A new framework for the ALICE-Masterclasses

Jonas Toth
CERN Summer Student

23. August 2018

Motivation and Goal

- **Increase public outreach**
- deploy everywhere
- if possible translate to local language
- **more ALICE MasterClasses in one package**
- Strangeness
- Raa
- J/Psi
- **Allow other experiments and institutes to join**
- **address classical software topics**

Motivation and Goal

- **Increase public outreach**
- deploy everywhere
- if possible translate to local language
- **more ALICE MasterClasses in one package**
- Strangeness
- Raa
- J/Psi
- **Allow other experiments and institutes to join**
- **address classical software topics**

Motivation and Goal

- **Increase public outreach**
- deploy everywhere
- if possible translate to local language
- **more ALICE MasterClasses in one package**
- Strangeness
- Raa
- J/Psi
- **Allow other experiments and institutes to join**
- **address classical software topics**

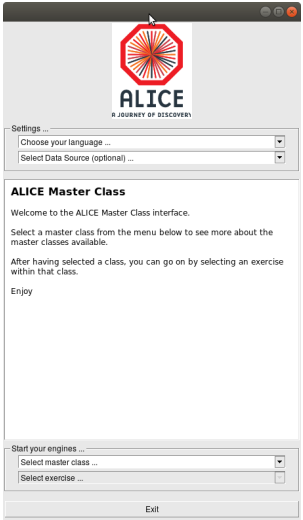
Motivation and Goal

- **Increase public outreach**
- deploy everywhere
- if possible translate to local language
- **more ALICE MasterClasses in one package**
- Strangeness
- Raa
- J/Psi
- **Allow other experiments and institutes to join**
- **address classical software topics**

Starting point

- Version from Christian Holm Christensen
- Repository on **gitlab.cern.ch**
- Macro-based solution with Strangeness and Raa Classes
- initial translation facilities

Starting point



New: Full Translation

- **Provide a way to easily add a new language**
- **translate whole program**
- added German translation, Dutch is WIP
- file-based solution does not require programming skills
- static analysis for consistency
- semi-automatic snippet translation through Google Translate
- **Groups are encouraged to add translations**

New: Full Translation

- Provide a way to easily add a new language
- translate whole program
- added German translation, Dutch is WIP
- file-based solution does not require programming skills
- static analysis for consistency
- semi-automatic snippet translation through Google Translate
- Groups are encouraged to add translations

New: Full Translation

- **Provide a way to easily add a new language**
- **translate whole program**
- added German translation, Dutch is WIP
- file-based solution does not require programming skills
- static analysis for consistency
- semi-automatic snippet translation through Google Translate
- **Groups are encouraged to add translations**

New: Full Translation

```
/// translation/EntryPoint/Description.en.html
R"(
<html>
<head>
</head>
<body>
<h1>ALICE Master Class</h1>

<p> Content of the user-facing document. Shortened... </p>
</body>
</html>)"
```

New: Full Translation

```
/// translation/EntryPoint/keys_gui_trans.txt.en

/// Key-Value store for English.
/// every language will define its own set of strings.
RegisterText("GUISettings", "Settings ...");
RegisterText("GUIDataSource", "Select Data Source (optional) ...");
RegisterText("GUILabelClassConfig", "Start your engines ...");
RegisterText("GUIChooseLanguage", "Choose your language ...");
RegisterText("GUIChooseClass", "Select master class ...");
RegisterText("GUIChooseExercise", "Select exercise ...");
RegisterText("GUIExitButton", "Exit");
```

New: Full Translation

```
/// src/EntryPoint/GUITranslation.h
/// Provide a class that will provide all available strings
/// in English.
struct TGUIEnglish : Utility::TLanguageProvider {
  TGUIEnglish(Utility::ESupportedLanguages lang = Utility::English)
    : TLanguageProvider(lang) {
      /// Registering a Key-Value store for
    /// small strings that are translated
    std::cerr << "Registering English  GUI Selection\n";
    #include "EntryPoint/keys_gui_trans.txt.en"
    /// Statically compile in whole translated HTML files.
    RegisterText("Description",
    #include "EntryPoint/Description.en.html"
    );
  }
};
```

