small_03004_fort.21  
small_03005_fort.21
```

```
small_04001_fort.21  
small_04002_fort.21  
small_04003_fort.21  
small_04004_fort.21  
small_04005_fort.21
```

# Run tab – Data view – 2

- Flair automatically identifies the logical units used from the inputfile

The screenshot shows the Flair software interface. The top menu bar includes options like Flair, Input, Geometry, Run, Plot, Compile, and Output. Below the menu is a toolbar with icons for Cut, Copy, Paste, New, Add, Remove, Filter, Refresh, Scan, and Clean Process. The main window is titled 'Run' and contains a tree view on the left and a data table on the right.

**Tree View:**

- + Run
  - <fully-working>
    - test/test
    - small\_prod/small (4)
      - . small\_01
      - . small\_02
      - . small\_03
      - . small\_04
    - large-prod/large (4)
      - . large\_01
      - . large\_02
      - . large\_03
      - . large\_04
    - + example-spawn/exe (4)

**Detectors Table:**

Run	Type	Output	Name/Unit
small_prod/small	usrbin	small_prod/small_21.bnn	21
small_prod/small	usrbin	small_prod/small_22.bnn	22

**Files Table:**

File	Type	Size	Date

Fluka: fully-working.flair | Files: 40

# Run tab – Data view – 3

- Flair finds all the corresponding file (per spawn and per cycle)

The screenshot shows the Flair software interface. At the top, there is a menu bar with options: Flair, Input, Geometry, Run, Plot, Compile, Output, and Calculator. Below the menu bar is a toolbar with various icons and labels: Paste, Copy, RunsFiles, Data, Scan, Remove, Refresh, Add, Remove, Filter, Clean Process, and Action. The main window is titled 'Run' and contains a tree view on the left and a data table on the right.

The tree view on the left shows a hierarchy of runs and spawns:

- + Run
- <fully-working>
- test/test
- small\_prod/small (4 spawns)
  - . small\_01
  - . small\_02
  - . small\_03
  - . small\_04
- large-prod/large (4 spawns)
  - . large\_01
  - . large\_02
  - . large\_03
  - . large\_04
- + example-spawn/exe (4 spawns)

The data table on the right is titled 'Detectors' and has columns: Run, Type, Output, and Name/Unit. It shows data for two runs: 'small\_prod/small' (Type: usrbin) and 'small\_prod/small' (Type: usrbin). The output files are 'small\_prod/small\_21.bnn' and 'small\_prod/small\_22.bnn' with Name/Unit values 21 and 22 respectively.

Below the 'Detectors' table, there is a 'Files' tab and a 'Parameters' tab. The 'Files' tab shows a table with columns: File, Type, Size, and Date. It lists 8 files with their respective types, sizes, and dates.

File	Type	Size	Date
small_prod/small_01001_fort.21	21	7488238	2020.08.28 17:16:54
small_prod/small_01001_fort.22	22	7488238	2020.08.28 17:16:54
small_prod/small_01002_fort.21	21	7488238	2020.08.28 17:17:02
small_prod/small_01002_fort.22	22	7488238	2020.08.28 17:17:02
small_prod/small_01003_fort.21	21	7488238	2020.08.28 17:17:10
small_prod/small_01003_fort.22	22	7488238	2020.08.28 17:17:10
small_prod/small_01004_fort.21	21	7488238	2020.08.28 17:17:18
small_prod/small_01004_fort.22	22	7488238	2020.08.28 17:17:18

At the bottom of the window, the status bar shows 'Fluka: fully-working.flair' and 'Files: 40'.

# Run tab – Data view – 4

- Process can be forced by hand:

- 1-Select the run
- 2-Refresh
- 3-Scan
- 4-Process (merge)

- Processed binary results files are generated (specific extensions: **.bnn**, **.bnx**, **.rnc**, etc. more in other lectures)

The screenshot shows the Flair software interface with the Run tab selected. The interface is divided into several sections:

- Toolbar:** Contains buttons for Cut, Copy, Paste, Runs, Files, Data, Scan (3), Add, Remove, Filter, Refresh (2), Clean, and Process (4).
- Run List:** A table with columns 'Run' and 'Spawn'. The row 'small\_prod/small' is selected and highlighted with a red box (1).
