Bug Squashing Seminar

CTK Commandline Parsing/Wrapping

Christian Weber

Medical and Biological Informatics (E130)

May 20, 2014

dkfz.

Outline

Arguments Parsing - Selected Features

CTK CommandLine Parser supports

- arbitrarily ordered arguments
- optional arguments
- self-documentation

What we want . . .

```
./bin/mitkBrainStrippingMiniApps ProgressionVisualization

Provide at least input folder.
  -h, --help              Show this help text
  -i, --input             Input folder
  -o, --output            Output folder
  -b, --blanked           Only Display Morphology
  -n, --noresampling      Do not resample
  -m, --morphology        Morphology postfix.
  -s, --segmentation      Segmentation postfix.
  -l, --loadxml           Load data from xml
```

How we get it . . . (1)

```
// Includes
#include <QCoreApplication>
#include <ctkCommandLineParser.h>

int ProgressionVisualization( int argc, char* argv[] )
{
 // Parse Command-Line Arguments
 QCoreApplication app(argc, argv);
 ctkCommandLineParser parser;
 parser.setArgumentPrefix("--","-");
 parser.addArgument("help", "h", QVariant::Bool, "Show this help text")
 parser.addArgument("input", "i", QVariant::String, "Input folder");
 parser.addArgument("output", "o", QVariant::String, "Output folder");
 ...

addArgument( LONGNAME , SHORTNAME , TYPE , DESCRIPTION )
```

dkfz.

How we get it . . . (1)

```
// Includes
#include <QCoreApplication>
#include <ctkCommandLineParser.h>

int ProgressionVisualization( int argc, char* argv[] )
{
 // Parse Command-Line Arguments
 QCoreApplication app(argc, argv);
 ctkCommandLineParser parser;
 parser.setArgumentPrefix("--","-");
 parser.addArgument("help", "h", QVariant::Bool, "Show this help text")
 parser.addArgument("input", "i", QVariant::String, "Input folder");
 parser.addArgument("output", "o", QVariant::String, "Output folder");
 ...
```

addArgument(LONGNAME , SHORTNAME , TYPE , DESCRIPTION)

dkfz.

How we get it . . . (1)

```
// Includes
#include <QCoreApplication>
#include <ctkCommandLineParser.h>

int ProgressionVisualization( int argc, char* argv[] )
{
 // Parse Command-Line Arguments
 QCoreApplication app(argc, argv);
 ctkCommandLineParser parser;
 parser.setArgumentPrefix("--","-");
 parser.addArgument("help", "h", QVariant::Bool, "Show this help text")
 parser.addArgument("input", "i", QVariant::String, "Input folder");
 parser.addArgument("output", "o", QVariant::String, "Output folder");
 ...
```

`addArgument( LONGNAME , SHORTNAME , TYPE , DESCRIPTION )`

dkfz.

How we get it . . . (2)

Check if everything went fine :

```
bool ok = false;
QHash<QString, QVariant> parsedArgs =
 parser.parseArguments(QCoreApplication::arguments(), &ok);
if (!ok)
{
 MITK_ERROR << "Error parsing arguments:"
 << parser.errorString().toStdString()
 return EXIT_FAILURE;
}
```

How we get it . . . (2)

Check if everything went fine :

```
bool ok = false;
QHash<QString, QVariant> parsedArgs =
 parser.parseArguments(QCoreApplication::arguments(), &ok);
if (!ok)
{
 MITK_ERROR << "Error parsing arguments:"
 << parser.errorString().toStdString()
 return EXIT_FAILURE;
}
```

How we get it . . . (3)

Read out arguments:

```
if (parsedArgs.contains("input") || parsedArgs.contains("i"))
  {
    inputFolder =  parsedArgs["input"].toString().toStdString();
  }
  else
  {
    std::cerr << "Provide at least input folder."  << std::endl;


    std::cout << parser.helpText().toStdString();


  }
```

How we get it . . . (3)

Read out arguments:

```
if (parsedArgs.contains("input") || parsedArgs.contains("i"))
  {
    inputFolder =  parsedArgs["input"].toString().toStdString();
  }
  else
  {
    std::cerr << "Provide at least input folder."  << std::endl;


    std::cout << parser.helpText().toStdString();


  }
```

Including CommandLine Tools

CTK CommandLine Plugin provides

- integration of arbitrary executables
- automatic GUI generation
- asynchronous communictaion

How to include an executable

- activate cmdlinemodules
- provide a wrapper for the executable(*)
- add wrapper script directory in MITK

