A decorative graphic consisting of blue lines and corner markers. A vertical line on the left and a horizontal line at the top intersect at a small circle in the top-left corner. Another horizontal line is positioned below the top one, and a vertical line on the right intersects it at a small circle in the bottom-right corner. The main title is centered between the top and middle horizontal lines.

# FLUKA: Flair Introduction

4<sup>th</sup> FLUKA Course  
Univ. of Houston TX, May 14-18, 2007



# Why is UI design important

- User Interfaces are what allows end users to interact with an application.
- A good UI will make an application intuitive and easy to use
- Excellent applications without good UI will be less popular than inferior ones with a good UI

# What makes a good UI?

## General:

- Simple
- Intuitive
- Respects the commonly accepted conventions
- Visually organized
- Native look
- Easily install and setup
- Extensible / Programmable

## FLUKA:

- Do not hide the inner functionality
- Provide a platform for working/analyzing results

# Software choices [1/2]

## Requirements:

- Open source software
- Multiplatform with easy installation
- Minimum requirements on other package
- Large community of users and years of development

## Python [<http://www.python.org>]

is a scripting language which is:

- interpreted
- interactive
- object-oriented
- like pseudo code
- dynamically typed
- available for many platforms
- extensible with C-API

# Software choices [2/2]

- **Tkinter** [<http://wiki.python.org/moin/TkInter>]  
default GUI toolkit for Python.  
Good for simple UIs.  
Portable, wrapper around tk/tcl
- **Gnuplot** [<http://www.gnuplot.info>]  
is a command-line driven, interactive function plotting program specially suited for scientific data representation. Gnuplot can be used to plot functions and data points in both two and three dimensions and in many different formats.
- **Povray** [<http://www.povray.org>]  
POV-Ray™ is short for the Persistence of Vision™ Raytracer, a tool for producing high-quality computer graphics. POV-Ray™ is copyrighted freeware. POV-Ray is the worlds most popular raytracer.

# What is flair [1/2]

- **FLUKA Advanced Interface** [<http://www.fluka.org/flair>]
- **All-in-one** Graphical Interface
- With 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:** transformation, optimizations and debugging
- **Compilation** of the FLUKA Executable
- **Running and monitoring** of the status of a/many run(s)

# What is flair [2/2]

## Back-End interface:

- Inspection of the output files
- Post processing (merging) the output data files
- Plot generation through an interface with **gnuplot**  
Input information, **USRxxx**, **RESNUCLEI** and **geometry**
- 3D photo-realistic images with PovRay (ToDo)

## Other Goodies:

- Access to FLUKA manual as hyper text
- Checking for release updates of FLUKA and flair
- Nuclear wallet cards
- Library of materials
- Database of geometrical objects (ToDo)
- Programming python **API**

# 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 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.
- Can handle any FLUKA input format (reading & writing), but internally it works using the **names format** for the input, **free with names** for the geometry (Recommended way of working)
- 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 web site to download code and documentation  
<http://www.fluka.org/flair>
- Installation procedure
  - RPM method: **strongly recommended!** on systems that support the RPM. The RPM will create all **file association**, **menu items** and keep track of updates and files installed.  
The RPM 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
  - tar.gz method: Manual installation on platforms like (MS-Windows, Mac OS...). Please follow the instructions on the <http://www.fluka.org/flair/download.html> and for special instruction on the FAQ <http://www.fluka.org/flair/faq.html>

# Version Numbering

- Flair version numbering, has the form: **flair-M.m-R**
- **M: Major version**, this number is increased by one only when a major modification in the structure of the program takes place. During the initial phase of development is 0.
- **m: Minor version**, this number is increased by one every time an addition in the program's functionality is made. e.g. Adding new plotting forms, mechanism, databases etc.
- **R: Patch level**, this number is increased when bug fixes take place or minor changes in the functionality. e.g. Addition of extra fields in a form etc.
- The **About Dialog** of the program displays the **Major** and **minor** version as well the CVS release number, every time changes in the program are committed to the CVS server.
- The Update dialog of flair will ask the user to check on the web every 30 days (default) for a new **flair version (M.m)** and FLUKA release.

