

# Programming into Slicer3

Sonia Pujol, Ph.D.

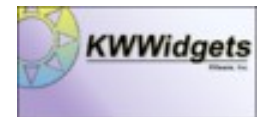
Surgical Planning Laboratory  
Harvard University

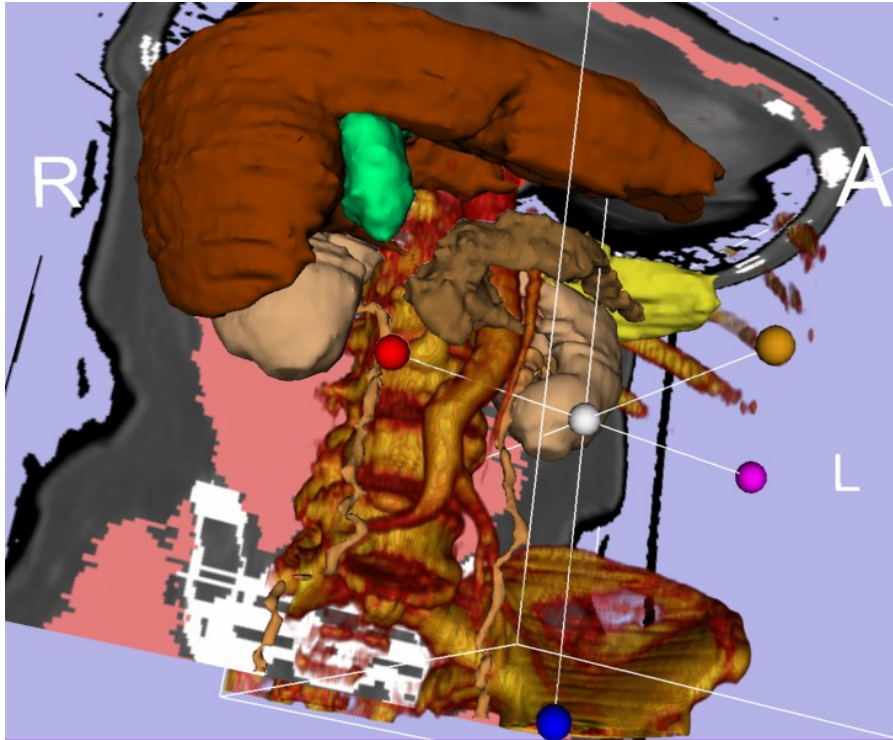


*Ferdinand Bol (1616-1680), The Officers of the Amsterdam Guild of Wine Merchants Alte Pinakothek, München*



# The NA-MIC Kit

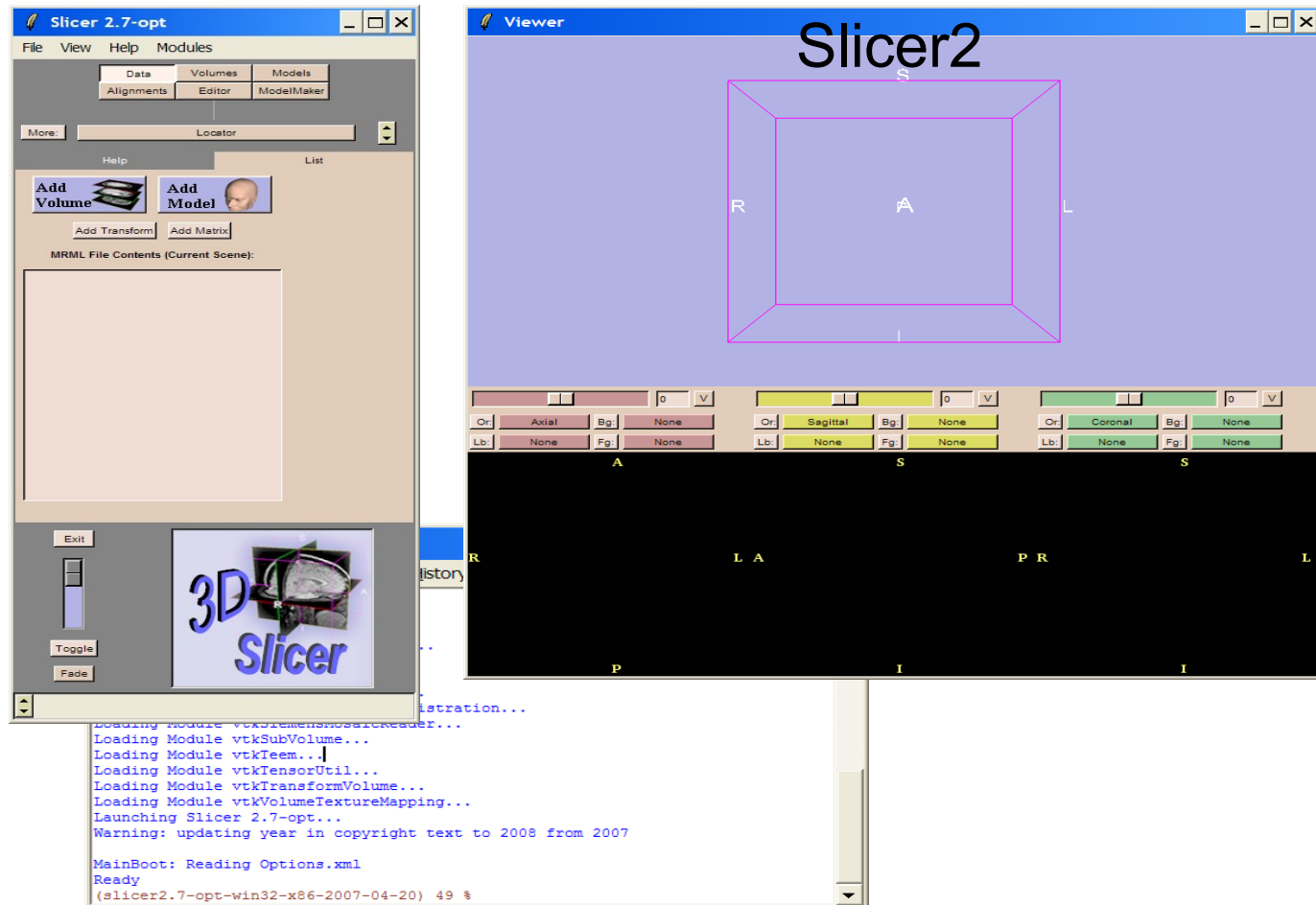




*Integrated Volume Rendering.  
R.Kikinis*

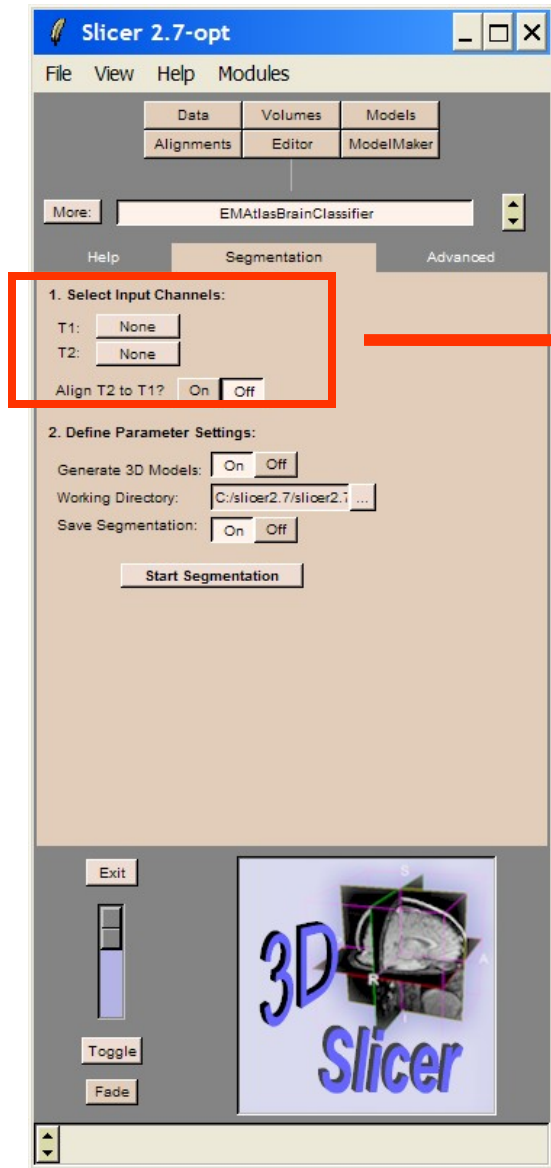
- An **end-user application** for image analysis
- An **open-source environment** for software development
- A software platform that is both **easy to use** for clinical researchers and **easy to extend** for programmers

# Before Slicer3





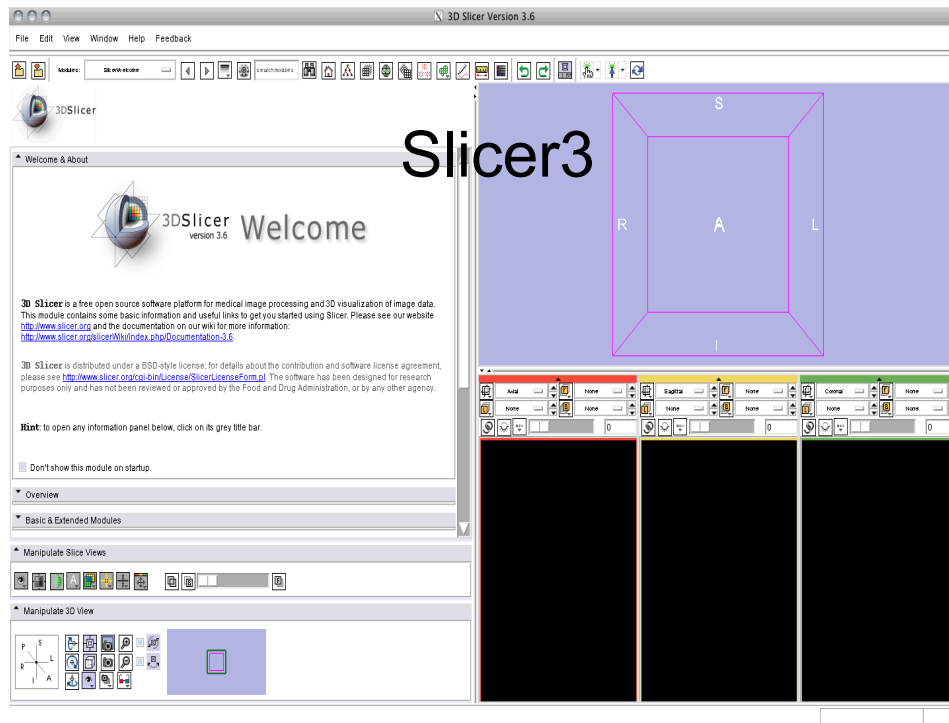
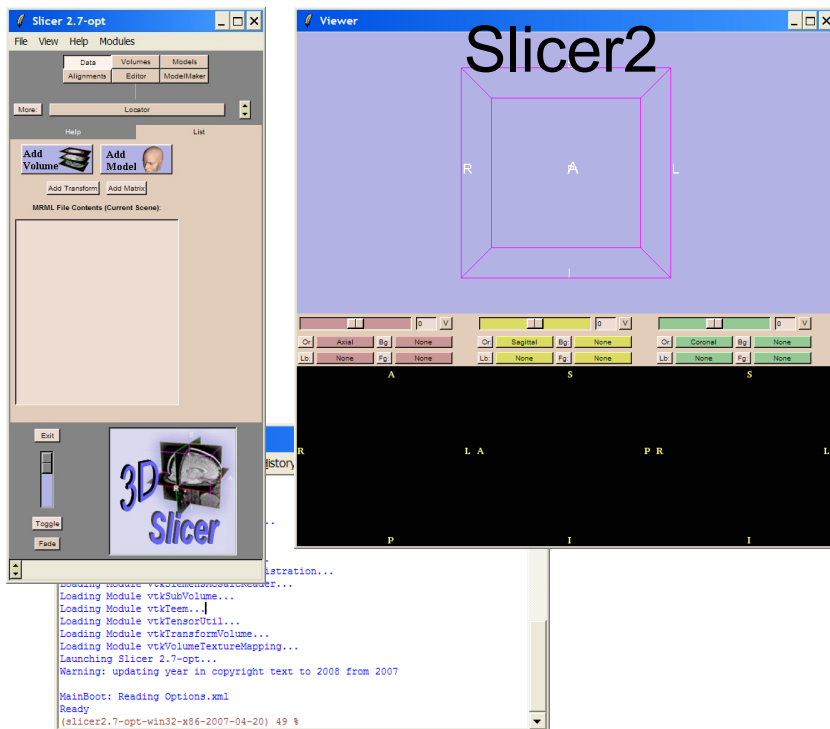
# Programming into Slicer2



```
#-----  
# 1. Step  
#-----  
set f $fSeg.fStep1  
DevAddLabel $f.Title "1. Select Input Channels: " WTA  
pack $f.Title -side top -padx $Gui(pad) -pady 1 -anchor w  
frame $f.flInput -bg $Gui(activeWorkspace)  
pack $f.flInput -side top -padx 0 -pady 0 -anchor w  
foreach frame "Left Right" {  
    frame $f.flInput.$frame -bg $Gui(activeWorkspace)  
    pack $f.flInput.$frame -side left -padx 0 -pady $Gui(pad) }  
foreach LABEL "T1 T2" Input "SPGR T2W" {  
    DevAddLabel $f.flInput.$Input " ${LABEL}:"  
    pack $f.flInput.$Input -side top -padx $Gui(pad) -pady 1 -anchor w  
    set menubutton $f.flInput.fRight.m${Input}Select  
    set menu $f.flInput.fRight.m${Input}Select.m  
    eval {menubutton $menubutton -text [Volume($EMAtlasBrainClassifier(Volume,$Input)),node) GetName]  
        relief raised -bd 2 -width 9 -menu $menu} $Gui(WMBA)  
    eval {menu $menu} $Gui(WMA)  
    TooltipAdd $menubutton "Select Volume defining ${Input}"  
    set EMAtlasBrainClassifier(mbSeg-${Input}Select) $menubutton  
    set EMAtlasBrainClassifier(mSeg-${Input}Select) $menu  
    # Have to update at UpdateMRML too  
    DevUpdateNodeSelectButton Volume EMAtlasBrainClassifier Seg-${Input}Select Volume,$Input  
    pack $menubutton -side top -padx $Gui(pad) -pady 1 -anchor w }  
frame $f.fAlign -bg $Gui(activeWorkspace)  
TooltipAdd $f.fAlign "If the input T1 and T2 are not aligned with each other set flag here"  
pack $f.fAlign -side top -padx 0 -pady 2 -padx $Gui(pad) -anchor w  
DevAddLabel $f.fAlign.lAlign "Align T2 to T1? "  
pack $f.fAlign.lAlign -side left -padx $Gui(pad) -pady 1 -anchor w  
foreach value "1 0" text "On Off" width "4 4" {  
    eval {radiobutton $f.fAlign.r$value -width $width -indicatoron 0  
        -text "$text" -value "$value" -variable EMAtlasBrainClassifier(AlignInput) } $Gui(WCA)  
    pack $f.fAlign.r$value -side left -padx 0 -pady 0 }
```