- Detectors Table:** A table with columns 'Run', 'Type', and 'Output'. It lists detectors for 'small\_prod/small'.
- Files Table:** A table with columns 'File', 'Type', and 'Size'. It lists files generated for the selected run.

Run	Spawn
<fully-working>	
test/test	
- small_prod/small	4
small_01	
small_02	
small_03	
small_04	
- large-prod/large	4
large_01	
large_02	
large_03	
large_04	
+ example-spawn/exe	4

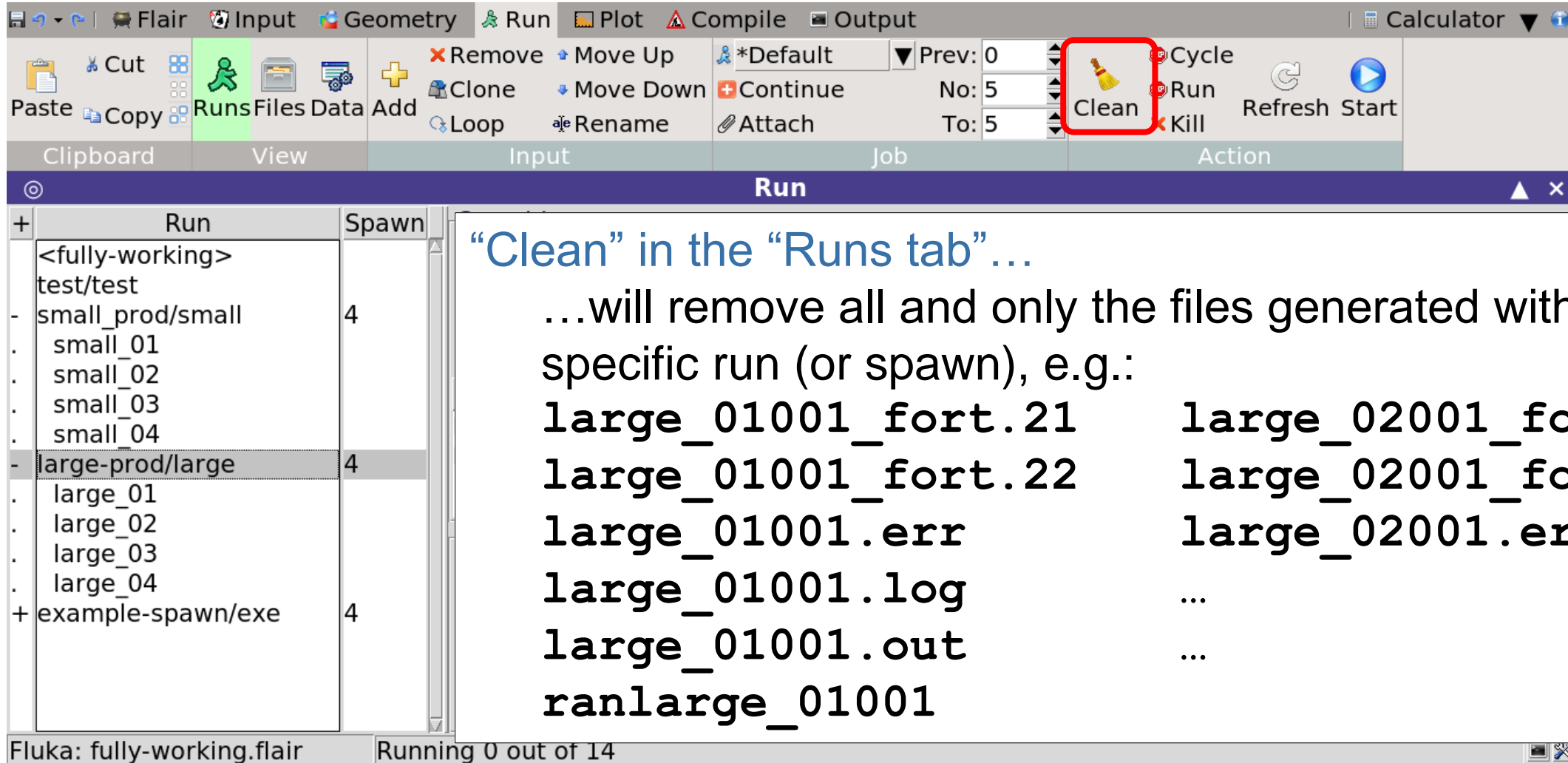
Run	Type	Output
small_prod/small	usrbin	small_prod/small_21.bnn
small_prod/small	usrbin	small_prod/small_22.bnn

File	Type	Size
small_prod/small_01001_fort.21	21	7488238
small_prod/small_01001_fort.22	22	7488238
small_prod/small_01002_fort.21	21	7488238
small_prod/small_01002_fort.22	22	7488238
small_prod/small_01003_fort.21	21	7488238
small_prod/small_01003_fort.22	22	7488238
small_prod/small_01004_fort.21	21	7488238
small_prod/small_01004_fort.22	22	7488238

Fluka: fully-working.flair Files: 40

# Run tab – Cleaning – 1

- Removing files generated for the cycles and merged files are different actions!