# Starting flair

## Programs Menu

- Click the icon of Flair from the programs menu. Flair is registered under the Science/Physics category

Note: **freedesktop** however places this category in various places depending your linux distribution and window manager

Typical places:

- Applications
- Edutainment
- Edutainment / Science

## Window Manager (only via RPM installation):

- Flair makes an association of the following extensions:



\*.flair



\*.fluka, \*.inp

## Console

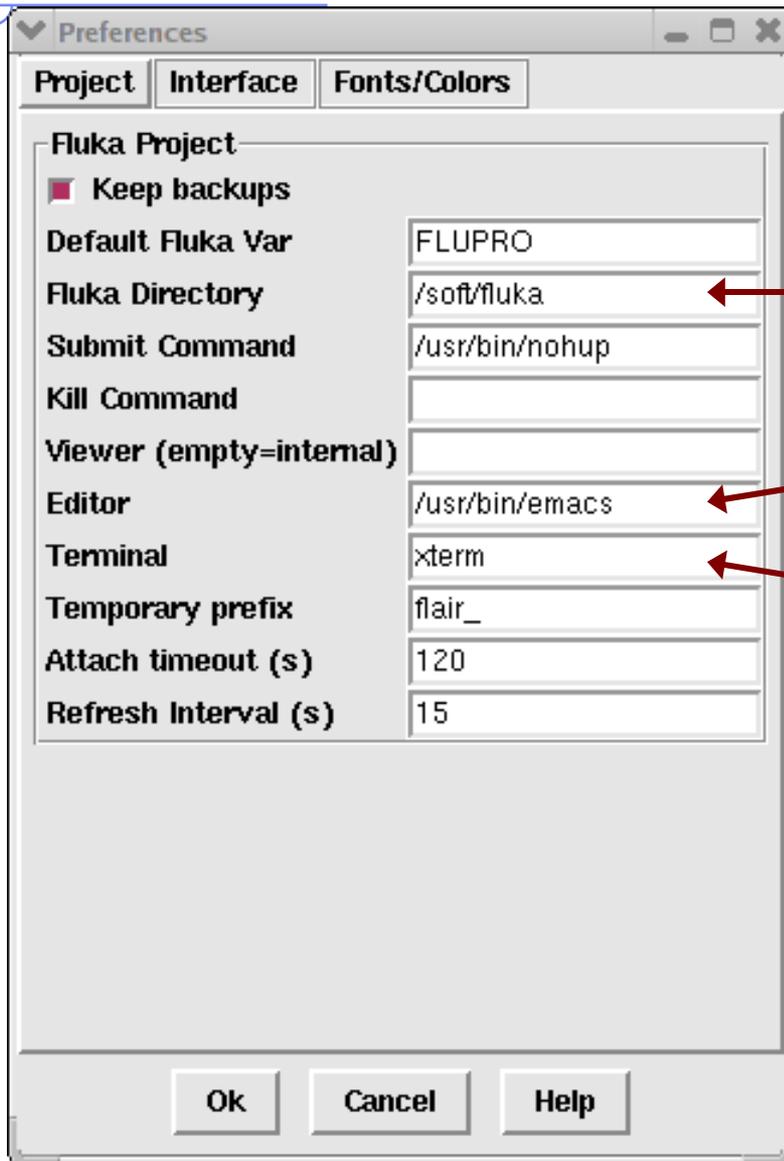
- Type the command `/usr/local/bin/flair`
- It is recommended to place in your PATH the `/usr/local/bin` directory

# Startup

During startup flair will perform the following:

- Open an “xterm” to be used as the Output window
- Display the About Dialog
- Check for the existence of FLUPRO variable, and if not found will open the Preferences dialog to set explicitly the FLUKA path.  
**WARNING:** Window managers (GNOME, KDE...), as well command shells (bash, tcsh, ash...) have a different configuration file where they expect the environment variable.
- Display the “Tip of the Day” dialog
- Open the “Check for Updates” dialog

# Basic Preferences



Set your FLUKA directory

Set your favorite editor

Set your favorite console program

# Program Interface

The screenshot shows the FLUKA program interface. On the left is a tree browser showing a hierarchical structure of the simulation setup, including folders for Input, Geometry, Media, Physics, Transport, Biasing, Scoring, Processors, and Data. The main window displays configuration parameters for a simulation titled 'n\_TOF lead target'. The parameters are organized into sections: #define, GLOBAL, DEFAULTS, BEAM, BEAMPOS, GEOBEGIN, and various object definitions (SPH, RPP, PBTARGET, NICHE). The BEAM section shows parameters like Energy (20.0), Part (PROTON), and various beam characteristics. The object definitions section lists parameters for BLKBODY, VOID, WATERCNT, and PBTARGET. At the bottom, there is a code snippet defining beam characteristics and a table of parameters.