# From Slicer2 to Slicer3





# The New Execution Model

The screenshot displays the 3D Slicer Version 3.6 interface. On the left, the 'Hello World' module configuration panel is highlighted with a red border. It includes a 'Parameter set' dropdown, 'Status Idle', 'Input/Output' section with 'Input Volume' and 'Output Volume' dropdowns, and 'Discrete Gaussian Parameters' with a 'Variance' slider set to 0.5. Below the configuration panel are sections for 'Manipulate Slice Views' and 'Manipulate 3D View'. A red arrow points from the 'Input Volume' dropdown to the XML code on the right. The XML code, also enclosed in a red border, defines the module's metadata and parameters. The main 3D view area is currently empty, showing a black background.

```
<?xml version="1.0" encoding="utf-8"?>
<executable>
  <category> Demonstration </category>
  <title> Hello World </title>
  <description> Slicer Developer Course </description>
  <version> 1.0 </version>
  <documentation-uri> </documentation-uri>
  <license></license>
  <contributor>
    Sonia Pujol, Ph.D., Surgical Planning Laboratory, Harvard Medical School </contributor>
  <acknowledgements> National Alliance for Medical Image Computing (NAMIC), Grant U54 EB005149.
    </acknowledgements>
  <parameters>
    <label>Input/Output</label>
    <description>Input/output parameters</description>
    <image>
      <name>helloWorldInputVolume</name>
      <label>Input Volume</label>
      <channel>input</channel>
      <index>0</index>
      <default>None</default>
      <description>Input volume</description>
    </image>
    <image>
      <name>helloWorldOutputVolume</name>
      <label>Output Volume</label>
      <channel>output</channel>
      <index>1</index>
      <default>None</default>
      <description>Output filtered</description>
    </image>
  </parameters>
</executable>
```



# Slicer3 Execution Model

---

- This course is based on the [Execution Model](#) which provides a mechanism for incorporating command line programs as Slicer modules.
- Slicer Communication with external executables
- Jim Miller, Dan Blezek, Bill Lorensen (GE)

[http://www.slicer.org/slicerWiki/index.php/Slicer3:Execution\\_Model\\_Documentation](http://www.slicer.org/slicerWiki/index.php/Slicer3:Execution_Model_Documentation)



# Learning objective

---

Following this course, you'll be able

- 1) to **plug-in an external program** into Slicer3
- 2) to **implement an image filter** and to run the analysis from Slicer3
- 3) to **write and run a test** using the CTest tool

This course requires the following material

- Slicer 3.6 Software
- HelloWorld\_Plugin.zip

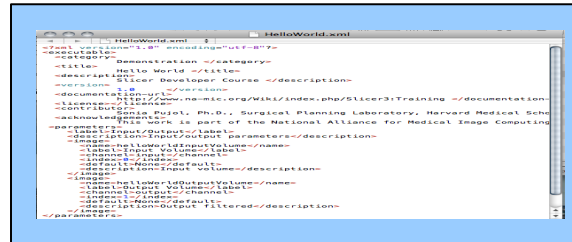
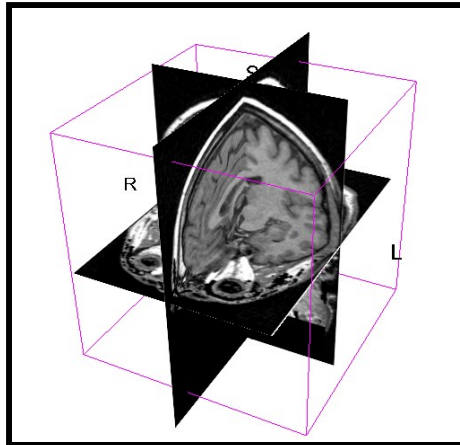
## **Disclaimer**

It is the responsibility of the user of 3DSlicer to comply with both the terms of the license and with the applicable laws, regulations and rules.

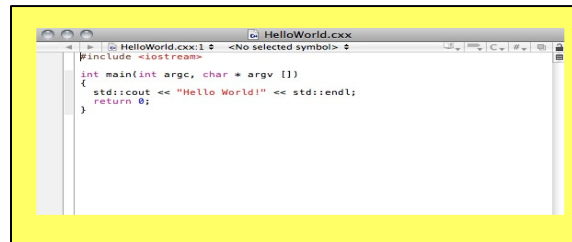


# HelloWorld\_Plugin.zip archive

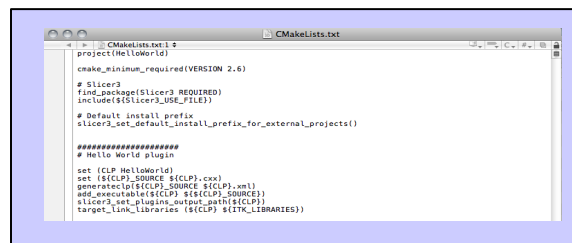
Unzip HelloWorld\_Plugin.zip



HelloWorld.xml  
(Execution Model)



HelloWorld.cxx  
(application)



CMakeLists.txt  
(CMake)

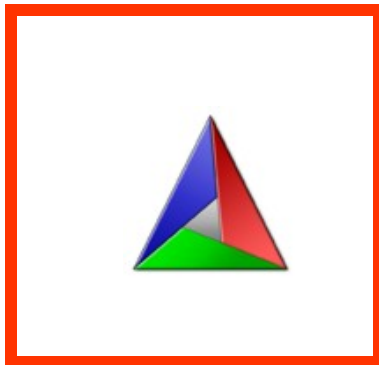
spgr.nhdr spgr.raw.gz  
(124 SPGR images)

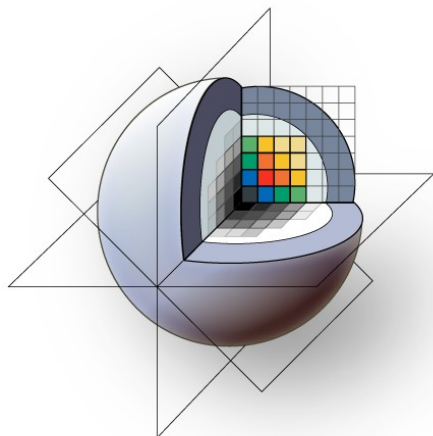
- **Part A:** integration of the HelloWorld program into Slicer3
- **Part B:** implementation of a Discrete Gaussian filter within the HelloWorld module
- **Part C:** implementation of a test for the HelloWorld module



# Slicer Programming Course

---





# 3DSlicer

## Part A: Integrating an executable into Slicer3

```

HelloWorld.xml
HelloWorld.xml
<?xml version="1.0" encoding="utf-8"?>
<executable>
  <category> Demonstration </category>
  <title> Hello World </title>
  <description>
    Slicer Developer Course </description>
  <version> 1.0 </version>
  <documentation-url>
    http://www.na-mic.org/Wiki/index.php/Slicer3:Training </documentation-
  <license></license>
  <contributor>
    Sonia Pujol, Ph.D., Surgical Planning Laboratory, Harvard Medical Scho
  <acknowledgements>
    This work is part of the National Alliance for Medical Image Computing
  <parameters>
    <label>Input/Output</label>
    <description>Input/output parameters</description>
    <image>
      <name>helloWorldInputVolume</name>
      <label>Input Volume</label>
      <channel>input</channel>
      <index>0</index>
      <default>None</default>
      <description>Input volume</description>
    </image>
    <image>
      <name>helloWorldOutputVolume</name>
      <label>Output Volume</label>
      <channel>output</channel>
      <index>1</index>
      <default>None</default>
      <description>Output filtered</description>
    </image>
  </parameters>
</executable>

```

```

HelloWorld.cxx
HelloWorld.cxx
#include <iostream>

int main(int argc, char * argv [])
{
  std::cout << "Hello World!" << std::endl;
  return 0;
}

```



# Slicer3 Execution Model

---

- The **Execution Model** which provides a mechanism for incorporating command line programs as Slicer modules.
- The Slicer modules are described using **XML files** which are used to generate
  - the C++ command line code
  - the Graphical User Interface (GUI).



# Modifying CMakeLists.txt

Open the file **CMakeLists.txt** located in the directory  
**/HelloWorld\_Plugin/HelloWorld/**

```
project(HelloWorld)

cmake_minimum_required(VERSION 2.6)
```

```
# Slicer3
find_package(Slicer3 REQUIRED)
include(${Slicer3_USE_FILE})
```

```
# Default install prefix
slicer3_set_default_install_prefix_for_external_projects()
```

```
#####
# Hello World plugin

#####
```

```
#####
# Tests for the plugin

enable_testing()

#####
```



# Editing CMakeLists.txt – part 1

Add the following lines to CMakeLists.txt

**set (CLP HelloWorld)**

**set (\${CLP}\_SOURCE \${CLP}.cxx)**

**generateclp(\${CLP}\_SOURCE \${CLP}.xml)**

```
project(HelloWorld)
```

```
cmake_minimum_required(VERSION 2.8.12)
```

```
# Slicer3
```

```
find_package(Slicer3 REQUIRED)
```

```
include(${Slicer3_USE_FILE})
```

```
# Default install prefix
```

```
sllicer3_set_default_install_prefix()
```

```
#####
```

```
# Hello World plugin
```

```
set (CLP HelloWorld)
```

```
set (${CLP}_SOURCE ${CLP}.cxx)
```

```
generateclp(${CLP}_SOURCE ${CLP}.xml)
```

```
#####
```

```
#####
```

```
# Tests for the plugin
```

```
enable_testing()
```

*GENERATECLP generates the file **HelloWorldCLP.h** for parsing the command line arguments.*

*‘CLP’ means Command Line Processing*



# Editing CMakeLists.txt – part 2

Add the following lines to CMakeLists.txt after the 'generateclp' line you just added

**add\_executable(\${CLP} \${\${CLP}\_SOURCE})**

**slicer3\_set\_plugins\_output\_path(\${CLP})**

**target\_link\_libraries (\${CLP} \${ITK\_LIBRARIES})**

```
#####  
# Hello World plugin  
  
set (CLP HelloWorld)  
set (${CLP}_SOURCE ${CLP}.cxx)  
generateclp(${CLP}_SOURCE ${CLP}.xml)  
  
add_executable(${CLP} ${${CLP}_SOURCE})  
slicer3_set_plugins_output_path(${CLP})  
target_link_libraries (${CLP} ${ITK_LIBRARIES})  
  
#####  
  
#####  
# Tests for the plugin  
  
enable_testing()  
add_test(NAME ${CLP}_TEST COMMAND ${CLP})
```

Save the file after editing

*ADD\_EXECUTABLE creates the stand-alone executable **HelloWorld.exe** that can be run from a command line.*



# Configuring HelloWorld - WINDOWS (1/5)

---

- Launch the CMake executable located in the directory Slicer3-lib/CMake-build/bin

A screenshot of a Windows command prompt window. The title bar is blue and contains the text "/cygdrive/c/slicer3dev/Slicer3-lib/CMake-build/bin". The window has standard minimize, maximize, and close buttons. The command prompt shows the user "Sonia PUJOL@SONIA-DELL-M70" at the prompt "~". The user enters the command "\$ cd c:slicer3dev/Slicer3-lib/CMake-build/bin/" and then "\$ ./CMakeSetup.exe". The prompt is followed by a cursor.