The screenshot shows the FLUKA software interface with the 'Run' tab selected. The 'Clean' button, represented by a bell icon, is highlighted with a red rectangle. Below the toolbar, a table lists the contents of the 'Run' tab:

Run	Spawn
<fully-working>	
test/test	
- small_prod/small	4
. small_01	
. small_02	
. small_03	
. small_04	
- large-prod/large	4
. large_01	
. large_02	
. large_03	
. large_04	
+ example-spawn/exe	4

A text box titled "Clean" in the "Runs tab" ... explains that this action will remove all and only the files generated within a specific run (or spawn), e.g.:

```
large_01001_fort.21      large_02001_fort.21
large_01001_fort.22      large_02001_fort.22
large_01001.err          large_02001.err
large_01001.log          ...
large_01001.out          ...
ranlarge_01001
```

# Run tab – Cleaning – 2

- Removing files generated for the cycles and merged files are different actions!

The screenshot shows the Flair software interface. The top menu bar includes 'Flair', 'Input', 'Geometry', 'Run', 'Plot', 'Compile', and 'Output'. Below the menu bar is a toolbar with various icons. The 'Data' tab is selected, and the 'Clean' button (represented by a broom icon) is highlighted with a red box. The 'Run' window is open, showing a list of runs and a table of detectors. The 'Data' sub-tab is active, displaying a table of detectors with columns for 'Run', 'Type', 'Output', and 'Name/Unit'. A text box is overlaid on the interface, explaining the 'Clean' action.

Run	Spawn	Detectors
<fully-working>		
test/test		
- small_prod/small	4	
small_01		
small_02		
small_03		
small_04		
- large-prod/large	4	
large_01		
large_02		
large_03		
large_04		
+ example-spawn/exe	4	

Run	Type	Output	Name/Unit
small_prod/small	usrbin	small_prod/small_21.bnn	21
small_prod/small	usrbin	small_prod/small_22.bnn	22

“Clean” in the “Data tab”...  
...will remove only the merged results, e.g.:  
**small\_21.bnn**  
**small\_22.bnn**

Fluka: fully-working.flair    Files: 40

# Compile tab

- Only very basic information are given here

- Routine to be compiled

- Add routines

- Select linker

- Build executable

File	Type	Size	Date
source.f	Fortran	9203	2020.08.31 15:16:21
mgdraw.f	Fortran	14783	2020.08.31 15:16:23
own_routine.cc	C++	24064	2020.08.31 15:17:07

- User routine are discussed in advanced courses



# Do you remember slide 6?

The screenshot shows the Flair software interface. The top menu bar includes: Flair, Input, Geometry, Run, Plot, Compile, Output, Calculator. Below the menu bar are several toolbars: Clipboard (Paste, Copy), Project (New, Open, Save), Edit (B, I, U), Publish (Document, Print, Refresh), Tools (Config, Report, Updates, About), and Close (Exit). The main window has a title bar 'Flair' and a notes area. Red arrows point from the following menu items to text in the notes area: 'Input' to 'Build input and geometry', 'Geometry' to 'Build geometry and plot results', 'Run' to 'Run and merge results', 'Plot' to 'Plot results', 'Compile' to 'Compile own executable', and 'Output' to 'Visualize output files and messages'. The status bar at the bottom shows 'Fluka: Dir: /home/fluka\_user Exe:'.