**Tree Browser**

**Embedded Applications**

**Configuration Parameters:**

```

TITLE n_TOF lead target
#define Name: test1
#define Name: test2
#define Name: test3
#define Name: test4
GLOBAL Max #reg: Analogue: DNear:
Input: Names Geometry: Free
DEFAULTS EET/TRAN
Beam characteristics
BEAM Beam: Energy E: 20.0 Part: PROTON
      Δp: Gauss Δp(FWHM): 0.082425 Δφ: Gauss Δφ: 1.7
      Shape: Rectangular Δx: Δy: Weight: 1.0
BEAMPOS x: -25.0 y: -0.5 z: -10.0
        x: 10.0 y: 0.0 Dirz: POSITIVE
GEOBEGIN
Title: n_TOF lead target
Black body
SPH BLKBODY X: 0.0 Y: 0.0 Z: 0.0
        R: 10000000.0
Void sphere
SPH VOID X: 0.0 Y: 0.0 Z: 0.0
         R: 1000000.0
Water container
RPP WATERCNT Xmin: -43.0 Xmax: 43.0
              Ymin: -53.6 Ymax: 53.6
              Zmin: -32.5 Zmax: 35.0
Lead Target
RPP PBTARGET Xmin: -40.0 Xmax: 40.0
             Ymin: -40.0 Ymax: 40.0
             Zmin: -30.0 Zmax: 30.0
DDD NICHE Ymin: -15.0 Ymax: 15.0
define beam characteristics, properties of primary particle
*.....1.....2.....3.....4.....5.....6.....7.....
BEAM -20.0 -0.082425 -1.7 1.0 PROTON
    
```

**Status Bar:** Inp: ntof33.inp Exe: Dir: /home/bnv/prg/physics/fluka/flair/examples Filtered 37 out of 37

- Wrapper of standalone applications
- Tree browser to select application
- Allow different ways of viewing the same object
- Input:
  - ♦ Filtering Cards
  - ♦ Show card links
  - ♦ Units: i.e. 20 GeV/c (ToDo)
  - ♦ Data validation
  - ♦ Import/Export on various formats
- Process:
  - ♦ Debugging
  - ♦ Compilation
  - ♦ Run monitoring
  - ♦ Merging
- Plotting:
  - ♦ Interface to plot packages
  - ♦ Table of Isotopes
- Python Libraries:
  - ♦ Input file manipulation
  - ♦ Processing
  - ♦ Plotting

# Menus

**File   Edit   Card   Input   Tools   View   Options   Help**

File	I/O, export to other formats, printing, recent projects
Edit	Common editing features: Cut, Paste, Add, Del, Clone, Filter
Card	Add or change cards in input; grouped by Categories
Input	Commands to manipulate input cards
Tools	General purpose commands: Terminal, Append cards
View	Accessing various views of flair
Options	Customize preferences
Help	Access to help, check for updates, web page, about dialog

# Toolbar



Project Control I/O

Undo/Redo

Cut/Copy/Paste

Find/Filter

Add/Del/Clone

View  
Backward/Forward



Move Up/Down

## Quick Access to:

- |                    |                      |
|--------------------|----------------------|
| 2. Project Frame   | 2. Run               |
| 3. Input Editor    | 3. View output files |
| 4. Process Summary | 4. Data merging      |
| 5. Compile         | 5. Plots             |
| 6. Debug Geometry  | 6. Databases         |
|                    | 7. Help              |

# Input Templates

- When requesting a new input or a new project flair will prompt to select an input template
- flair default templates are prefixed with "D:"
- The user can create his own set of input templates. They are normal FLUKA input files and they have to be places in the directory (create if not existing)  
~/.flair/templates
- User templates will be prefixed with "U:"

# Input Editor

- With the input editor the user can manipulate the input cards.
  - Create new ones
  - Edit & modify existing ones
  - Copy & Paste
  - Clone (Duplicate)
  - Import from other input files
  - Validate the correctness of the cards
  - Error filtering
  - Rearrange order
- The editor will always try to rearrange the input cards (only if needed) to create a valid FLUKA input file.

