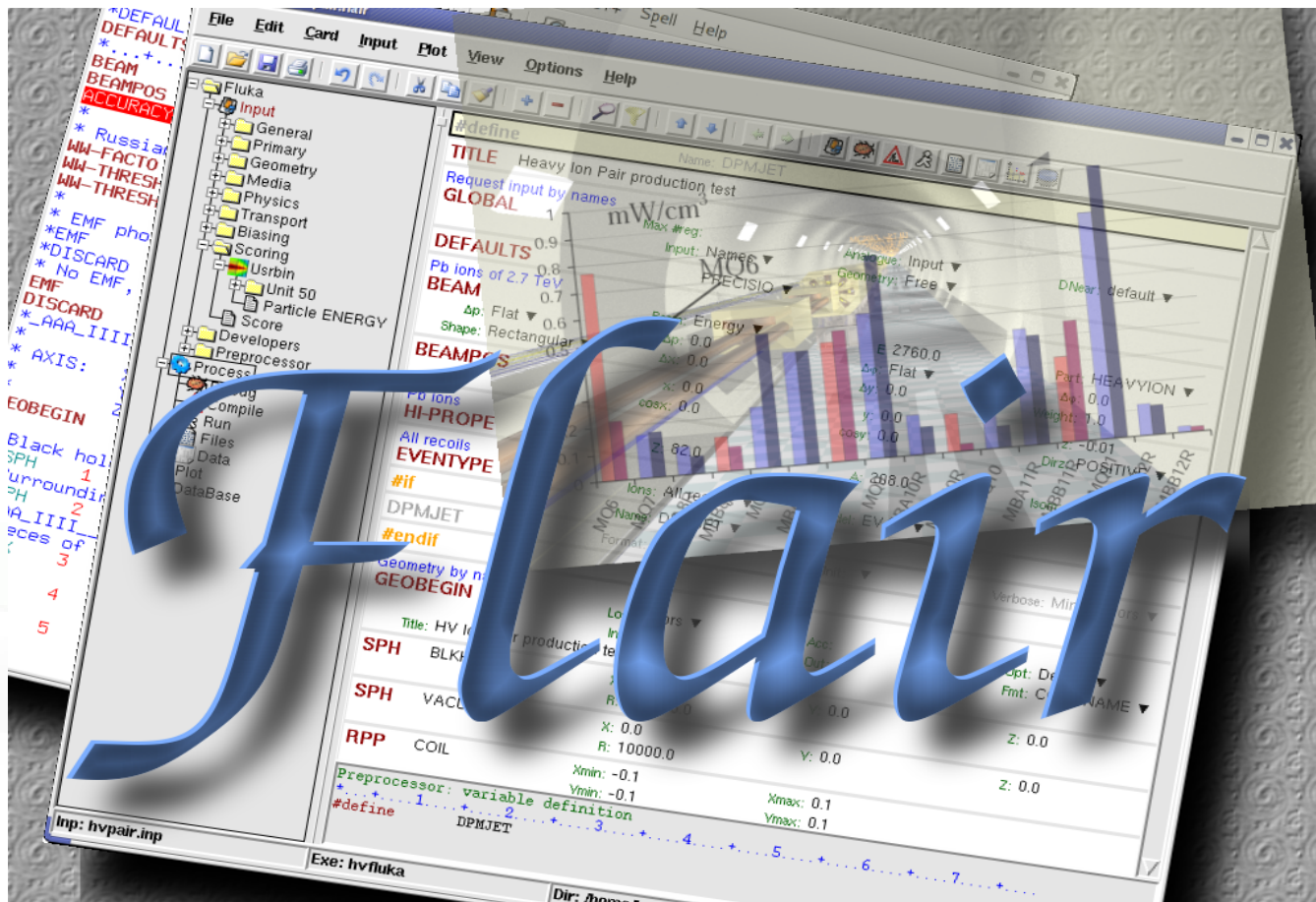




# Introduction to Flair

21<sup>st</sup> FLUKA Beginner's Course  
ALBA Synchrotron (Spain)  
April 8-12, 2019