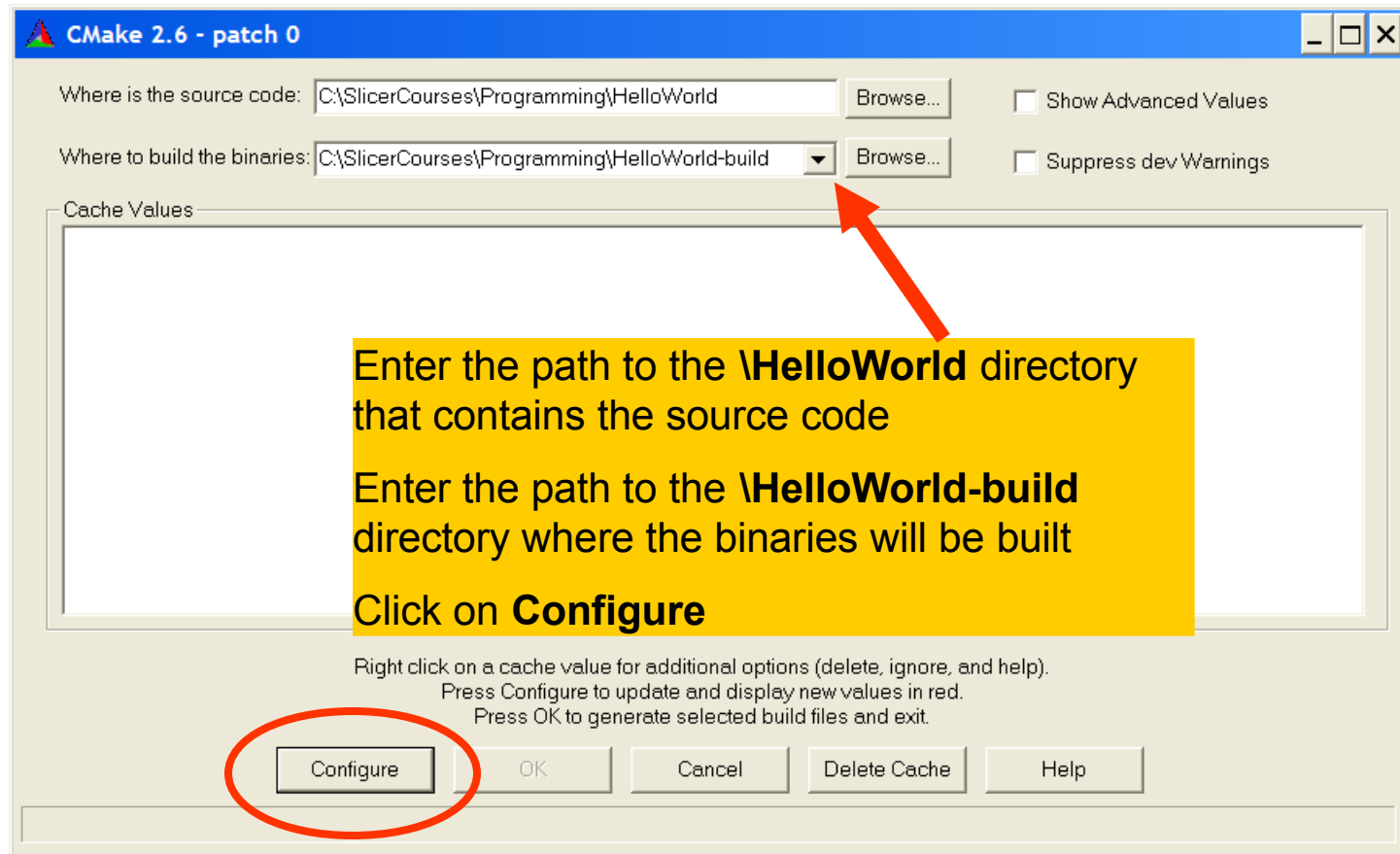
```
/cygdrive/c/slicer3dev/Slicer3-lib/CMake-build/bin

Sonia PUJOL@SONIA-DELL-M70 ~
$ cd c:slicer3dev/Slicer3-lib/CMake-build/bin/

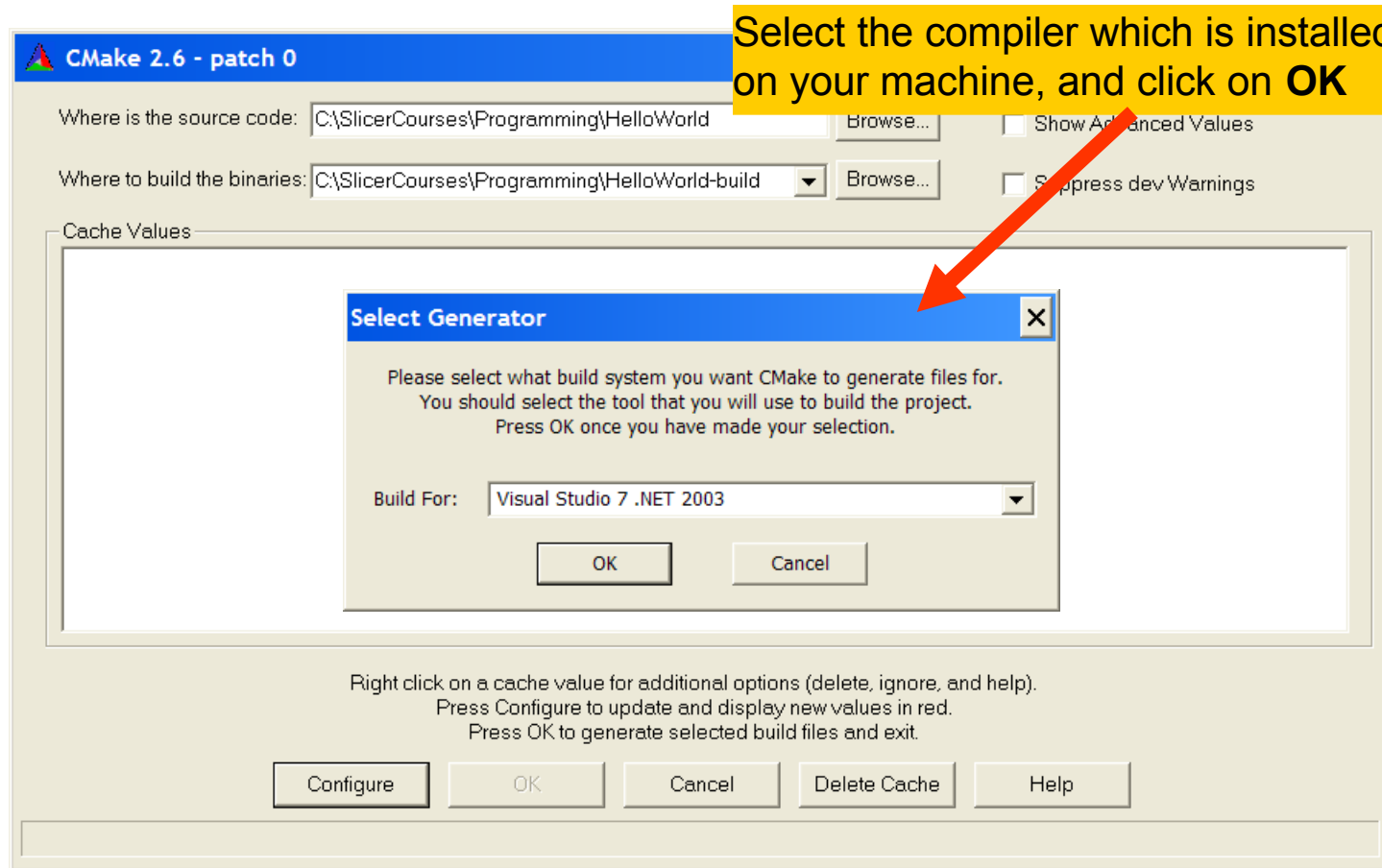
Sonia PUJOL@SONIA-DELL-M70 /cygdrive/c/slicer3dev/Slicer3-lib/CMake-build/bin
$ ./CMakeSetup.exe
```



# Configuring HelloWorld - WINDOWS (2/5)

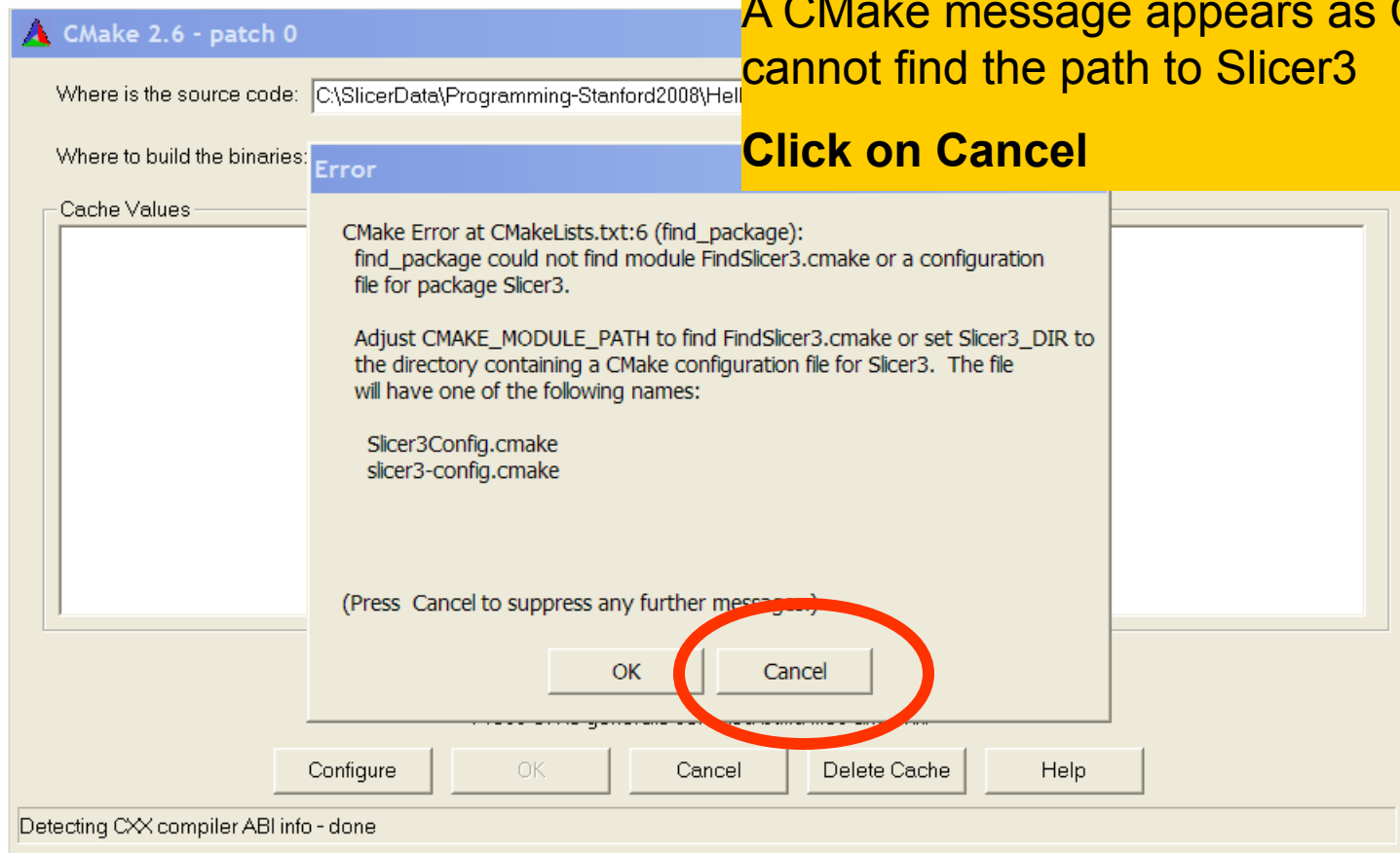


# Configuring HelloWorld - WINDOWS (3/5)





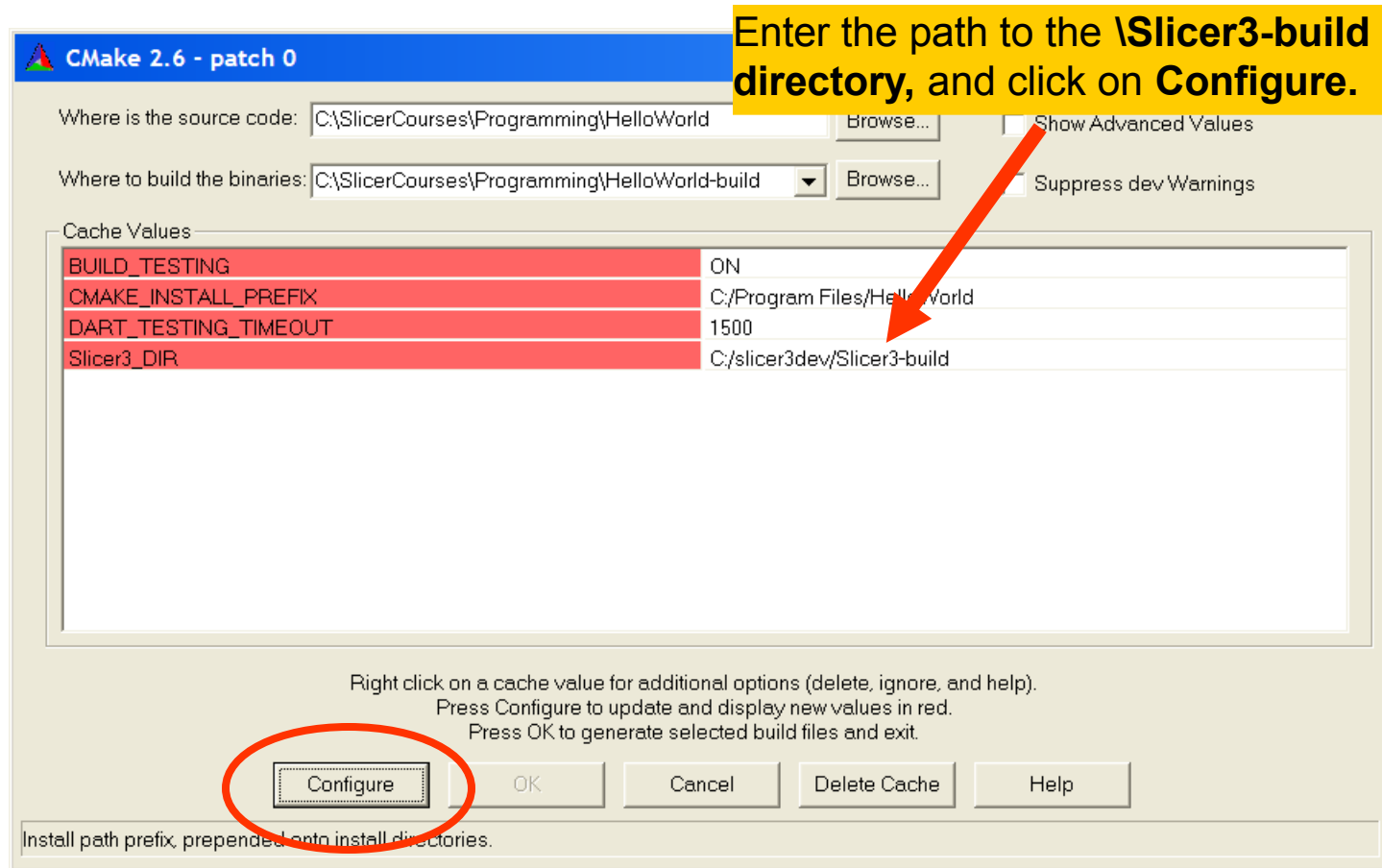
# Configuring HelloWorld - WINDOWS (3/5)



A CMake message appears as CMake cannot find the path to Slicer3

**Click on Cancel**

# Configuring HelloWorld - WINDOWS (4/5)





# Configuring HelloWorld - WINDOWS (5/5)

Click on OK to generate the build files in the **HelloWorld-build** directory, and exit

**CMake 2.6 - patch 0**

Where is the source code: C:\SlicerCourses\Programming\HelloWorld

Where to build the binaries: C:\SlicerCourses\Programming\HelloWorld-build Browse... ☐ Suppress dev Warnings

Cache Values

|                      |                             |
|----------------------|-----------------------------|
| BUILD_TESTING        | ON                          |
| CMAKE_INSTALL_PREFIX | C:/Program Files/HelloWorld |
| DART_TESTING_TIMEOUT | 1500                        |
| Slicer3_DIR          | C:/slicer3dev/Slicer3-build |

Right click on a cache value for additional options (delete, ignore, and help).  
Press Configure to update and display new values in red.  
Press OK to generate selected build files and exit.