Including CommandLine Tools

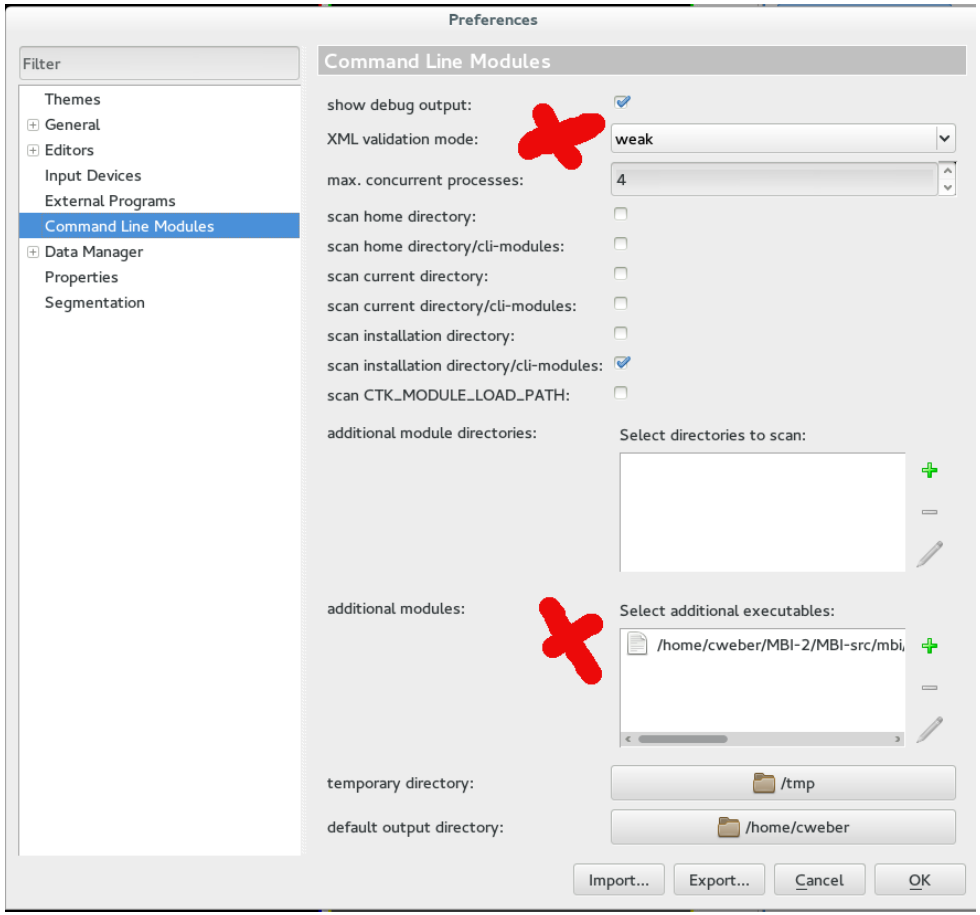
CTK CommandLine Plugin provides

- integration of arbitrary executables
- automatic GUI generation
- asynchronous communictaion

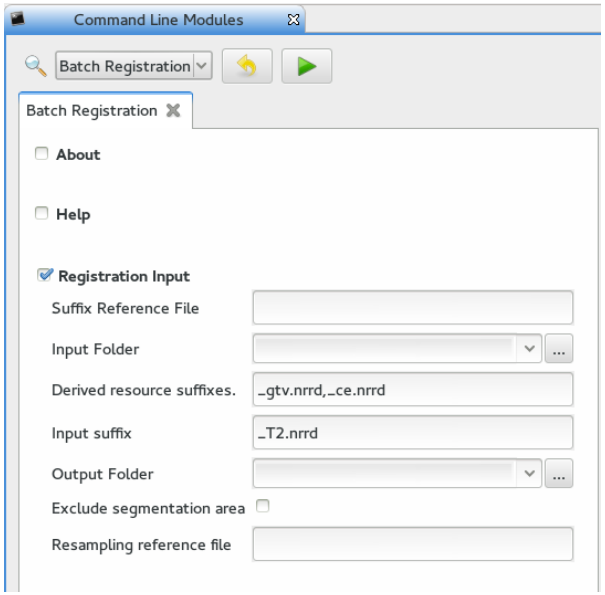
How to include an executable

- activate cmdlinemodules
- provide a wrapper for the executable(*)
- add wrapper script directory in MITK

Including CommandLine Tools



Including CommandLine Tools



Including CommandLine Tools

More Information at:

http://docs.mitk.org/nightly-qt4/org_mitk_views_cmdlinemodules.html

http://www.commontk.org/docs/html/CommandLineModules_Page.html

http://www.slicer.org/slicerWiki/index.php/
Slicer3:Execution_Model_Documentation