# About



/fleə(r)/ n [U,C] natural or instinctive ability (to do something well, to select or recognize what is best, more useful, etc.  
[Oxford Advanced Dictionary of Current English]

# What is flair

## FLUKA Advanced Interface

[\[http://www.fluka.org/flair\]](http://www.fluka.org/flair)

- **All-in-one** User friendly graphical Interface
- Minimum requirements on additional software
- Working in an intermediate level

**Not hiding the inner functionality of FLUKA**

### Front-End interface:

- Fully featured **Input file Editor**
  - Mini-dialogs for each card, allows easy and almost error free editing
  - Uniform treatment of all FLUKA cards
  - Card grouping in categories and card filtering
  - Error checking and validation of the input file during editing
- **Geometry:** interactive visualization editing, transformation, and debugging
- **Compilation** of the FLUKA Executable
- **Running** and **monitoring** of the status of a/many run(s)

# What is flair

## Back-End interface:

- Inspection of the output files (core dumps and directories)
- Output file(s) viewer divided in sections
- Post processing (merging) the output data files
- Plot generation through an interface with **gnuplot** (matplotlib under development)

## Other Goodies:

- Access to FLUKA manual as hyper text
- Checking for release updates of FLUKA and flair
- Import export to various formats: MCNP/X, GDML, Povray...
- Nuclear wallet cards
- Library of materials
- Database of geometrical objects (Not yet completed)
- Programming python **API**
- Everything is accessible with keyboard shortcuts

# Concepts: Flair Project

- Store in a **single file** all relevant information:
  - Project notes
  - Links to needed files: **input file**, **source routines**, **output files** ...
  - **Multiple runs** from the same input file, as well as running status
  - Procedures on how to **run the code**
  - **Rules** on how to perform **data merging**
  - Information on how to post process and **create plots** of the results
- You can consider Flair as an **editor** for the project files.
- It can handle any FLUKA input format (reading & writing), but internally it works using the **names format** for the input, **free format with names** for the geometry
- The format is plain ASCII file with extension: **.flair**

**Note:** If you want to copy a project you need to copy **also** all linked files especially the input and source routines!

# Installation

- Flair website <http://www.fluka.org/flair> (download and documentation)
- Installation using **RPM/DEB packages** (strongly recommended!)
  - The package will create all **file associations, menu items,** and keep track of updates and files installed.  
The package will install the program to: **/usr/local/flair** and will create the following launcher programs:
    - ◆ **/usr/local/bin/flair**      flair program
    - ◆ **/usr/local/bin/fm**      FLUKA manual
    - ◆ **/usr/local/bin/pt**      Periodic Table
    - ◆ **/usr/local/bin/fless**      FLUKA output viewer

# Starting Flair

## Programs Menu (Linux)

- Click the icon of Flair from the programs menu  
Usually, Flair is in the **Science/Physics** sub-menu but this can change depending on the **Linux** distribution and window manager (look also in Applications, Education, Science, or Others sub-menus)

## Window Manager (Linux, only via RPM or DEB installation)

- Flair makes an association of the following extensions:



## Console

- Type the command **flair**  
Check that your **\$PATH** includes the directory where flair is installed

# Startup

By default flair will perform the following operations:

- Checks for the existence of a FLUKA installation (looking for the **FLUPRO** environment variable)  
if not found, the Preferences dialog box opens asking for the FLUKA path  
**WARNING:** different command shells (e.g.: bash, tcsh), as well as different windows managers (e.g.,: GNOME, KDE), look for the environment variable in different configuration files
- Opens the "Check for Updates" dialog box (every 30 days interval)



# Input Templates

- When requesting a new input or a new project flair will prompt to select an input template:

Default template: **basic.inp**

```
TITLE
GLOBAL                                1.0      1.0
DEFAULTS
BEAM
BEAMPOS
GEOBEGIN                               COMBNAME
    0 0
* Black body
SPH blkbody 0.0 0.0 0.0 1000000.0
* Void sphere
SPH void 0.0 0.0 0.0 1000000.0
* 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
RANDOMIZ 1.0
START
STOP
```

User can create their own set of input templates. They are normal FLUKA input files and they have to be placed in the `~/flair/templates` directory

# Input Editor

- With the input editor the user can manipulate the input cards:
  - Add card to input
  - Edit existing ones
  - Copy & Paste
  - Clone (Duplicate)
  - Import from other input files
  - Validate the correctness of the cards
  - Error filtering
  - Rearrange order
- If necessary, the editor will try to rearrange the input cards to create a valid FLUKA input file
  - e.g. body cards outside the **GEOBEGIN/GEOEND** parts will be moved inside

**Note:** Automatic rearranging of cards cannot work if “**#include**” cards are present. The user have to do it manually

# Card Categories

For easier access, cards are grouped in the following categories:

- **General** General purpose (TITLE, DEFAULTS, GLOBAL...)
- **Primary** Definition of the primary starting particles
- **Geometry** Cards related to the definition of the geometry bodies/regions/lattices plotting and rotations/translations
  - **Bodies** Subcategory containing only the bodies definition
  - **Transformations** Subcategory containing only the geometrical directives
- **Media** Definition and assignment of materials
- **Physics** Setting physics properties of the simulation
- **Transport** Modify the way particles are transported in FLUKA
- **Biasing** Cards for importance biasing definition
- **Scoring** Cards related to scoring
- **Flair** flair special cards
- **Preprocessor** Definitions for creating conditional input files

# Concepts: Extended Cards

- Flair is treating the input file as a **list of extended cards**;
- Each extended card contains:
  - **Comment**: All commented lines preceding the card(s) as well the inline comments
  - **Tag**: The 8 character word identifying the card. All tags not recognized by flair will be converted to **#error**
  - **WHATs**: **Multiple number of WHATs**  
(0=sdum, 1-6 first line, 7-12 continuation line...)
  - **Extra**: multi line string of extra information for special cards like **REGION, TITLE, PLOTGEOM** etc.
  - **State** (Enable/Disable)
- Flair automatically recognizes all disabled FLUKA cards  
(and separates them from the comments)