Build input and geometry

Build geometry and plot results

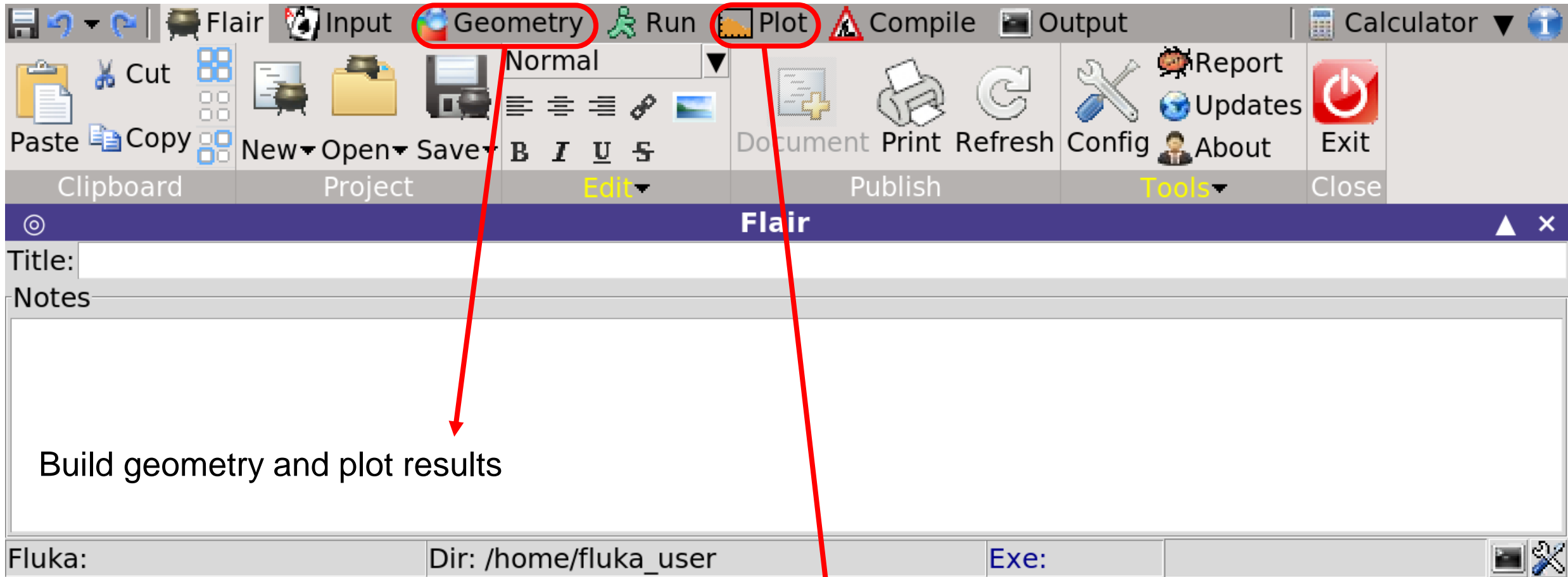
Run and merge results

Plot results

Compile own executable

Visualize output files and messages

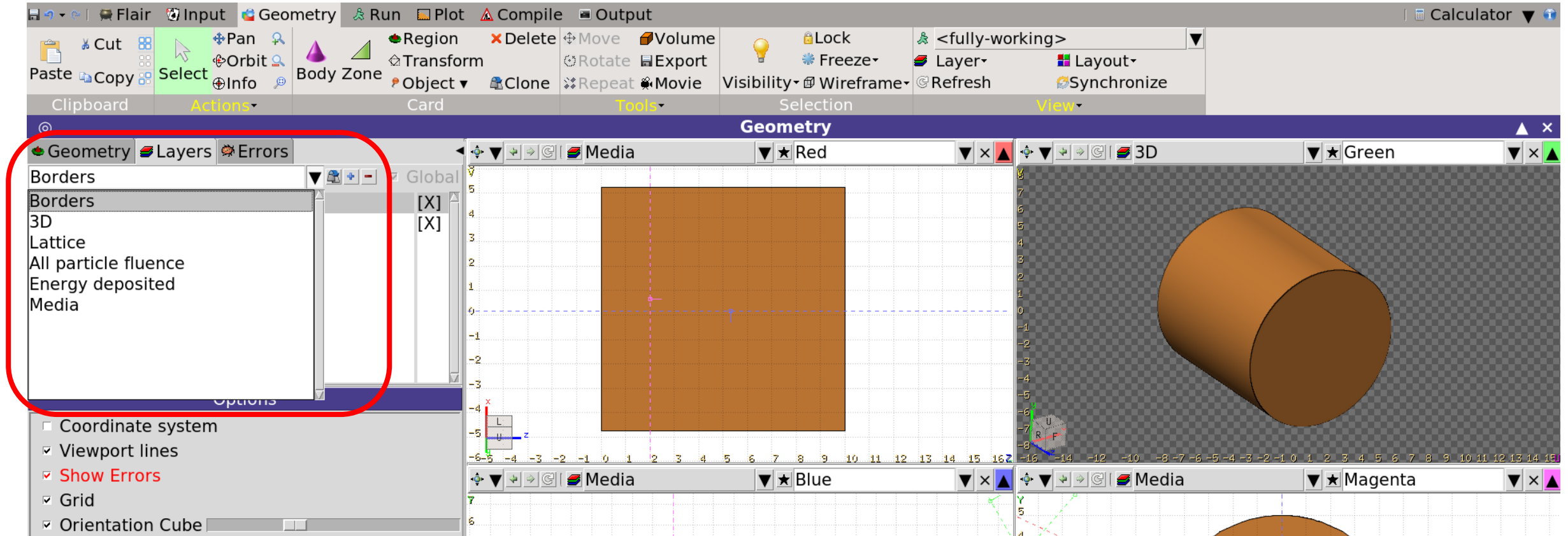
# Do you remember slide 6?



Plot results

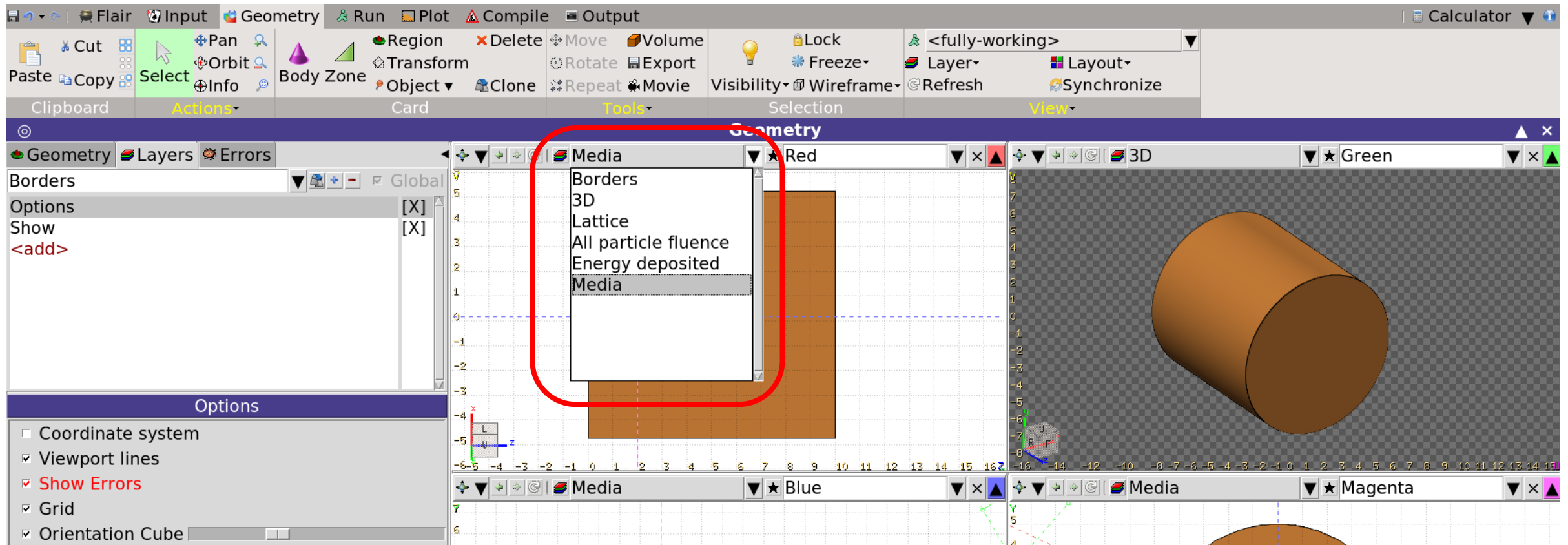
# Plotting result in the Geometry tab – 1

- This is just a mention of this feature, details and how-to in the dedicated lecture