New: Full Translation

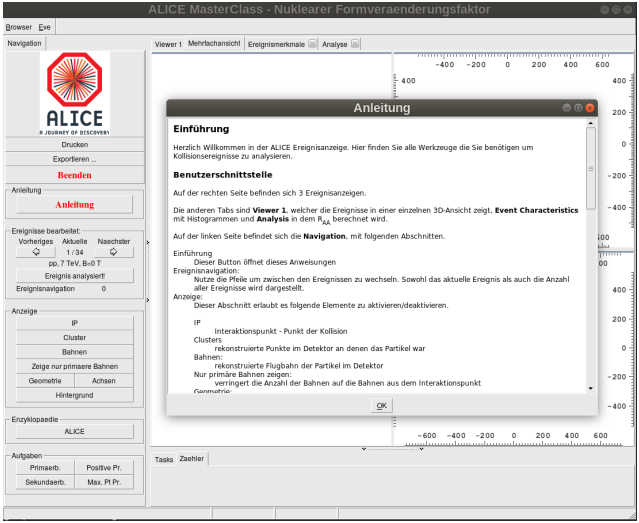
```
/// src/EntryPoint/GUITranslation.h
/// Makes the strings available as methods.
/// This ensures on compile-time that only existing strings are
/// used and avoids common errors like typos in the Key-Value-Access.
struct TGUIEnglish : Utility::TLanguageProvider {
  TGUIEnglish(Utility::ESupportedLanguages lang = Utility::English);
  /// Create all Methods to request the corresponding text snippets.
  LANG_KEY(GUISettings)
  LANG_KEY(GUIDataSource)
  LANG_KEY(GUILabelClassConfig)
  LANG_KEY(GUIChooseLanguage)
  LANG_KEY(GUIChooseClass)
  LANG_KEY(GUIChooseExercise)
  LANG_KEY(Description)
  LANG_KEY(GUIExitButton)
};
```

New: Full Translation

```
/// src/EntryPoint/GUITranslation.h
/// Adding another translation in code is very easy.
/// The 'LANG_TRANSLATION' macro creates a new class that inherits
/// all methods from the English pendant and only replaces the
/// string registrations.
LANG_TRANSLATION(TGUI, German)
{
  std::cerr << "Registering German GUI Selection - EntryPoint\n";
  #include "EntryPoint/keys_gui_trans.txt.de"

  RegisterText("Description",
  #include "EntryPoint/Description.de.html"
  );
}
```

New: Full Translation



Refactoring

Refactoring

Structural Changes of Code without affecting behaviour.

- Missing tests can not catch random breakage
- To refactor you need to test, to add tests you need to refactor.
- **Obstacle** every masterclass was copy&pasted

Refactoring

Refactoring

Structural Changes of Code without affecting behaviour.

- Missing tests can not catch random breakage
- To refactor you need to test, to add tests you need to refactor.
- **Obstacle** every masterclass was copy&pasted

Refactor to what?

SOLID Principles

- Single Responsibility Principle
- Open/Closed Principle
- Liskov Substitution Principle
- Interface Segregation Principle
- Dependency Inversion Principle

What we want

```
/// Collect statistics over all events in the dataset.
class TRaaStatistics
{
  public:
  void AddEventStatistics(const gsl::span<TEveTrack* const> Primaries);
  void AddSecondaryMultiplicity(Int_t NumberSecondaries);
  void ClearStatistics();

  const TH1D& HistMultiplicity() const { return fMultiplicityHist; }
  const TH1D& HistMultiplicityMinPt() const { return fMultHistMinPt; }
  const TH1D& HistMultiplicitySecondaries() const { return fMultHistSec; }
  const TH1D& HistPt() const { return fPtHist; }
  const TH1D& HistCharge() const { return fChargeHist; }
  const TH1D& HistPhi() const { return fPhiDist; }
};
```

What we don't want

```
/// Just the public interface, a lot more coupling happens through
/// implicitly shared state.
class Counter : public TGMainFrame
{
  public:
  Bool_t fEnableCharge;
  Bool_t fAllowAuto;
  Double_t fHighestPt;
  TEveTrack* fHPtTrack;
  Int_t fSecondaries;
  Counter(const TGWindow* p, UInt_t w, UInt_t h,
          ECollisionSystem collSys, TTrackAnalysis* hists,
          Bool_t allowAuto = false);
  TGNumberEntryField* AddField(TGCompositeFrame* cf, TGLayoutHints* lh,
                               Bool_t isInt = false, Bool_t neg = false);
  void Build();
  /// ...
```

What we don't want

```
  /// ...
  void Reset(ECollisionSystem collSys, Bool_t ResetAll = true);
  void DoExit() { this->UnmapWindow(); }
  void CountAutomatic();
  void Reset(Bool_t ResetAll = true);
  void Instructions();
  void Fill(Double_t px, Double_t py, Double_t pz, Int_t charge);
  void IncreaseMult(Double_t pt);
  void Publish();
  void FillHistAuto(Double_t pt, Double_t q, TEveTrack* track,
                    Double_t phi);
  /// End of interface
};
```

Extend Raa-Class

- criticism was a too monotonic experience when selecting only primary tracks
- Implement four tasks to choose from
- distinguish between primary and secondary particle
- distinguish between charges
- estimate particle momentum from curvature
- **explore more physics**