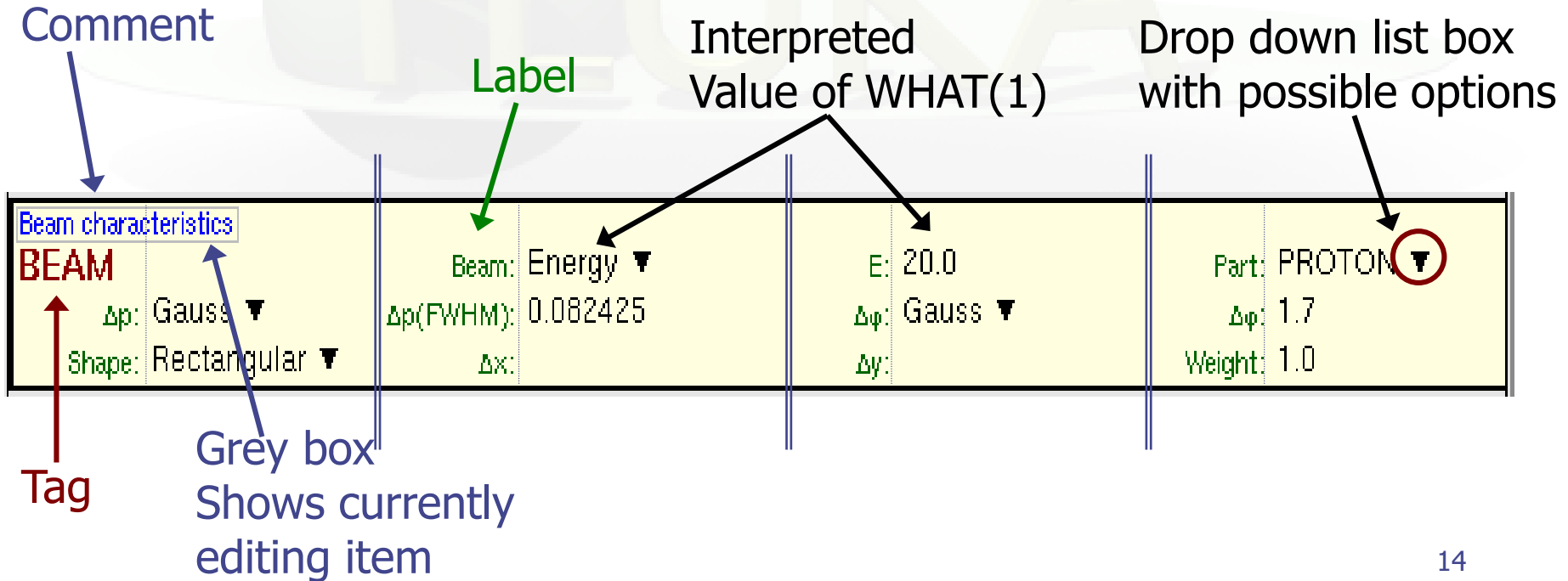
# Concepts: Extended Cards

- The region definition in the in geometry is emphasized by the presence of a card named "REGION"
- All COMPOUND cards related to one material are joined in one card
- Cards are edited with the flair editor through the use of the mini-dialogs, forcing the user to enter a *proper* information
- The user gets full control of the card using the Edit dialog  
(See button at *lower right corner*)
- Flair will try to find the best floating point representation of each number, to ensure the maximum accuracy; number of digits that fits in the specific width (10 for the fixed format, 22 for the free format)

# Anatomy of a card mini-dialog

- For each extended card flair has a mini dialog (currently in 4 columns), interpreting all information stored in the card

```
* Beam characteristics
BEAM          -20.0 -0.082425      -1.7          1.0PROTON
```



# Anatomy of a card mini-dialog

## \* Energy deposition in 3D binning

```
USRBIN      10.0    ENERGY    -50.0     45.0     54.0     36.0EneDep
USRBIN      -45.0    -54.0     -33.0    100.0    100.0    100.0&
```

### USRBIN

Type: X-Y-Z ▼

Part: ENERGY ▼

Xmin: -45.0

Ymin: -54.0

Zmin: -33.0

Unit: 50 BIN ▼

Xmax: 45.0

Ymax: 54.0

Zmax: 36.0

Name: EneDep

NX: 100.0

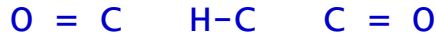
NY: 100.0

NZ: 100.0

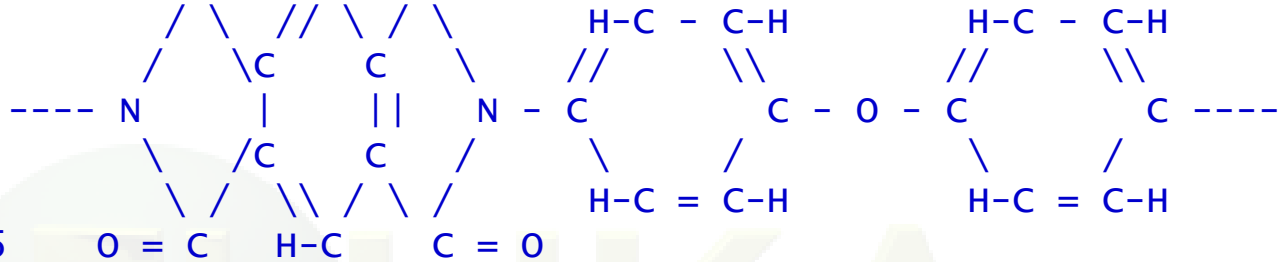
# Anatomy of a card mini-dialog

\* Polypyromellitimide Polyimide, Kapton

\* Chemical



\* Formula



\* C H N O  
\* 22 10 2 5

MATERIAL			1.43				Polyimid
COMPOUND	10.0	HYDROGEN	22.0	CARBON	2.0	NITROGEN	Polyimid
COMPOUND	5.0	OXYGEN					Polyimid