Configure OK Cancel Delete Cache Help

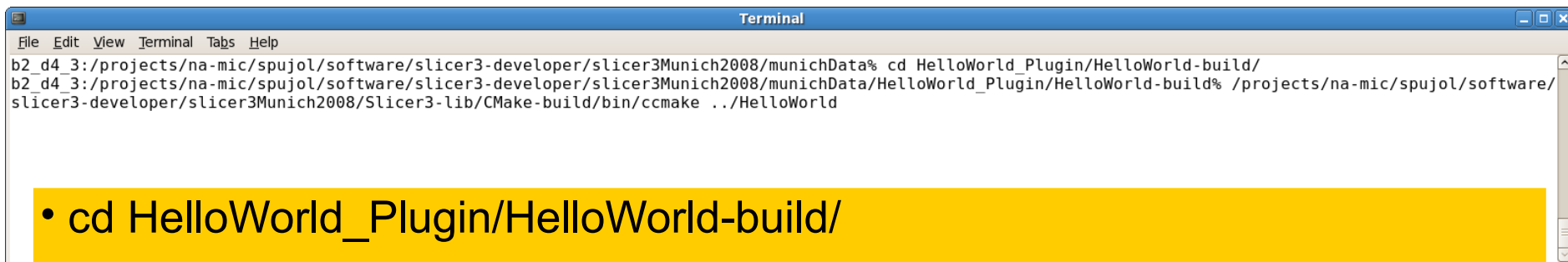
Build the testing tree.



# Configuring HelloWorld (Linux & Mac) 1/4

---

- From the **HelloWorld-build/** directory, launch the **ccmake** executable located in the **Slicer3-lib/CMake-build/bin/** directory

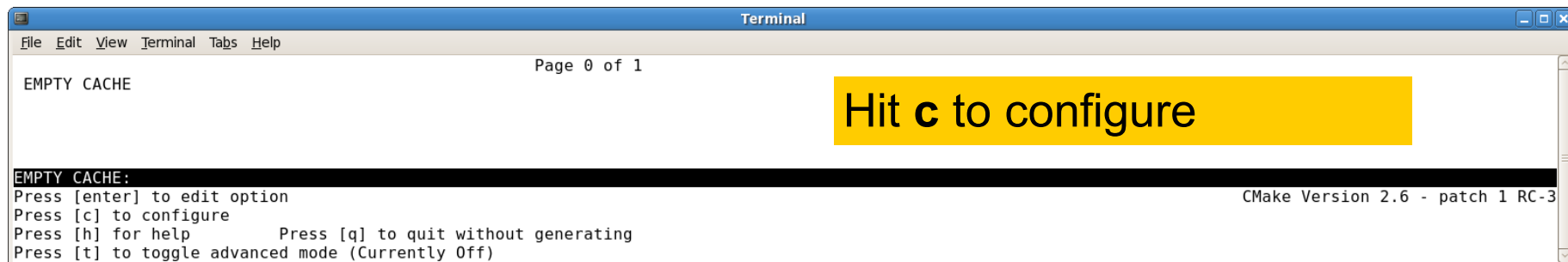
A screenshot of a terminal window titled "Terminal". The window shows a command prompt with the following text:

```
b2_d4_3:/projects/na-mic/spujol/software/slicer3-developer/slicer3Munich2008/munichData% cd HelloWorld_Plugin/HelloWorld-build/  
b2_d4_3:/projects/na-mic/spujol/software/slicer3-developer/slicer3Munich2008/munichData/HelloWorld_Plugin/HelloWorld-build% /projects/na-mic/spujol/software/  
slicer3-developer/slicer3Munich2008/Slicer3-lib/CMake-build/bin/ccmake ../HelloWorld
```

- `cd HelloWorld_Plugin/HelloWorld-build/`
- `/path/to/Slicer/build/Slicer3-lib/CMake-build/bin/ccmake ../HelloWorld`



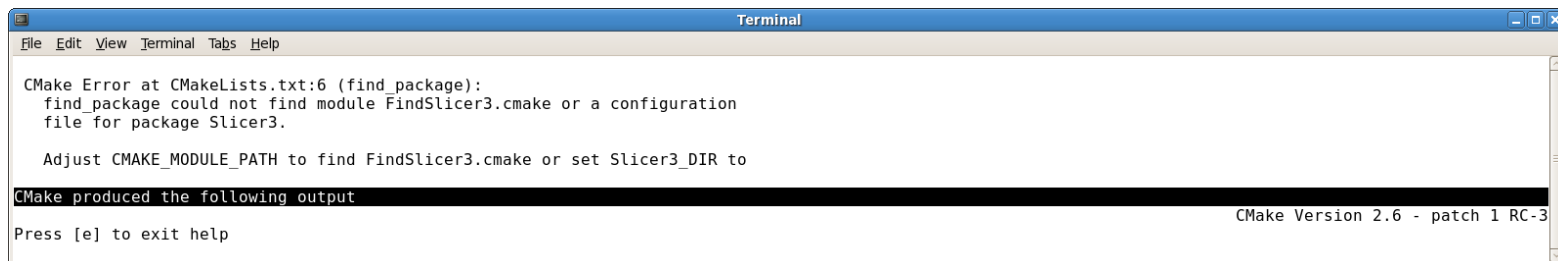
# Configuring HelloWorld (Linux & Mac) 2/4



A terminal window titled "Terminal" with a menu bar (File, Edit, View, Terminal, Tabs, Help) and a status bar (Page 0 of 1). The main text area shows "EMPTY CACHE" and a yellow box with the text "Hit c to configure". Below this, a black bar contains the text "EMPTY CACHE:". The terminal then displays the following instructions: "Press [enter] to edit option", "Press [c] to configure", "Press [h] for help", and "Press [t] to toggle advanced mode (Currently Off)". On the right side, it says "CMake Version 2.6 - patch 1 RC-3".

```
File Edit View Terminal Tabs Help
Page 0 of 1
EMPTY CACHE
Hit c to configure
EMPTY CACHE:
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [t] to toggle advanced mode (Currently Off)
CMake Version 2.6 - patch 1 RC-3
```

You need to enter the path to Slicer3 manually:  
Press **e** to get to the configuration options



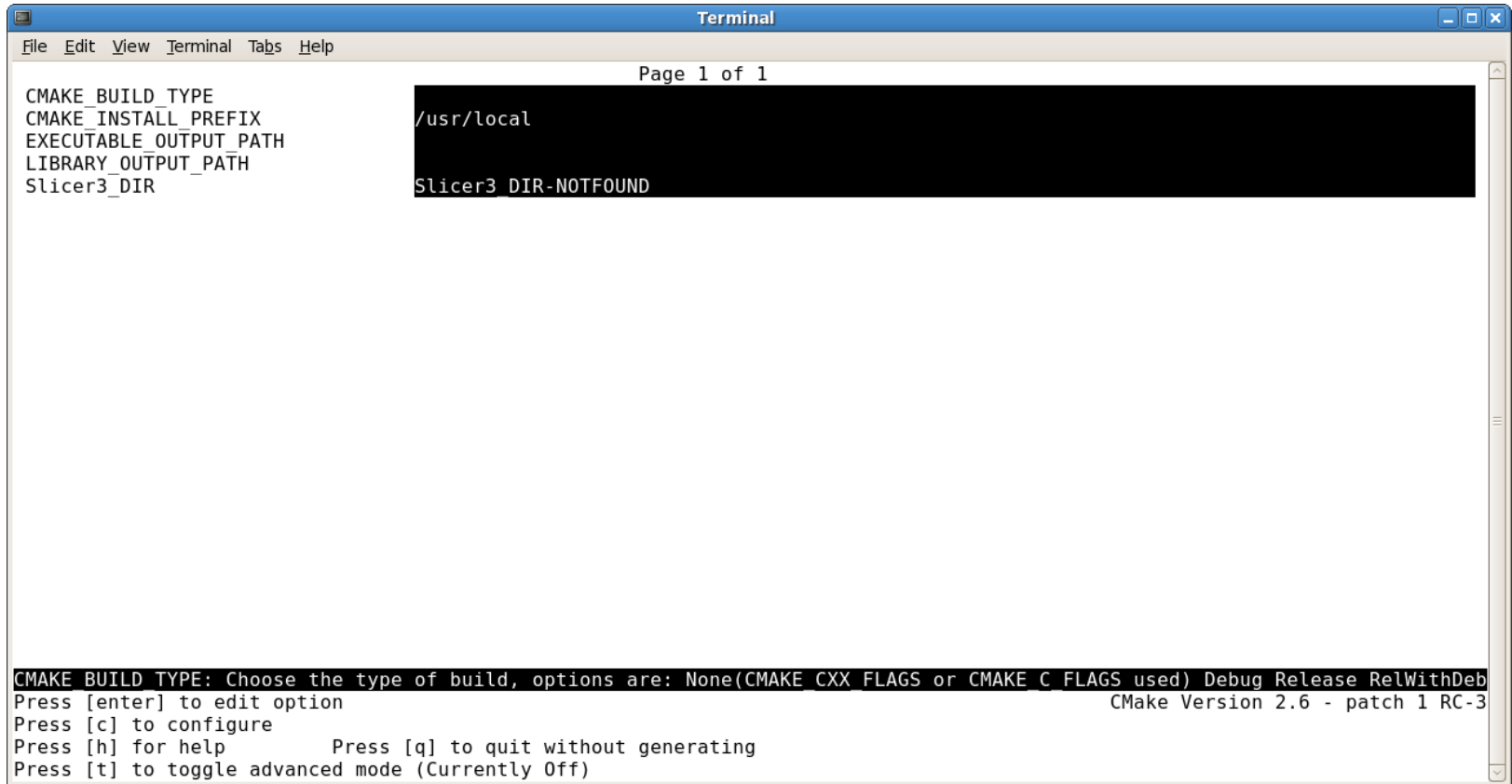
A terminal window titled "Terminal" with a menu bar (File, Edit, View, Terminal, Tabs, Help) and a status bar (CMake Version 2.6 - patch 1 RC-3). The main text area shows a CMake error message: "CMake Error at CMakeLists.txt:6 (find\_package): find\_package could not find module FindSlicer3.cmake or a configuration file for package Slicer3." Below this, it says "Adjust CMAKE\_MODULE\_PATH to find FindSlicer3.cmake or set Slicer3\_DIR to" and "CMake produced the following output". At the bottom, it says "Press [e] to exit help".

```
File Edit View Terminal Tabs Help
CMake Error at CMakeLists.txt:6 (find_package):
  find_package could not find module FindSlicer3.cmake or a configuration
  file for package Slicer3.

Adjust CMAKE_MODULE_PATH to find FindSlicer3.cmake or set Slicer3_DIR to
CMake produced the following output
Press [e] to exit help
CMake Version 2.6 - patch 1 RC-3
```



# Configuring HelloWorld (Linux & Mac) 3/4

A screenshot of a terminal window titled "Terminal". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The main content area shows a list of CMake configuration options: "CMAKE\_BUILD\_TYPE", "CMAKE\_INSTALL\_PREFIX", "EXECUTABLE\_OUTPUT\_PATH", "LIBRARY\_OUTPUT\_PATH", and "Slicer3\_DIR". The "Slicer3\_DIR" option is currently set to "/usr/local" and is highlighted with a black background. Below the list, there is a message: "CMAKE\_BUILD\_TYPE: Choose the type of build, options are: None(CMAKE\_CXX\_FLAGS or CMAKE\_C\_FLAGS used) Debug Release RelWithDeb". At the bottom of the terminal, there are instructions: "Press [enter] to edit option", "Press [c] to configure", "Press [h] for help", "Press [q] to quit without generating", and "Press [t] to toggle advanced mode (Currently Off)". The terminal also shows "CMake Version 2.6 - patch 1 RC-3".

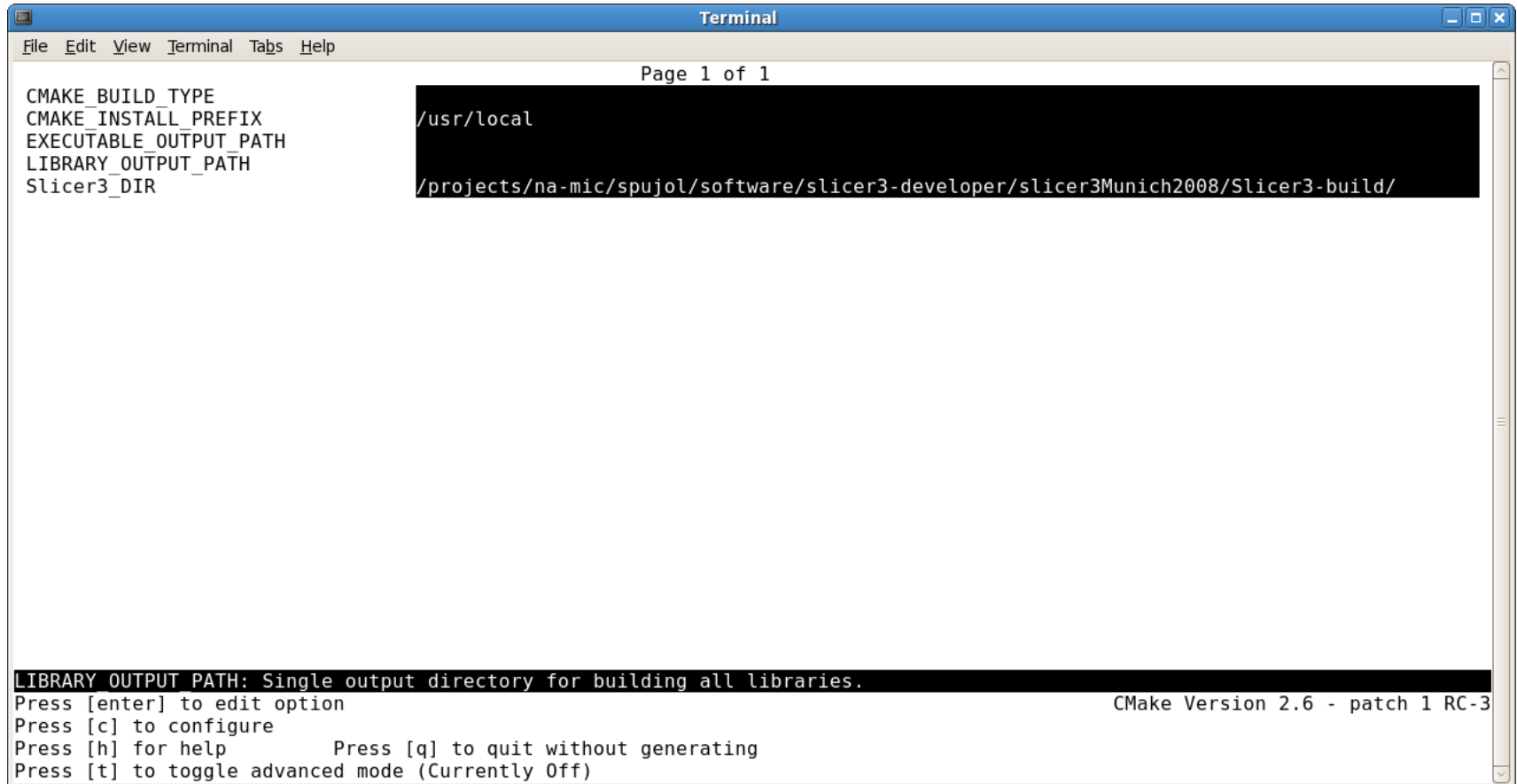
```
Terminal
File Edit View Terminal Tabs Help
Page 1 of 1
CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX
EXECUTABLE_OUTPUT_PATH
LIBRARY_OUTPUT_PATH
Slicer3_DIR
/usr/local
Slicer3_DIR-NOTFOUND
CMAKE_BUILD_TYPE: Choose the type of build, options are: None(CMAKE_CXX_FLAGS or CMAKE_C_FLAGS used) Debug Release RelWithDeb
Press [enter] to edit option
Press [c] to configure
Press [h] for help
Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)
CMake Version 2.6 - patch 1 RC-3
```