# Card Categories

For easier access, cards are groups 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
- **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
- **Developers** *...reserved for FLUKA developers...*
- **Preprocessor** definitions for creating conditional input files

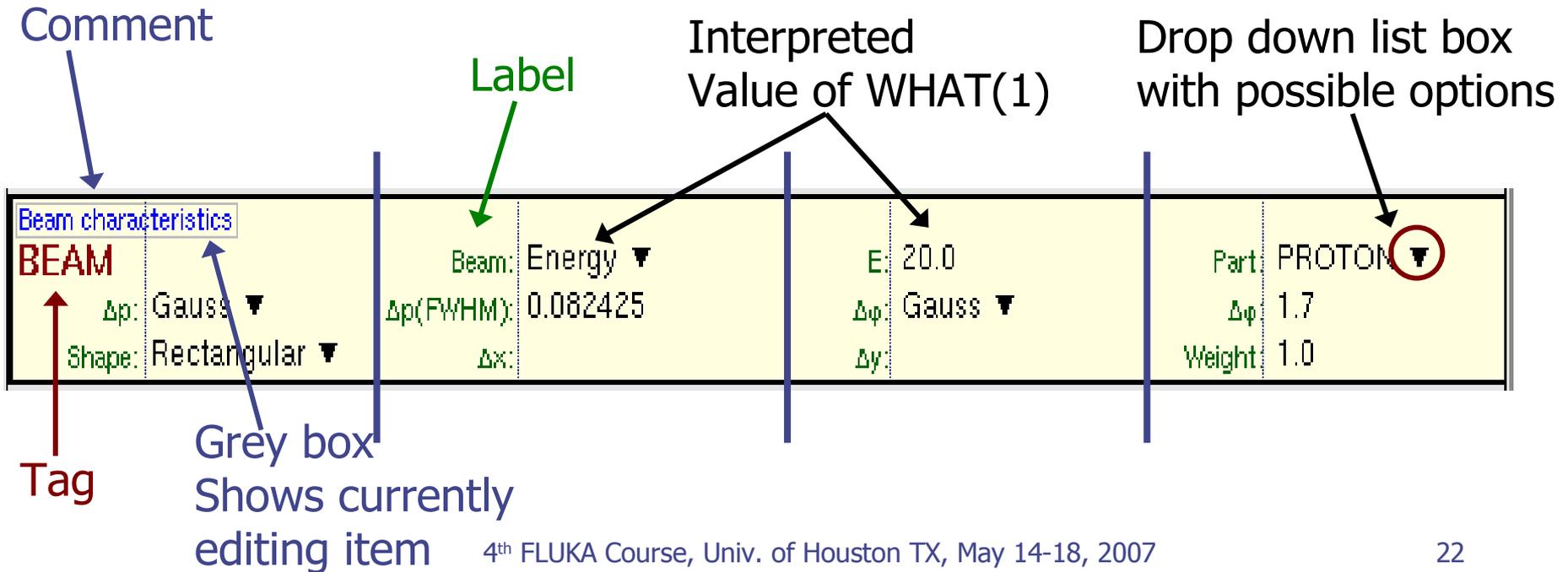
# Concepts: Extended Cards

- 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** and **Multiple number of whats** (0=sdum, 1-6 first line, 7-12 continuation line...) and one field of **extra** information (multi line string)
  - **State** (Enable/Disable)
- Special cards to homogenise the FLUKA input file representation:
  - **REGION**, referring to a region declaration in geometry
  - **COMPOUND**, all compound cards related to one material are joined in one card
  - **END** card for bodies/regions is no longer required
- 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 [1/2]

- 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 [2/2]