<b>MATERIAL</b>	Name: Polyimid	#	p: 1.43
Z:	Am:	A:	dE/dx: ▼
<b>COMPOUND</b>	Name: Polyimid ▼	Mix: Atom ▼	Elements: 6 ▼
f1: 10.0	M1: HYDROGEN ▼	f2: 22.0	M2: CARBON ▼
f3: 2.0	M3: NITROGEN ▼	f4: 5.0	M4: OXYGEN ▼
f5:	M5: ▼	f6:	M6: ▼



# Editing Cards

While in **input editor** you can work in two modes:

1. **Card mode**: manipulate the cards as a unit  
(e.g. to copy, paste, delete, change order of cards)
2. **Edit mode**: manipulate the contents of the card

Edit mode is activated immediately after adding a new Card, by hitting Enter or with the mouse click

To leave edit mode click the Esc or click somewhere else

The active item (**what**) is highlighted with a grey rectangle and highlighted also in the card viewer at the bottom of the editor

A range of cards can be selected with:

- Shift + Mouse
- Shift + Up/Down arrows

# Validating input and Error correction

- Flair validates the input file while loading and each card during editing
- Errors are highlighted in **red**
- As of now, only **syntactical errors** are checked, and a few **logical errors**
- Popup-menu option "Show errors" displays a short message on what is expected as correct value
- Menu item "Input / Filter Invalid" shows only the invalid cards from the last filtered view

# Material Database

- Flair contains an internal database of  $\sim 500$  predefined materials and/or compounds
- Some ( $\sim 300$ ) with the **Sternheimer** parameters

**Please use these data as Reference only!**

- Validate **always** the correctness of the data
- If errors found please contact the author
- The database can be edited, and populated with your own materials. In this case a local copy of the database will be made in  $\sim/.flair$  directory

# Starting a Run

- Flair can start a simulation (**single run**) based on the input file
- **Multiple runs** can be started by overriding some options, like **#defines**, **title**, **random number seed** and **number of starting particles** (primaries)
- Flair will try to **"attach"** to a run. Using only the information from the output files generated by FLUKA, flair will try to identify the directory where the run takes place and monitor the progress of the selected run
- During the execution of the run the user can view the output files in the **"Files Frame"** under the **"Run Tab"**



# Other goodies

Flair has a **lot of functionalities** that are not shown in this tutorial

Some of these will be shown during the exercises of the course

We would advice the users to go through the various menus and help page and try these out

# Tips & Tricks

## ● Mouse

- right-click

opens the popup-menu with the most important actions

## ● Keyboard

- Ctrl-Enter

Check the accelerators on the menus  
Performs the default action in every frame.  
e.g. Add a card in the Input Editor  
start a run in the Run Frame

- Ctrl-Space

Access popup-menu (like **right-clicking**)

- Listboxes

All listboxes in flair are **searchable** and case insensitive. Type the first characters of the string you are searching and the closest match will be highlighted

**Ctrl-G** repeats the search. **Space** selects/deselect item

- +, -, Ins

While editing the **REGION** expression shows a list of all available bodies

# Known Bugs / Limitations

- **Unicode / International** characters do not work well and should be avoided
- **Gnuplot:**
  - **<4.2** has a bug in the number of palette colors, and on the cblabel for the wxt terminal
  - **4.4.3** has a non linear behavior on the palette colors for the xterm terminal
- **Inline comments**, and comments inside REGION definition are treated as one comment preceding the input card
- **REMEMBER** always that the **.flair** and **.inp** are different thing  
Do not save the project as **.inp** or the input file as **.flair**