- Possible to plot USRBIN scoring, i.e. `.bnn` files (again, see dedicated lecture)



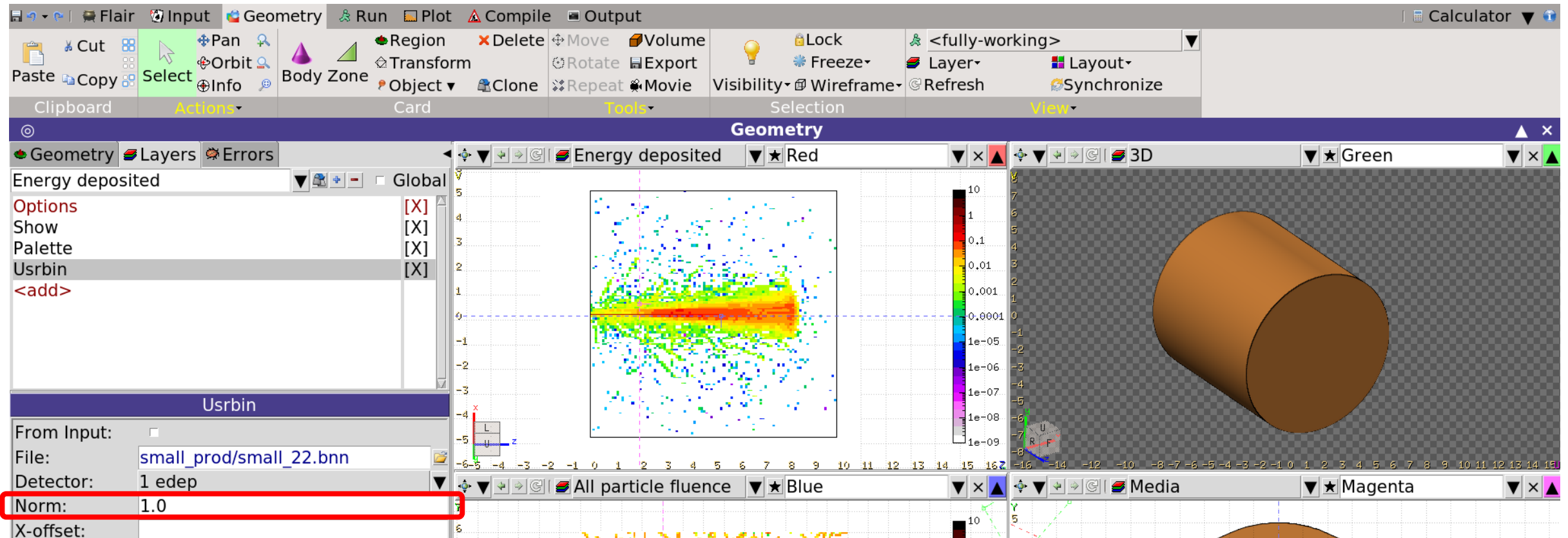
# Plotting result in the Geometry tab – 2

- This is just a mention of this feature, details and how-to in the dedicated lecture
- Possible to plot USRBIN scoring, i.e. `.bnn` files (again, see dedicated lecture)



# Plotting result in the Geometry tab – 3

- This is just a mention of this feature, details and how-to in the dedicated lecture
- Possible to plot USRBIN scoring, i.e. `.bnn` files (again, see dedicated lecture)



# Plotting result in the Geometry tab – 4

- WARNING: if the USRBIN used in a layer is missing, an error message is issued
- Not necessarily something to be worried about
- This will happen in the hands on that follows this lecture! Don't worry!