\* Energy deposition in 3D binning

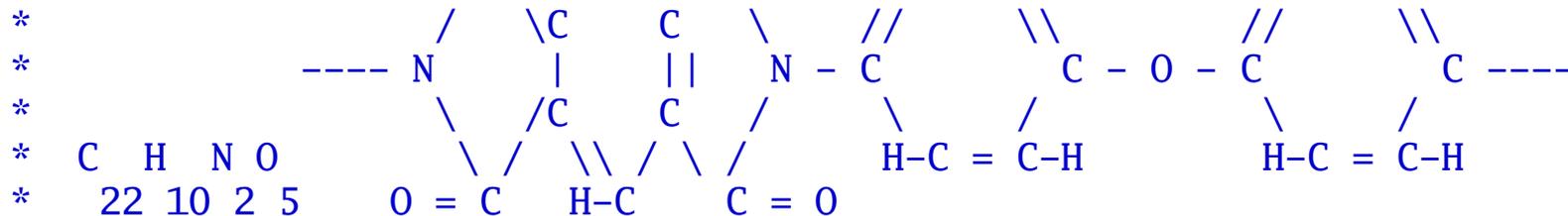
USRBIN	10.0	ENERGY	-50.0	45.0	54.0	36.0EneDep
USRBIN	-45.0	-54.0	-33.0	100.0	100.0	100.0&

<b>USRBIN</b>	Unit: 50 BIN ▼	Name: EneDep
Type: X-Y-Z ▼	Xmin: -45.0	NX: 100.0
Part: ENERGY ▼	Ymin: -54.0	NY: 100.0
	Zmin: -33.0	NZ: 100.0
	Xmax: 45.0	
	Ymax: 54.0	
	Zmax: 36.0	

\* Polypyromellitimide Polyimide, Kapton

\* Chemical

\* Formula



\* MATERIAL

Polyimid

1.43

COMPOUND

10.0

HYDROGEN

22.0

CARBON

2.0

**MATERIAL**

Name: Polyimid

#

p: 1.43

Z:

Am:

A:

dE/dx: ▼

**COMPOUND**

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 are working in two modes

2. **Card mode**: where you manipulate the cards as a unit. Copy, paste, delete, change order of cards
3. **Edit mode**: where you manipulate the contents of the card. Edit mode is activated immediately after adding a new card or by hitting Enter or with the mouse click. To leave edit mode click the Esc or with the mouse click somewhere else.

The active item (what) is highlighted with a grey rectangle and highlighted also in the card viewer below the editor.

- Shift+Mouse or Shift+Up/Down arrows selects a range of cards

# Validating input and Error correction

- flair validates the input while loading and each card during editing.
- Errors are highlighted with red.
- 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 / compounds.
- Some ( $\sim 300$ ) with the **Sternheimer** parameters
- **Please use this 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 **single run** based on the input file, or **multiple runs** by overriding some options, like #defines, title, random number seed and number of starting particles
- 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 Run.
- During the execution of the run the user can view the output files in the **“Files Frame”**

# 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.

Add a card in the Input Editor

- Ctrl-Space

Access popup-menu

- 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/deselects item

- +, -, Ins

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

# ToDo: Features to be added

- **Interface**
  - Working on multiple projects
  - Exportation of processing scripts and various formats (MCNP...)
- **Input Editor**
  - Undo/Redo
  - Show hidden cards
  - Geometry manipulation (Transformations, CSG optimization etc)
- **Post Processing**
  - Re-binning or USRBINS
  - Maximum trace
- **Plotting:**
  - Information of Input File
  - Double differential quantities (USRBDX, USRYIELD...)
  - 3D Ray Tracing interface to povray
  - Particle tracks

# How to Contribute

- Python programming
  - Parsing and processing output files
  - Web based database for sharing resources with other users
  - ...
- Input Editor
  - Cards Layout, in other formats (from 3 up to 8 columns)
  - Labels have to be intuitive, if something is not comprehensible please propose an alternative
- Manual, Online documentation, Tips database
- Icons for tool bars and cards
- Gnuplot scripts or ideas for better presentation
- Comments & Ideas, on new features that one wants to see
- **Testing, Bug reporting**
- ...