Enter the path to the directory Slicer3-build/:

- Arrow down to the Slicer3\_DIR and Hit Enter to edit the path
- Arrow up once you have finished editing the path



# Configuring HelloWorld (Linux & Mac) 4/4

A screenshot of a terminal window titled "Terminal". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The main content area shows the output of the CMake configuration process. It lists several options: CMAKE\_BUILD\_TYPE, CMAKE\_INSTALL\_PREFIX (set to /usr/local), EXECUTABLE\_OUTPUT\_PATH, LIBRARY\_OUTPUT\_PATH, and Slicer3\_DIR (set to /projects/na-mic/spujol/software/slicer3-developer/slicer3Munich2008/Slicer3-build/). At the bottom, there is a summary of the LIBRARY\_OUTPUT\_PATH, instructions on how to edit options (press [enter]), configure (press [c]), get help (press [h]), quit (press [q]), or toggle advanced mode (press [t]). The CMake version is 2.6 - patch 1 RC-3.

```
Terminal
File Edit View Terminal Tabs Help

Page 1 of 1

CMAKE_BUILD_TYPE
CMAKE_INSTALL_PREFIX      /usr/local
EXECUTABLE_OUTPUT_PATH
LIBRARY_OUTPUT_PATH
Slicer3_DIR                /projects/na-mic/spujol/software/slicer3-developer/slicer3Munich2008/Slicer3-build/

LIBRARY OUTPUT PATH: Single output directory for building all libraries.
Press [enter] to edit option
Press [c] to configure
Press [h] for help          Press [q] to quit without generating
Press [t] to toggle advanced mode (Currently Off)

CMake Version 2.6 - patch 1 RC-3
```

Press C to configure  
Press C to configure again  
Press G to generate the **Makefile**



# HelloWorld.xml

## Module Description

```
<?xml version="1.0" encoding="utf-8"?>
<executable>
  <category>
    Demonstration </category>
  <title>
    Hello World </title>
  <description>
    Slicer Developer Course </description>
  <version>
    1.0 </version>
  <documentation-url>
    http://www.na-mic.org/Wiki/index.php/Slicer
  </license></license>
  <contributor>
    Sonia Pujol, Ph.D., Surgical Planning Laboratory, Harvard Medical School </contributor>
  <acknowledgements>
    This work is part of the National Alliance for Medical Image Computing (NAMIC),
    funded by the National Institutes of Health through the NIH Roadmap for Medical Research,
    Grant U54 EB005149. </acknowledgements>

  <parameters>
    <label>Input/Output</label>
    <description>Input/output parameters</description>
    <image>
      <name>helloWorldInputVolume</name>
      <label>Input Volume</label>
      <channel>input</channel>
      <index>0</index>
      <default>None</default>
      <description>Input volume</description>
    </image>
    <image>
      <name>helloWorldOutputVolume</name>
      <label>Output Volume</label>
      <channel>output</channel>
      <index>1</index>
      <default>None</default>
      <description>Output filtered</description>
    </image>
  </parameters>
</executable>
```

Open the file **HelloWorld.xml**  
located in the directory  
**HelloWorld\_Plugin/HelloWorld**

## Module Parameters



# Module Description

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<executable>
```

```
<category>
```

```
Demonstration</category>
```

```
<title>
```

```
Hello World</title>
```

```
<description>
```

```
Slicer Developer Course</description>
```

```
<version>
```

```
1.0</version>
```

```
<documentation-url></documentation-url>
```

```
<license></license>
```

```
<contributor> Sonia Pujol, Ph.D., Surgical Planning Laboratory, Harvard Medical School </contributor>
```

```
<acknowledgements>
```

This work is part of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149.

```
</acknowledgements>
```

# Module Parameters

```
<parameters>
```

```
<label>Input/Output</label>
```

```
<description>Input/output parameters</description>
```

```
<image>
```

```
<name>helloWorldInputVolume</name>
```

```
<label>Input Volume</label>
```

```
<channel>input</channel>
```

```
<index>0</index>
```

```
<default>None</default>
```

```
<description>Input volume</description>
```

```
</image>
```

A file that  
specifies  
the image

```
<image>
```

```
<name>helloWorldOutputVolume</name>
```

```
<label>Output Volume</label>
```

```
<channel>output</channel>
```

```
<index>1</index>
```

```
<default>None</default>
```

```
<description>Output filtered</description>
```

```
</image>
```

```
</parameters>
```

```
</executable>
```

Input  
Volume

Output  
Volume



# Modifying the source code

---

Open the file HelloWorld.cxx

```
# include <iostream>

int main(int argc, char * argv [])
{

    std::cout<< "Hello World !"<<std::endl;

    return 0 ;
}
```



# Modifying the source code

Add the following lines to the file HelloWorld.cxx