The screenshot shows the Fluka software interface with the 'Input' tab selected. The main window displays the input file content, which includes the following sections:

```
TITLE course basic input hands on
Set the defaults for precision simulations
DEFAULTS : PRECISIO
Define the beam characteristics
*BEAM
  Beam: Momentum      p: 0.8      Part: PROTON
  Δp: Flat            Δφ: Flat
  Shape(X): Rectangular Δx:          Shape(Y): Rectangular Δy:
Define the beam position
*BEAMPOS
  x: 0.0      y: 0.0      z: -1.0
  cosx:       cosy:       Type: POSITIVE
  Accuracy:   Option:     Paren:
  Geometry:   Out:        Fmt: COMBNAME
*GEOBEGIN
  Title:
  Black body
  *SPH blkbody      x: 0.0      y: 0.0      z: 0.0
                   R: 100000.0
  Void sphere
  *SPH void         x: 0.0      y: 0.0      z: 0.0
                   R: 10000.0
  Cylindrical target
  *RCC target      x: 0.0      y: 0.0      z: 0.0
                   Hx: 0.0      Hy: 0.0      Hz: 10.0
                   R: 5.0
*END
  Black hole
  *REGION BLKBODY      Neigh: 5
  expr: +blkbody -void
```

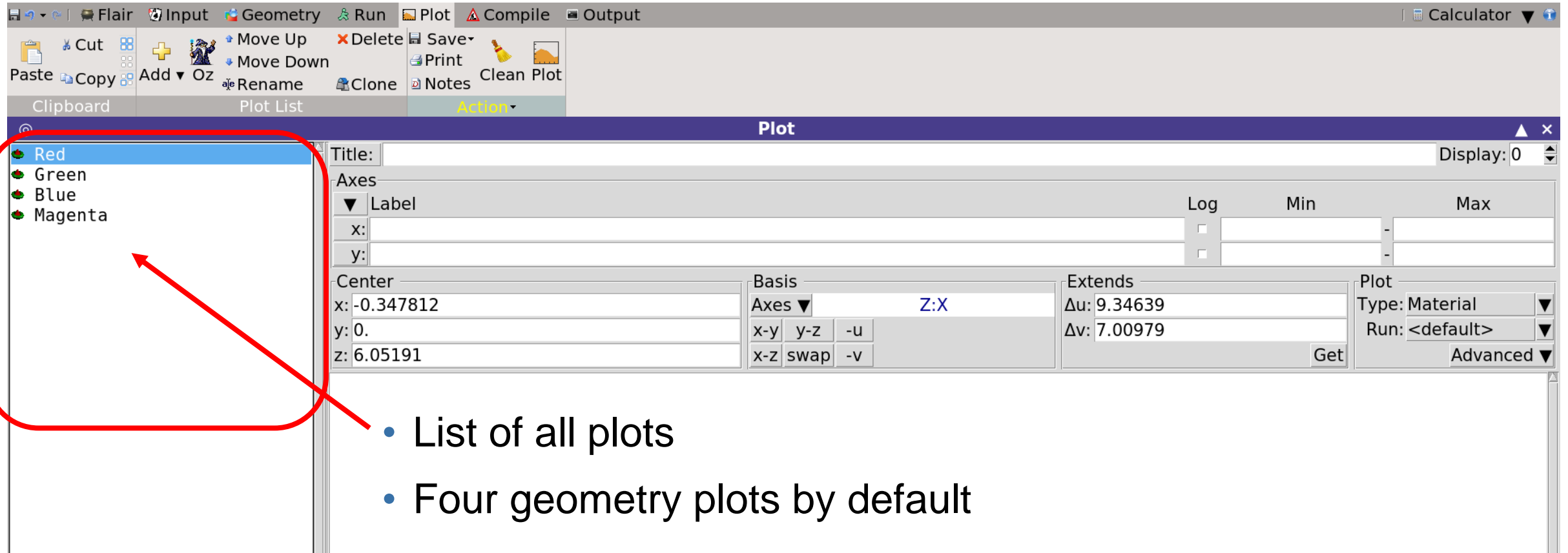
An error message is displayed in the bottom right corner of the window:

```
Error loading USRBIN
Unable to load usrbn
detector 'test-dir/test_22.bnn'
1
```