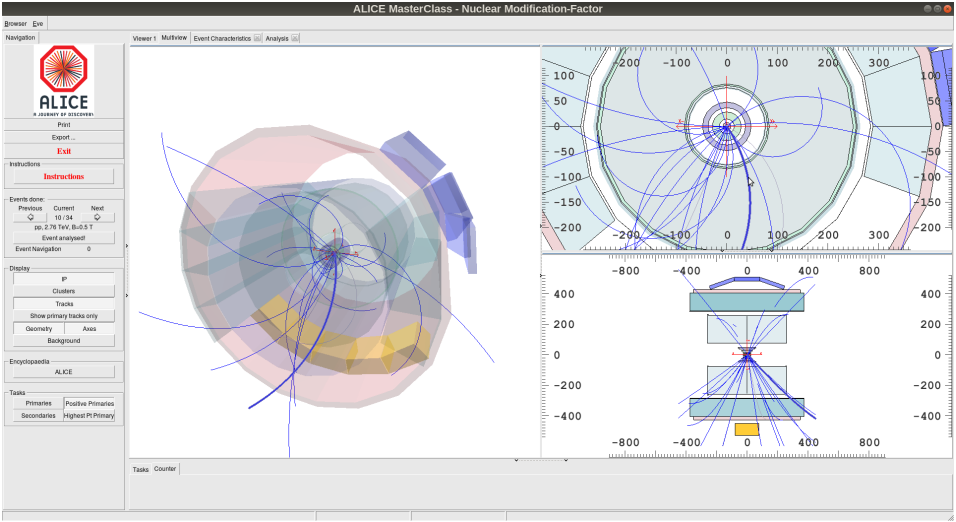
Extend Raa-Class

- criticism was a too monotonic experience when selecting only primary tracks
- Implement four tasks to choose from
- distinguish between primary and secondary particle
- distinguish between charges
- estimate particle momentum from curvature
- explore more physics

Extend Raa-Class

- criticism was a too monotonic experience when selecting only primary tracks
- Implement four tasks to choose from
- distinguish between primary and secondary particle
- distinguish between charges
- estimate particle momentum from curvature
- **explore more physics**

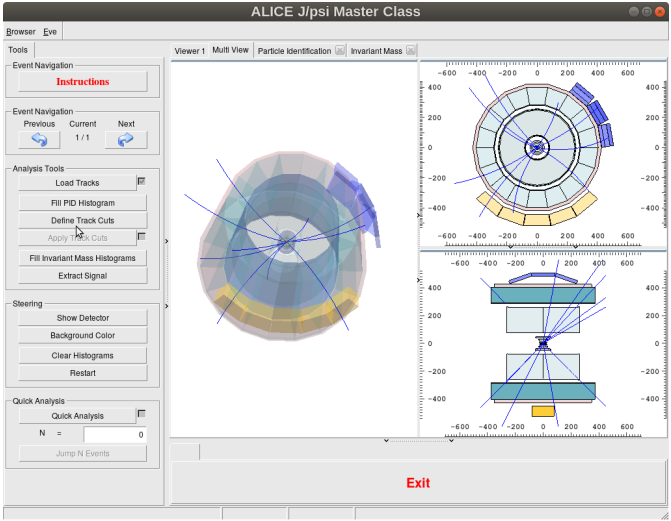
Extend Raa-Class



New J/Psi Masterclass

- Implemented from GSI
- right now just a code dump
- can be run the same way as Strangeness and Raa

New J/Psi Masterclass



Deployment and Usage

- ROOT6 to be future proof
- compiled executable (only Linux at the moment)
- requires only ROOT6 installed on the system
- CMake based process
- all masterclasses in one package
- Goal to have Linux AppImage

Deployment and Usage

- ROOT6 to be future proof
- compiled executable (only Linux at the moment)
- requires only ROOT6 installed on the system
- CMake based process
- all masterclasses in one package
- Goal to have Linux AppImage

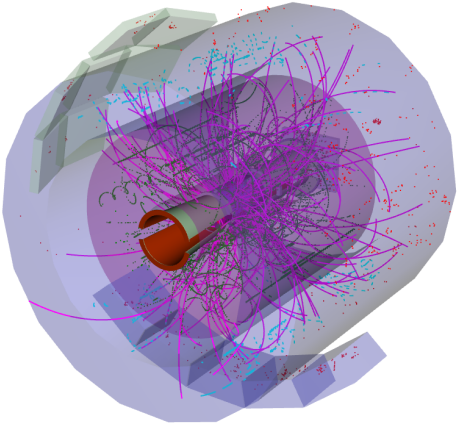
Deployment and Usage

- ROOT6 to be future proof
- compiled executable (only Linux at the moment)
- requires only ROOT6 installed on the system
- CMake based process
- all masterclasses in one package
- Goal to have Linux AppImage

Outlook

- jsROOT implements Event-Display for browser
- Link to Example
- Web-based Masterclasses instead of ROOT-GUI
- the basic architecture and structure can be reused
- WebAssembly, ASM.js and Emscrippten allow code-reuse

Outlook jsROOT



Questions

Finalizing my work

- automatically generating packages containing everything
- start refactoring J/Psi as well
- double-check all elements of the masterclasses to detect breakages