```
# include <iostream>
```

```
#include "HelloWorldCLP.h"
```

```
int main(int argc, char * argv [])
```

```
{
```

```
PARSE_ARGS;
```

```
std::cout<< "Hello World !"<<std::endl;
```

```
return EXIT_SUCCESS ;
```

```
}
```



# Building HelloWorld.exe

---

## Mac/Linux

Run '**make**' in the directory **HelloWorld-build/**

## Windows

In Visual Studio, select **Build→Build Solution** to build the solution **HelloWorld.sln** located in **HelloWorld-build/**



# Building HelloWorld.exe

---

## Mac/Linux

**HelloWorld.exe** is located in  
**/HelloWorld-build/lib/slicer3/plugins**

## Windows

**HelloWorld.exe** is located in  
**/HelloWorld-build/lib/slicer3/plugins/debug**



# Running Slicer3

---

## Mac/Linux

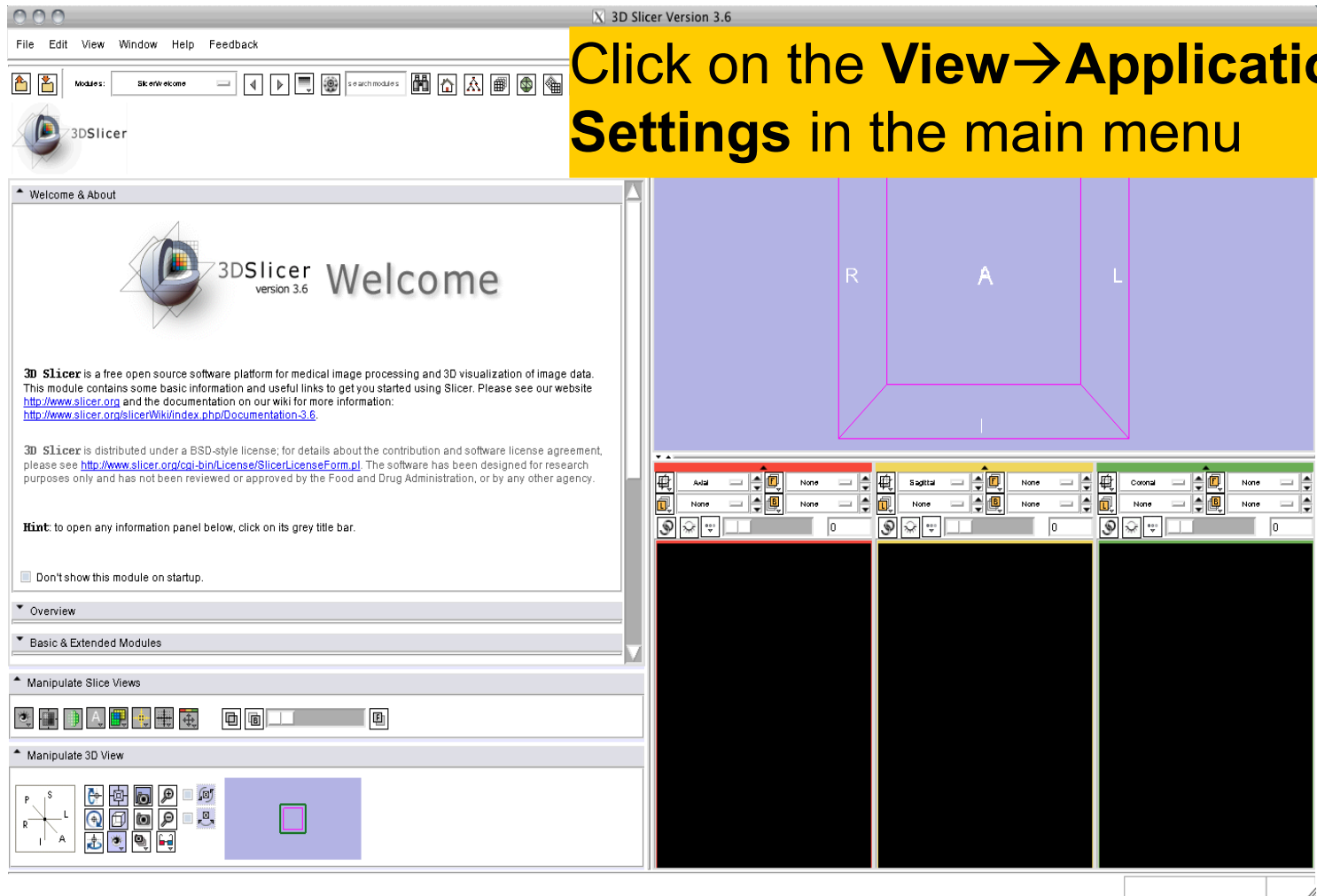
Run '**./Slicer3**' in Slicer3-build/

## Windows

Run '**./Slicer3**' in Slicer3-build/

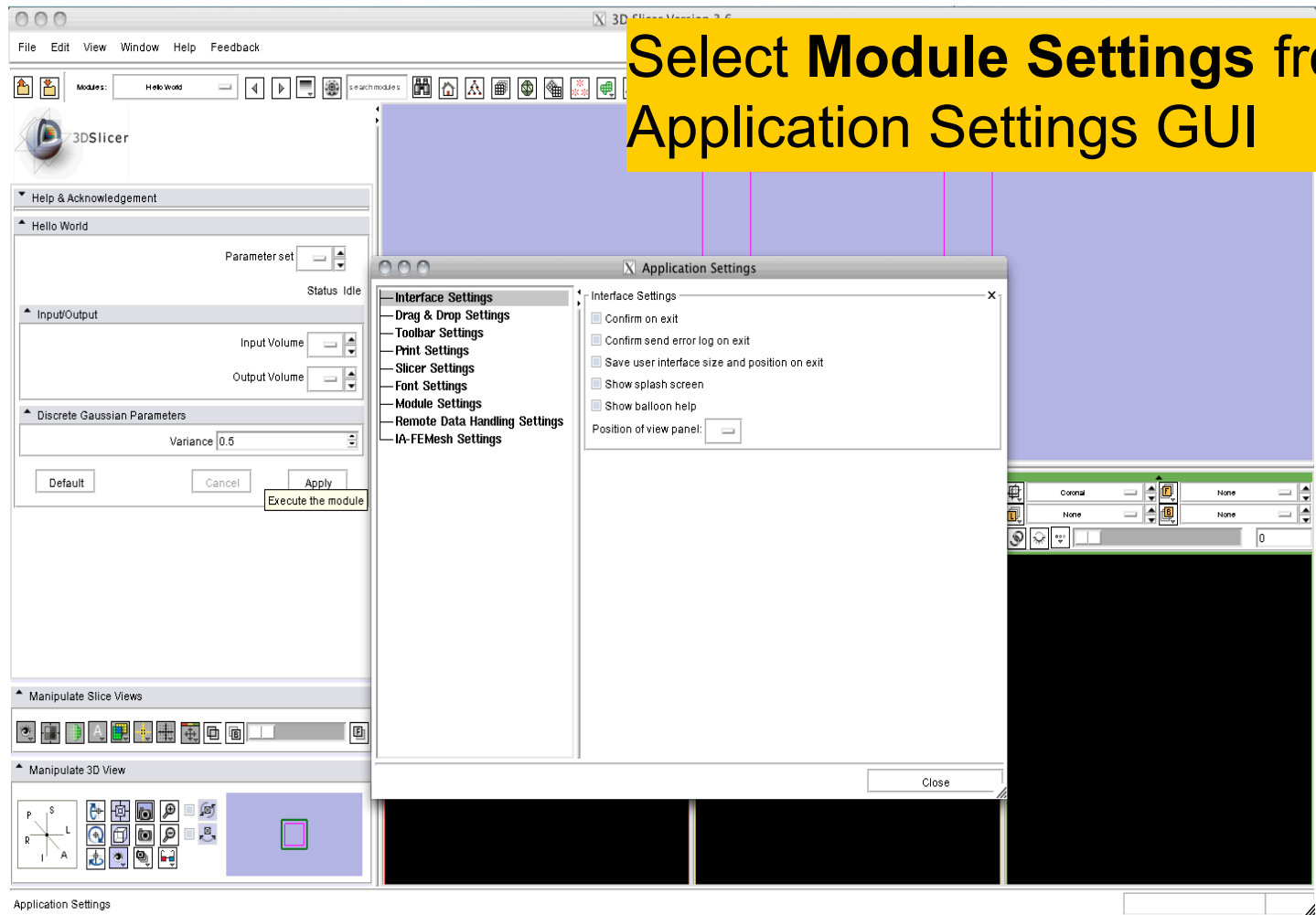


# Running Slicer3

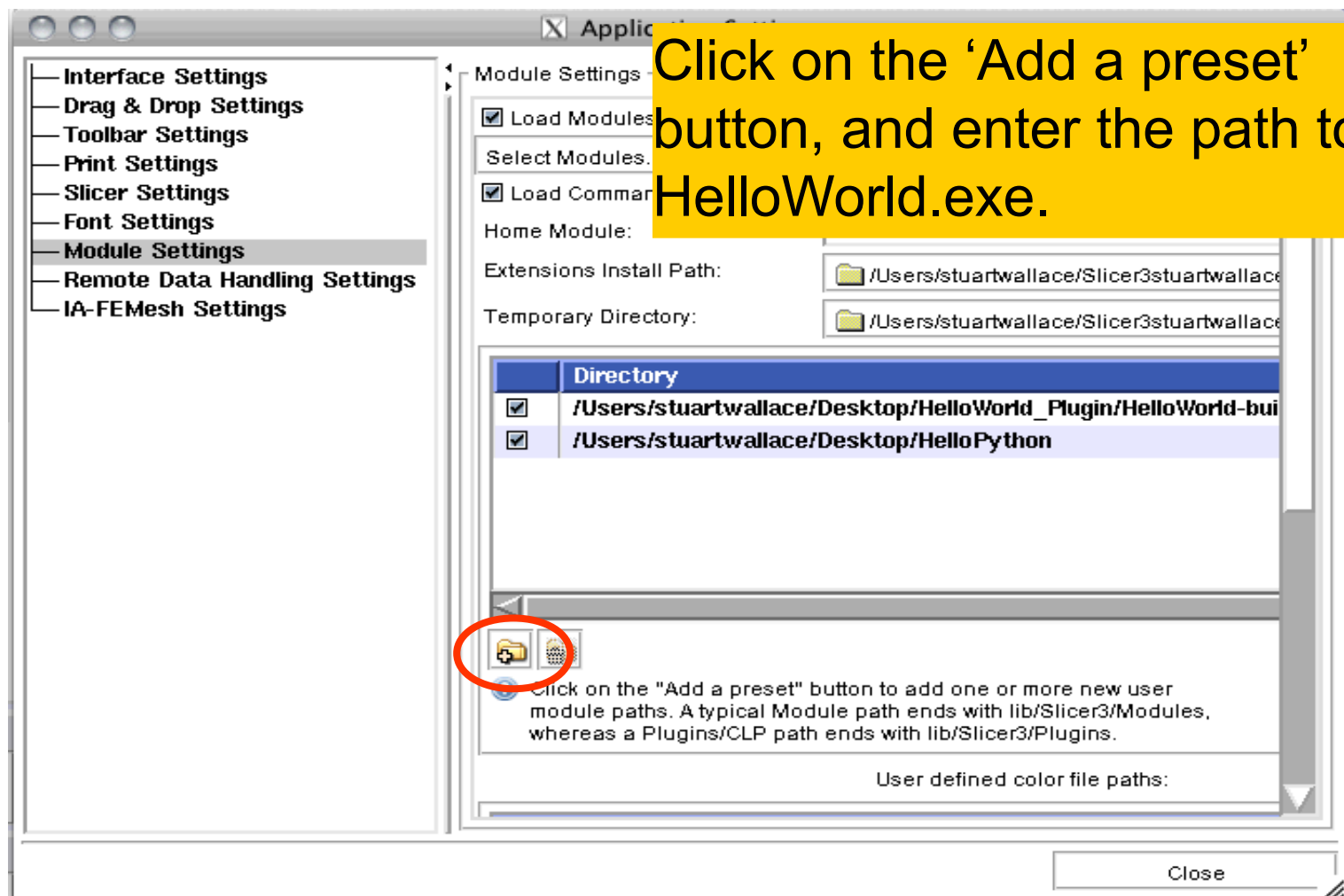




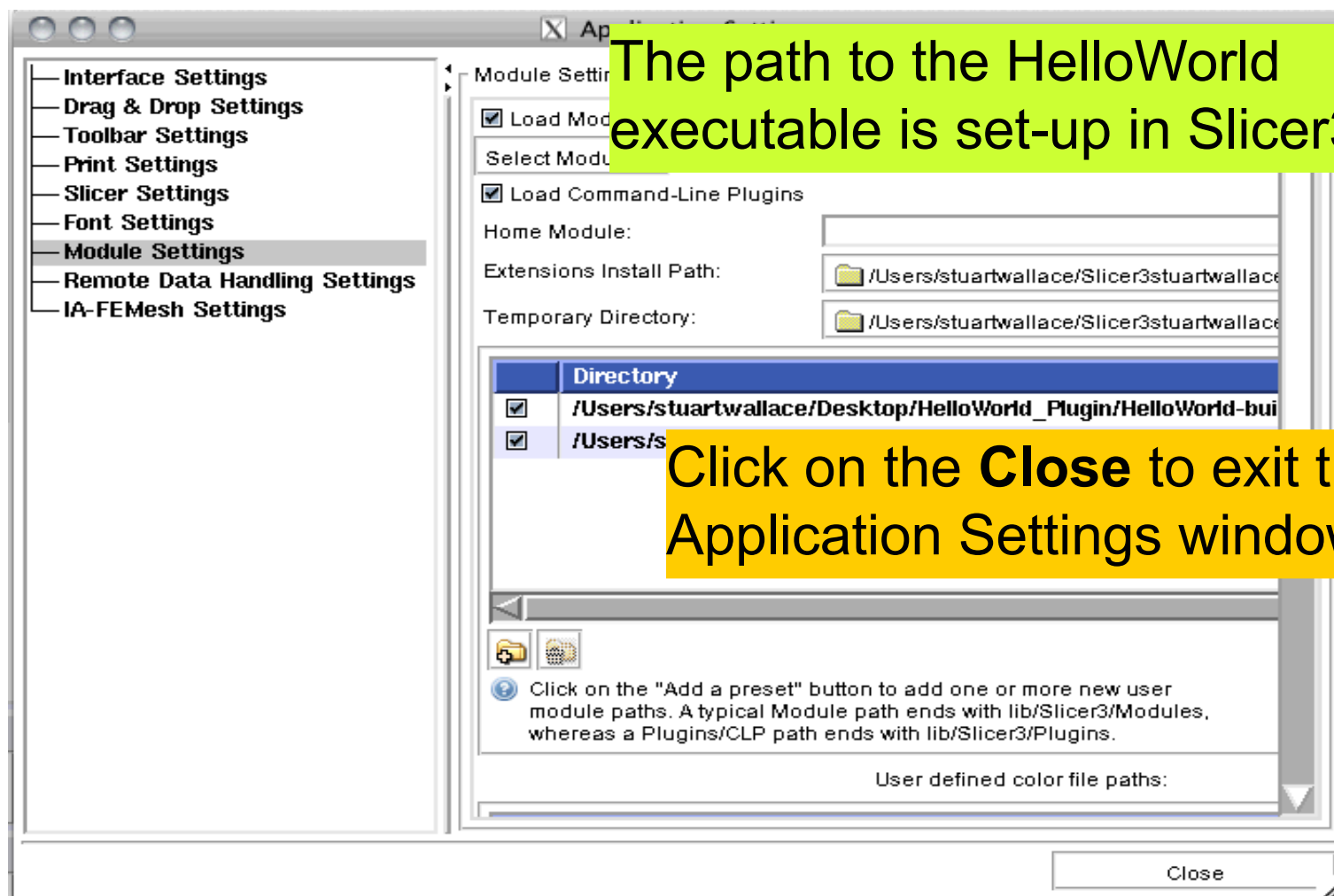
# Setting the HelloWorld plugin path



# Setting the HelloWorld plugin path

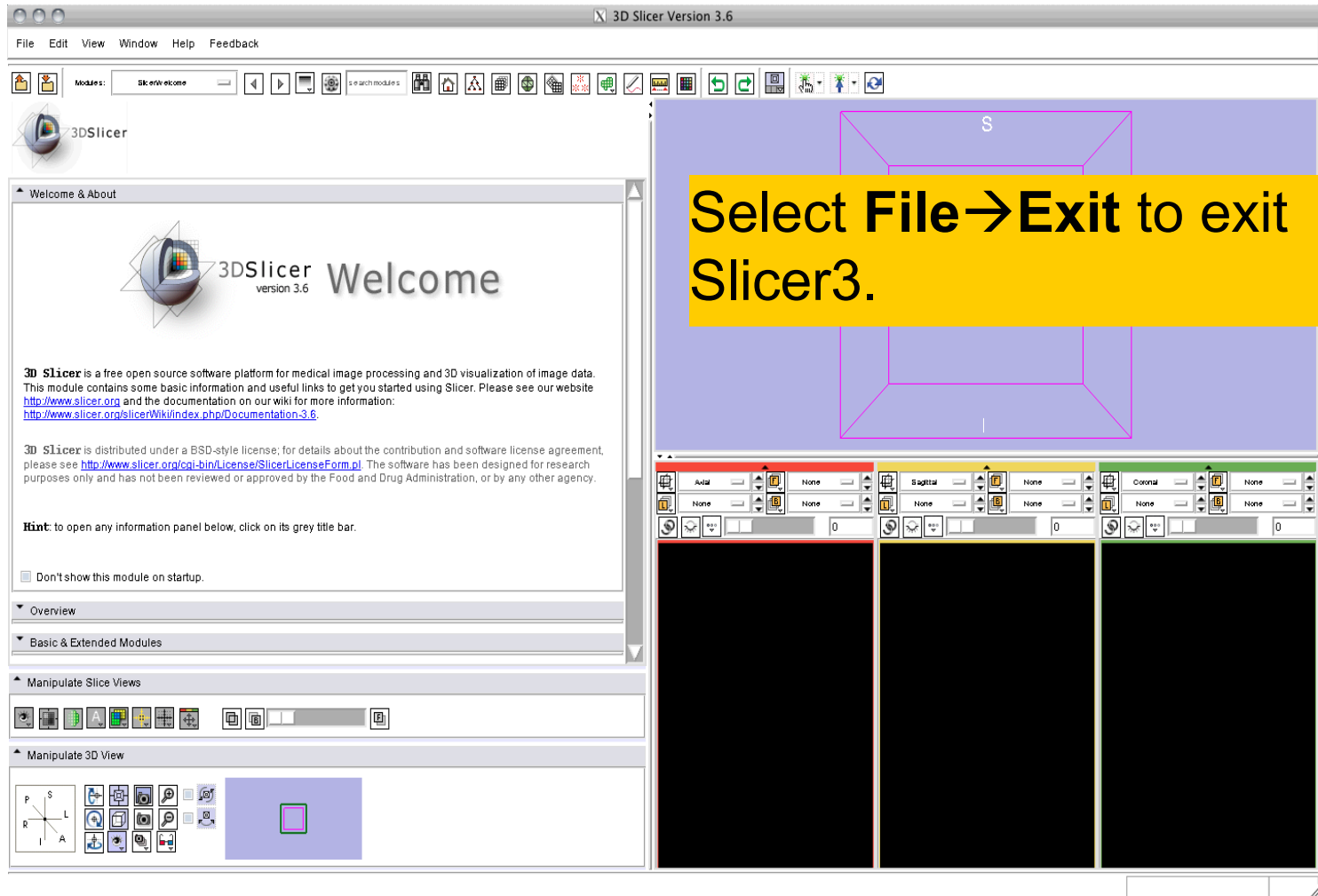


# Setting the HelloWorld plugin path





# Setting the HelloWorld plugin path





# Running Slicer3

---

## Mac/Linux

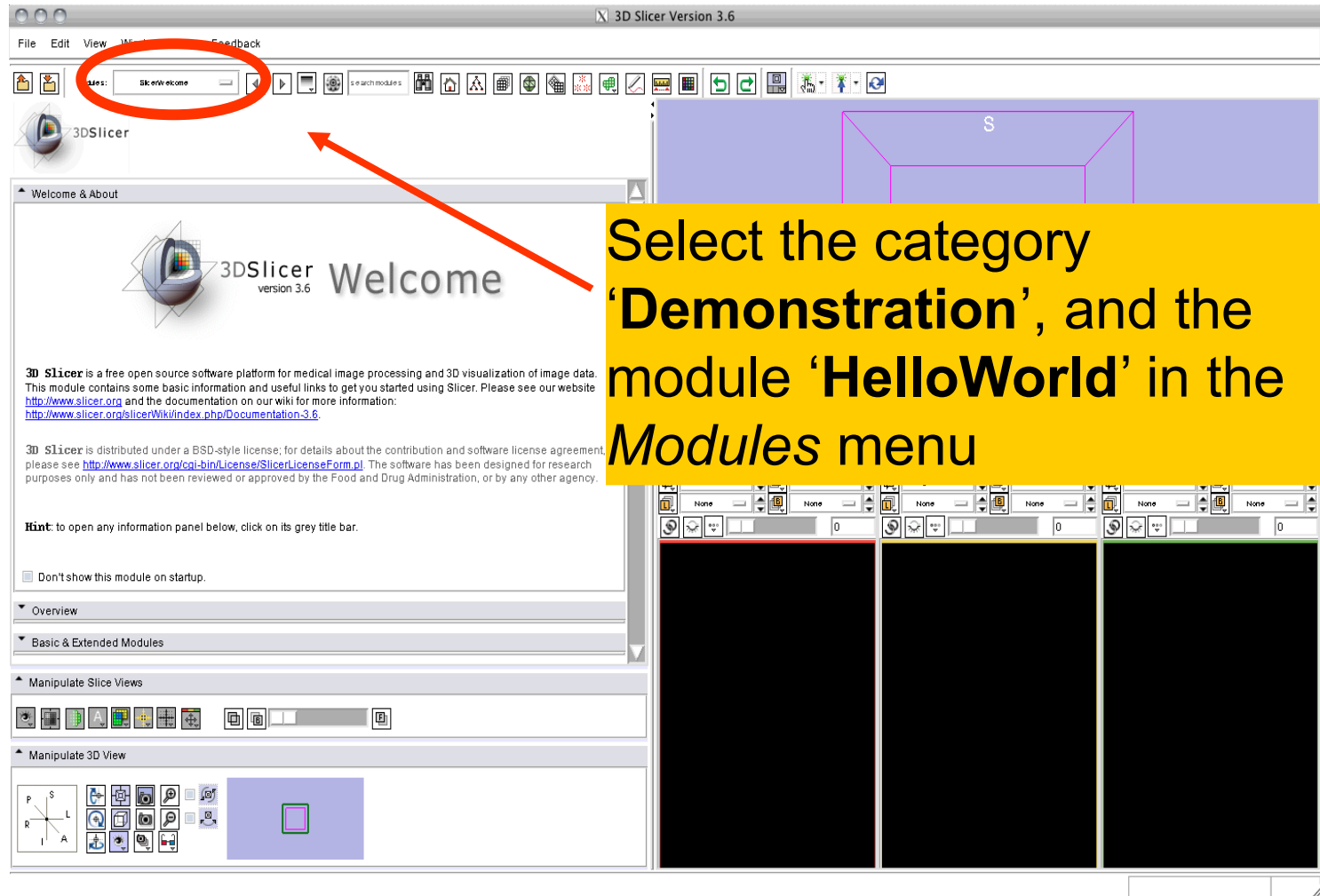
Run '**./Slicer3**' in Slicer3-build/

## Windows

Run '**./Slicer3.exe**' in Slicer3-build/

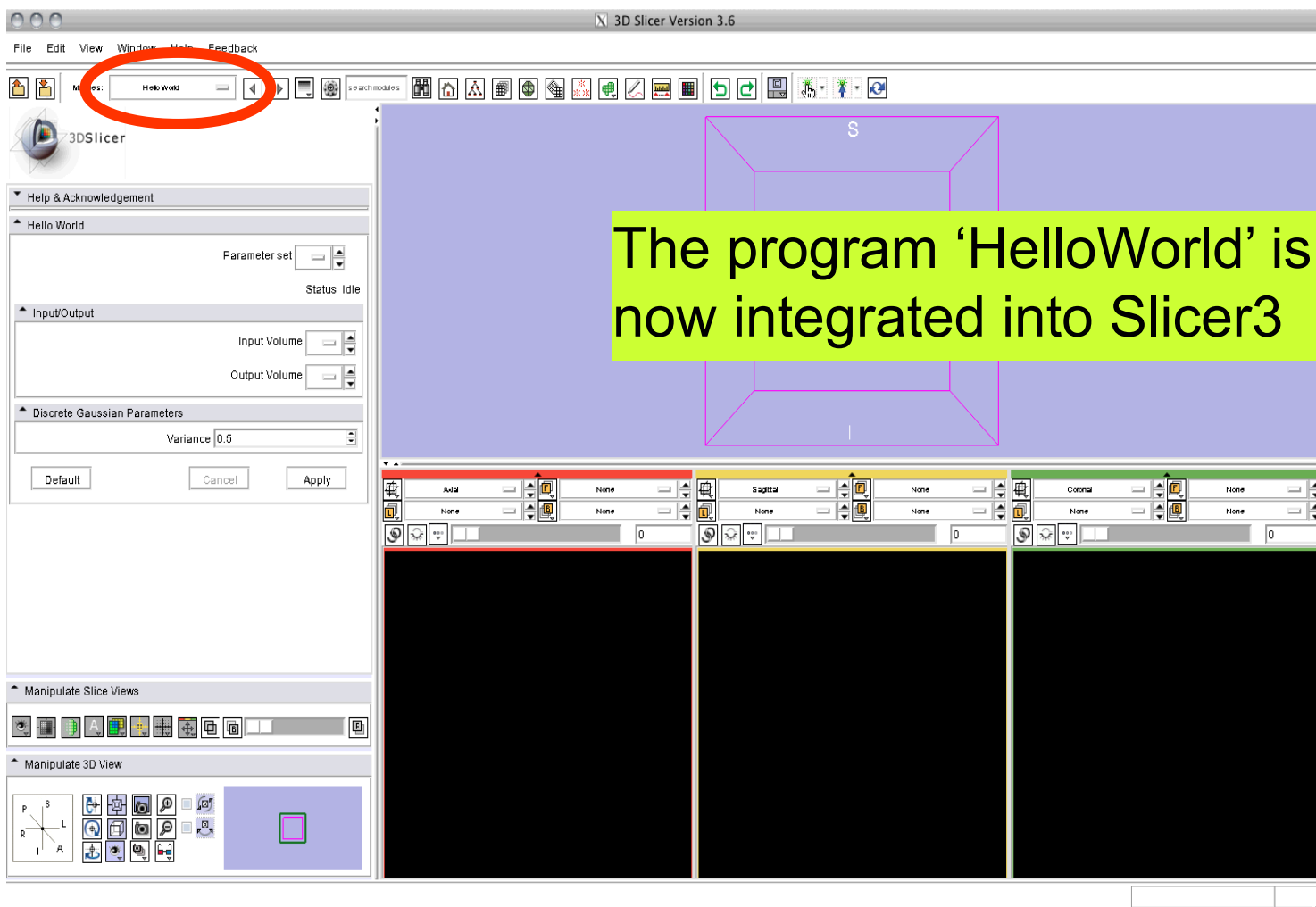


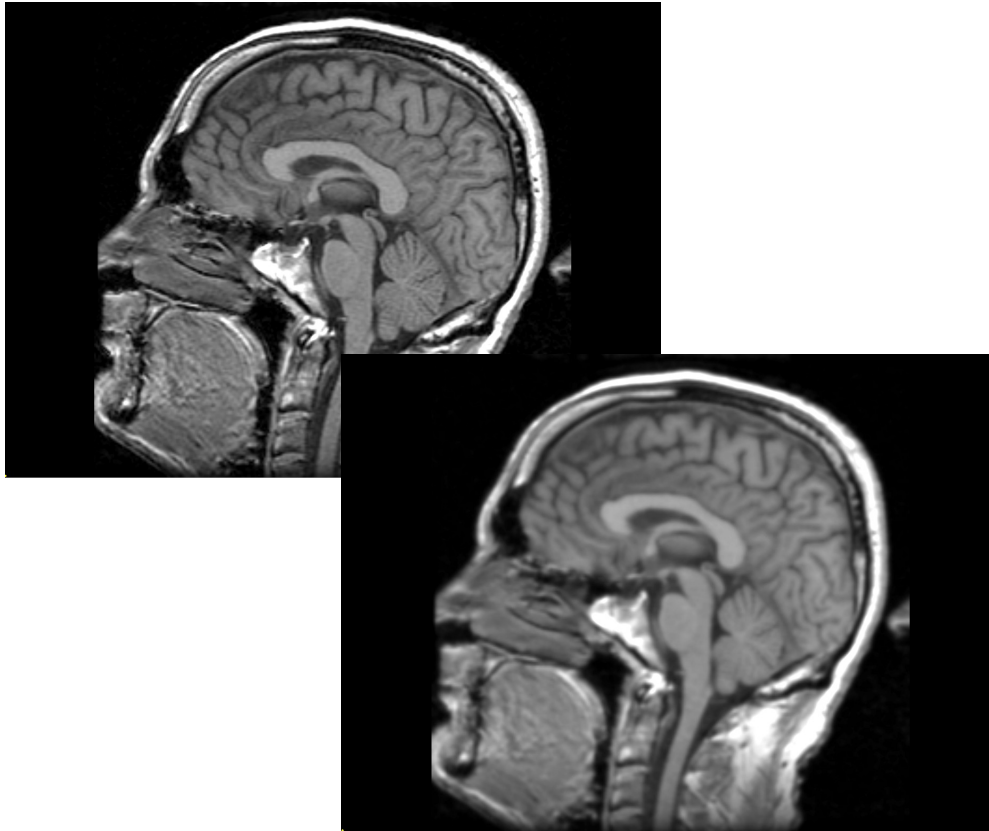
# HelloWorld module in Slicer3





# HelloWorld Module in Slicer3



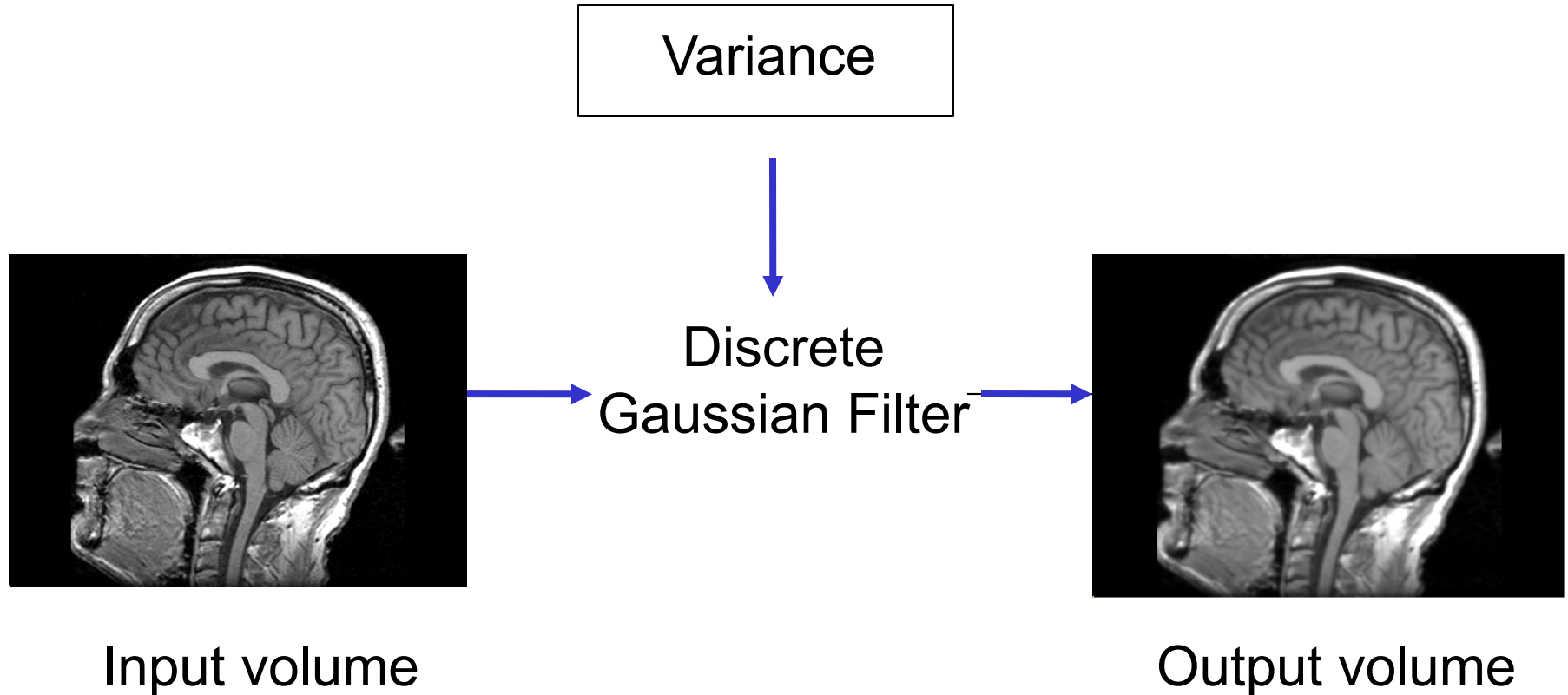


## Part B: Implementing an image filter

- In this section, we'll implement a **Gaussian smoothing operator** to 'blur' the images and remove detail and noise.
- This implementation will allow us to run the filter on volumes loaded in Slicer, and to integrate the resulting filtered volumes as MRML nodes.

# Discrete Gaussian Filter

---





# Editing the file HelloWorld.xml