The status bar at the bottom of the window shows: 'Fluka: course\_basic\_handson; Current:1 Selected:1 Total:23'

# Plotting result in the Plot tab – 1

- This is just a mention of this feature, details and how-to in the dedicated lecture
- Possible for geometry and all built-in scorings

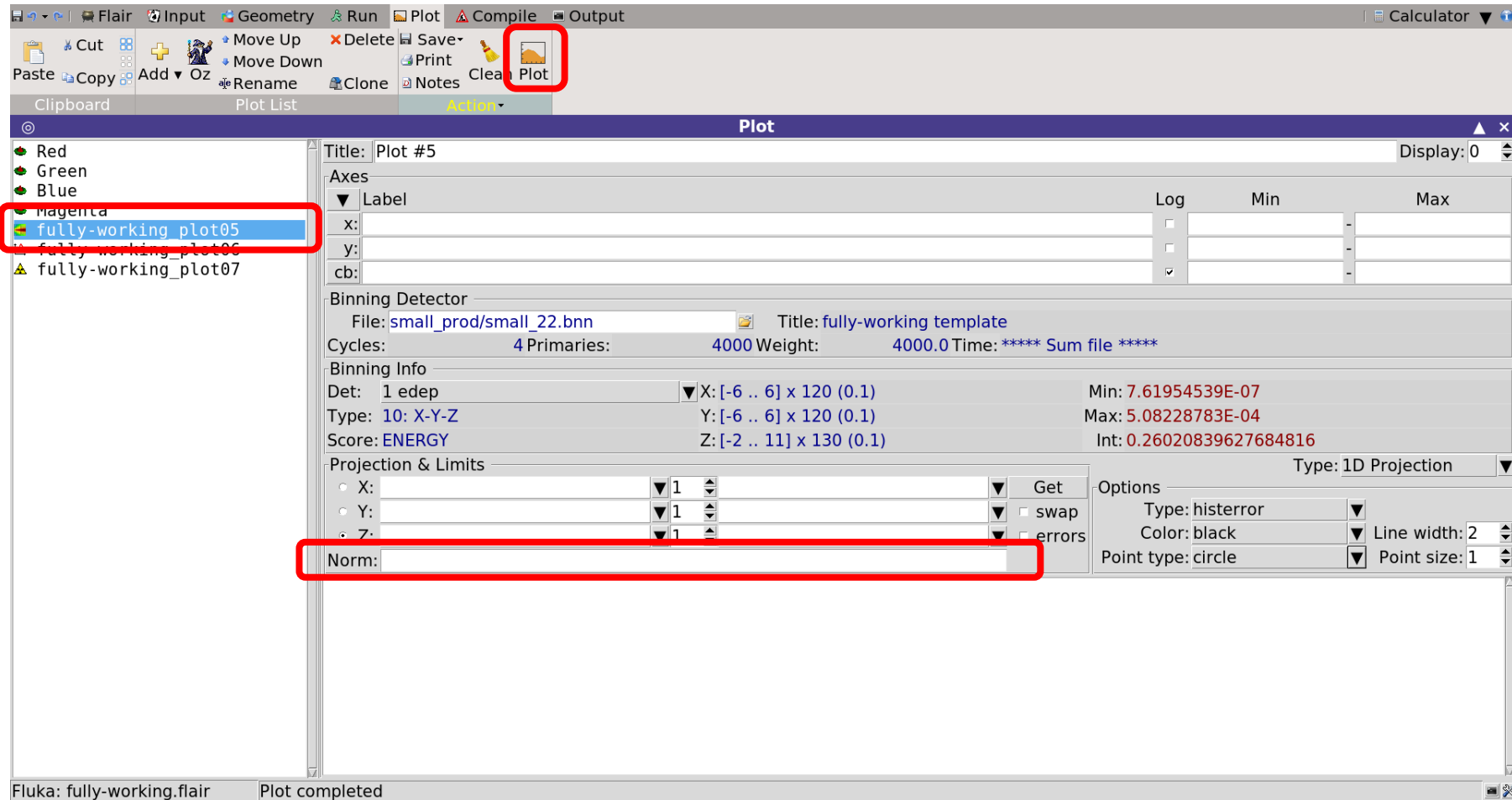


The screenshot shows the 'Plot' tab in a software interface. The 'Plot List' on the left contains four entries: Red, Green, Blue, and Magenta. A red box highlights this list, and a red arrow points from it to the text 'List of all plots' in the bullet points below. The main plot area shows settings for a plot, including axes labels, center coordinates, basis, and extends values.

- List of all plots
- Four geometry plots by default

# Plotting result in the Plot tab – 2

- This is just a mention of this feature, details and how-to in the dedicated lecture
- Possible for geometry and all built-in scorings





# Summary of the work flow

- Create your **input** in the Input tab and Geometry tab (see future lectures)
- Verify your geometry in the Geometry tab
- **Run** the simulations and **merge** the output files in the Run tab
- **Plot** your results in the Plot tab and Geometry tab (see future lectures)

## Time to do some practice!

- Let's start from the example file  
and run a simulation step by step