```
<?xml version="1.0" encoding="utf-8"?>
<executable>
  <category>
    Demonstration</category>
  <title>
    Hello World</title>
  <description>
    Slicer Developer Example</description>
  <version>
    1.0</version>
  <documentation-url></documentation-url>
  <license></license>
  <contributor>
    Sonia Pujol, Ph.D, Surgical Planning Laboratory, Harvard Medical School </contributor>
  <acknowledgements>
    This work is part of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149. </acknowledgements>

  <parameters>
    <label>Input/Output</label>
    <description>Input/output parameters</description>
    ....
  </parameters>
```

Add a new parameter group to HelloWorld.xml



```
  <parameters>
    <label>Discrete Gaussian Parameters</label>
    <description>Parameters of the Discrete Gaussian Filter </description>
  </parameters>
```

```
</executable>
```

# Editing the file HelloWorld.xml

```
<parameters>
  <label>Discrete Gaussian Parameters</label>
  <description>Parameters of the Discrete Gaussian Filter </description>

  <double>
    <name>variance</name>
    <longflag>--variance</longflag>
    <description>Variance ( width of the filter kernel) </description>
    <label>Variance</label>
    <default>0.5</default>
  </double>

</parameters>
```



Add the parameter 'variance' which corresponds to the variance of the Discrete Gaussian Filter to HelloWorld.xml

# Implementing I/O functionalities

Add the following lines to HelloWorld.cxx

```
#include <iostream>
#include "HelloWorldCLP.h"
#include "itkImage.h"
#include "itkImageFileReader.h"
#include "itkImageFileWriter.h"
int main(int argc, char * argv [])
{
    PARSE_ARGS;
    std::cout << "Hello World!" << std::endl;
    return EXIT_SUCCESS ;
}
```

# Implementing I/O functionalities

Add the following command lines to set-up the reading and writing functionalities in the 'main' procedure in HelloWorld.cxx

```
int main ( int argc, char * argv[])
{
    PARSE_ARGS;
    std::cout << "Hello World!" << std::endl;

    typedef itk::Image<short,3>  ImageType;
    typedef itk::ImageFileReader<ImageType>  ReaderType;
    typedef itk::ImageFileWriter<ImageType>  WriterType;
    ReaderType::Pointer reader = ReaderType::New();
    WriterType::Pointer writer = WriterType::New();

    return EXIT_SUCCESS;
}
```

# Implementing I/O functionalities

Set the input and output volumes parameters defined in HelloWorld.xml

```
int main ( int argc, char * argv[])
{
    PARSE_ARGS;
    std::cout << "Hello World!" << std::endl;
    typedef itk::Image< short, 3 >  ImageType;
    typedef itk::ImageFileReader< ImageType >  ReaderType;
    typedef itk::ImageFileWriter< ImageType >  WriterType;
    ReaderType::Pointer reader = ReaderType::New();
    WriterType::Pointer writer = WriterType::New();

    reader->SetFileName(helloWorldInputVolume.c_str() );
    writer->SetFileName (helloWorldOutputVolume.c_str());

    return EXIT_SUCCESS;
}
```



# Implementing the filter in HelloWorld.cxx

## Implement the filter itk::DiscreteGaussianImageFilter

```
#include "itkDiscreteGaussianImageFilter.h"
```

```
int main ( int argc, char * argv[])
{
    PARSE_ARGS;
    std::cout << "Hello World!" << std::endl;
    typedef itk::Image< short, 3 > ImageType;
    typedef itk::ImageFileReader< ImageType > ReaderType;
    typedef itk::ImageFileWriter< ImageType > WriterType;
    ReaderType::Pointer reader = ReaderType::New();
    WriterType::Pointer writer = WriterType::New();
    reader->SetFileName( helloWorldInputVolume.c_str() );
    writer->SetFileName(helloWorldOutputVolume.c_str());
```

```
typedef itk::DiscreteGaussianImageFilter <ImageType, ImageType> FilterType;
FilterType::Pointer filter = FilterType::New();
```

```
return EXIT_SUCCESS;
}
```



# Implementing the filter in HelloWorld.cxx

```
int main ( int argc, char * argv[])
{
    PARSE_ARGS;
    std::cout << "Hello World!" << std::endl;
    typedef itk::Image< short, 3 > ImageType;
    typedef itk::ImageFileReader< ImageType > ReaderType;
    typedef itk::ImageFileWriter< ImageType > WriterType;
    ReaderType::Pointer reader = ReaderType::New();
    WriterType::Pointer writer = WriterType::New();
    reader->SetFileName( helloWorldInputVolume.c_str() );
    writer->SetFileName (helloWorldOutputVolume.c_str());
    typedef itk::DiscreteGaussianImageFilter <ImageType, ImageType> FilterType;
    FilterType::Pointer filter = FilterType::New();
    try {
        filter->SetInput(reader->GetOutput());
        filter->SetVariance(variance);
        writer->SetInput(filter->GetOutput());
        writer->Update();
    }
    catch (itk::ExceptionObject &excep){
        std::cerr << argv[0] << ": exception caught !" << std::endl;
        return EXIT_FAILURE;}
    return EXIT_SUCCESS;}
```

Add the following lines  
for the filter execution:





# Building HelloWorld

---

## Mac/Linux

Run **'make'** in the directory **HelloWorld-build/**

## Windows

Select **Build→Build Solution** to build the solution **HelloWorld.sln** located in **HelloWorld-build/**



# Running Slicer3

---

## Mac/Linux

Run '**./Slicer3**' in Slicer3-build/

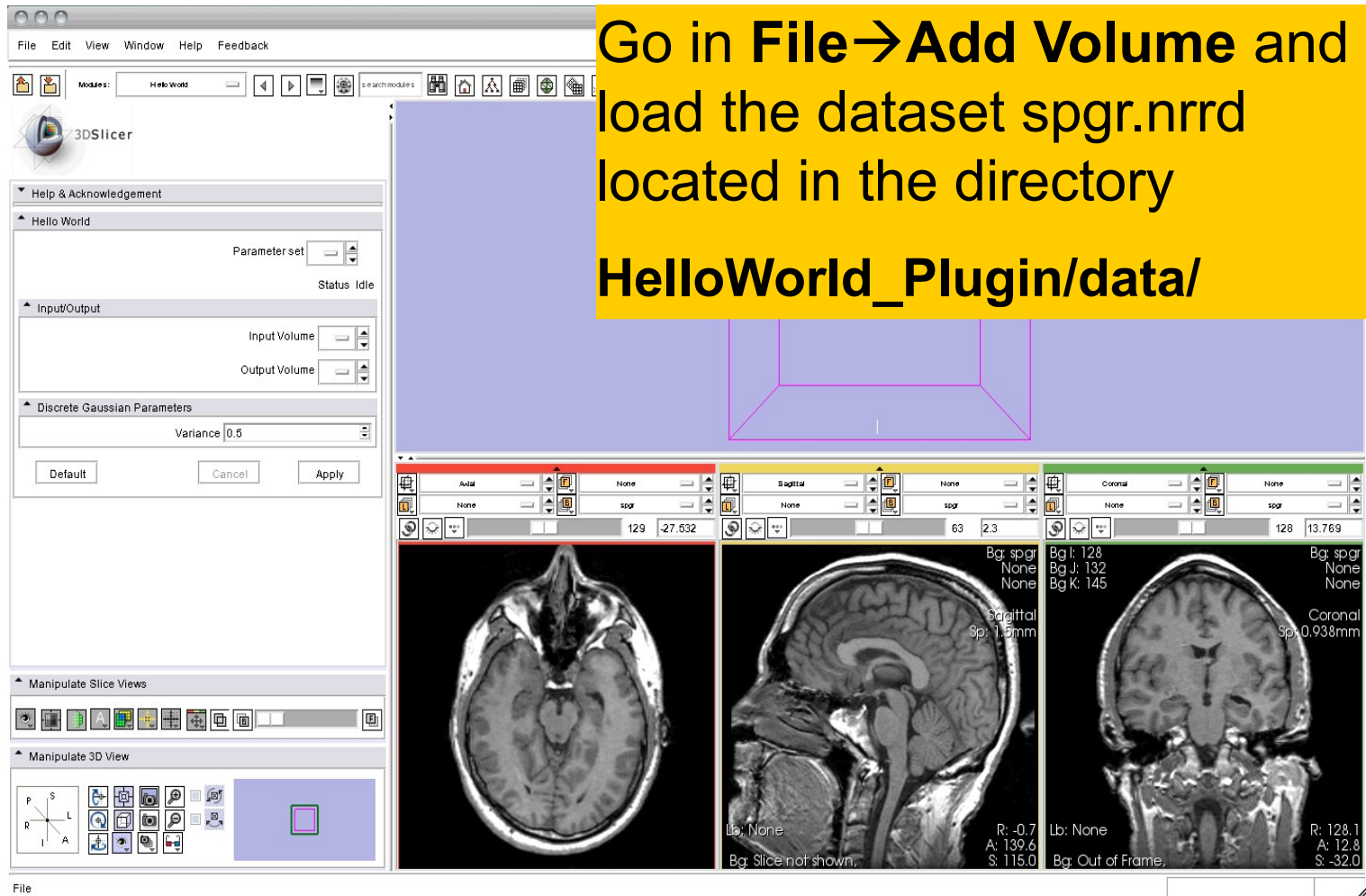
## Windows

Run '**./Slicer3.exe**' in Slicer3-build/

# Running the Filter

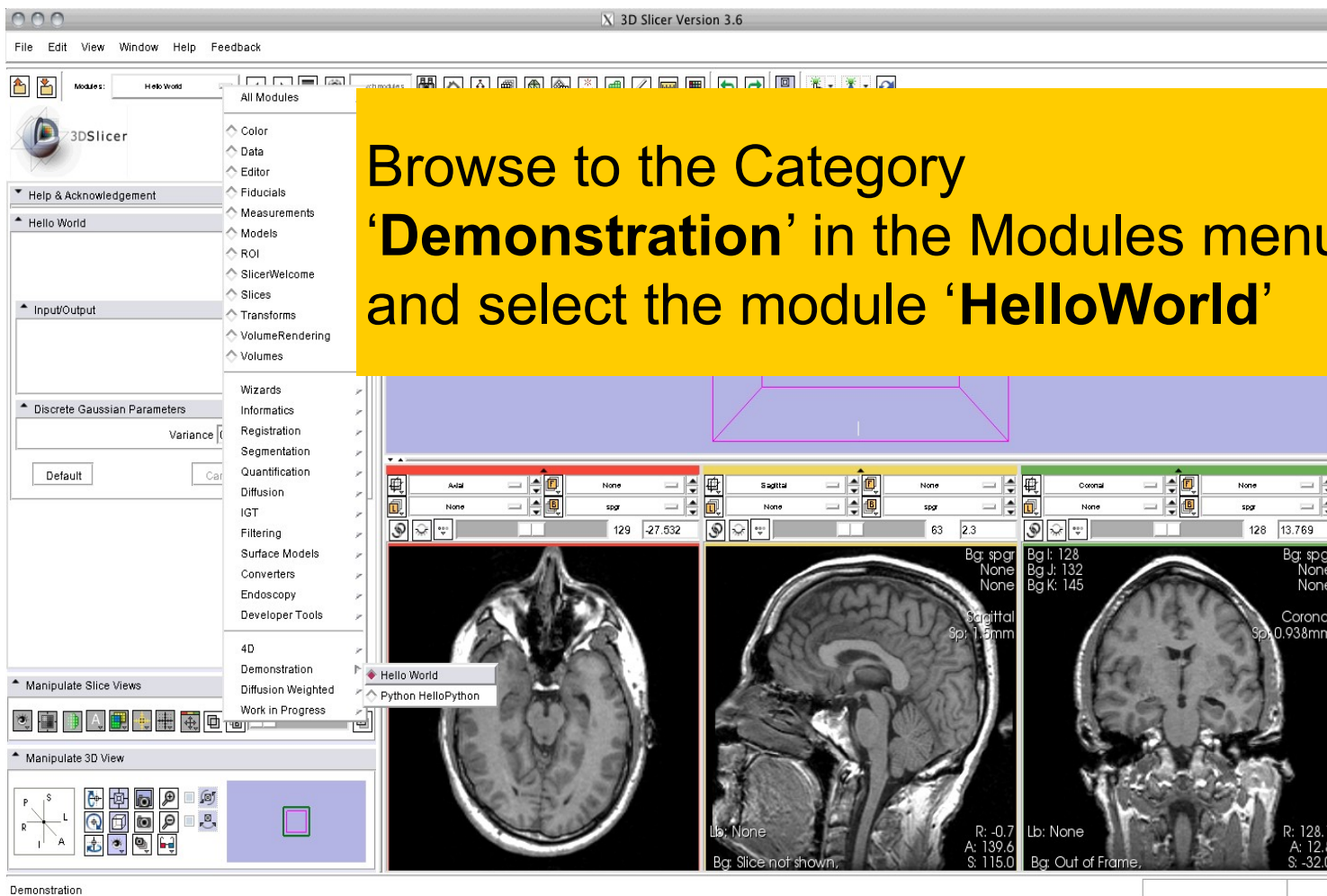
Go in **File**→**Add Volume** and load the dataset spgr.nrrd located in the directory

**HelloWorld\_Plugin/data/**

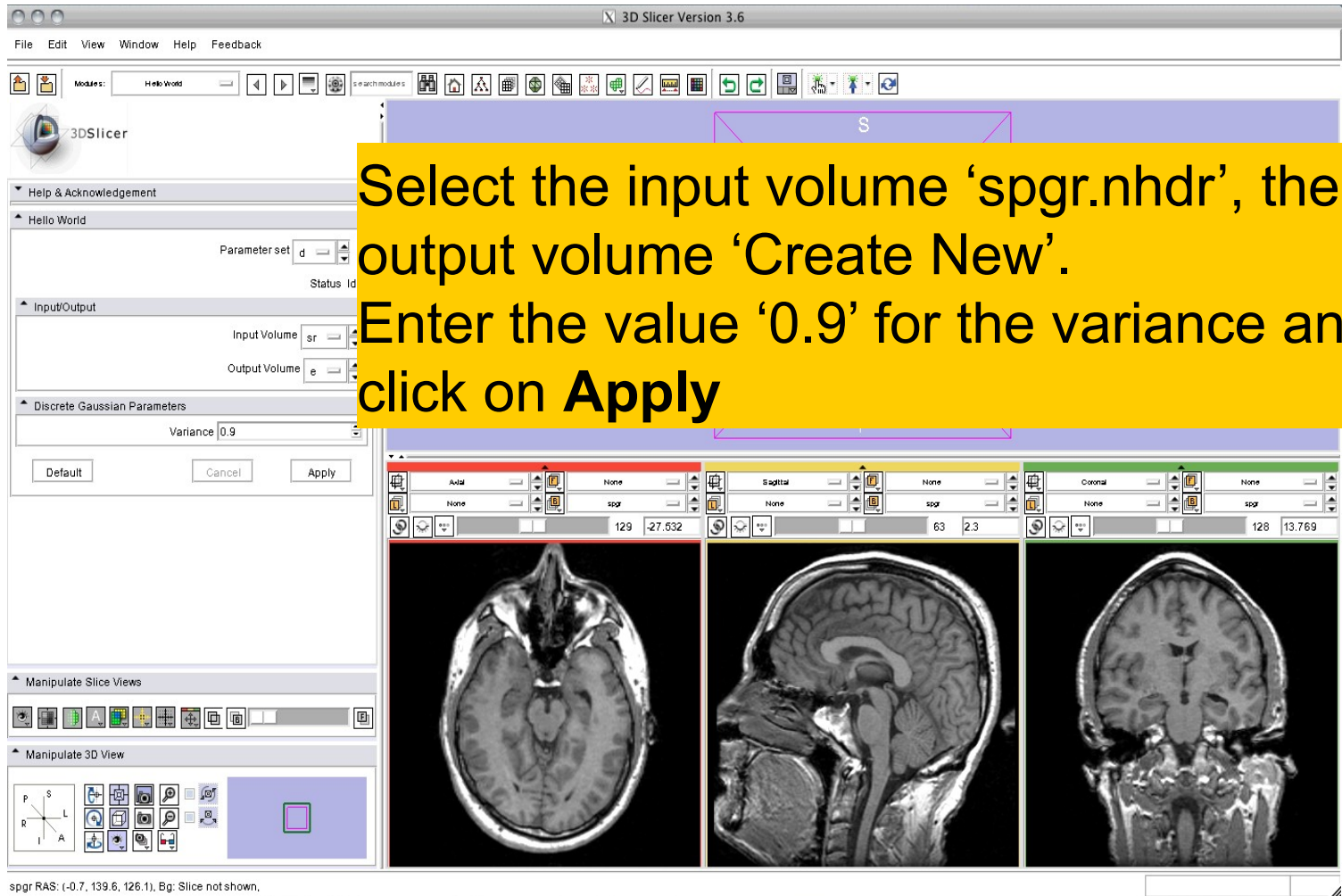




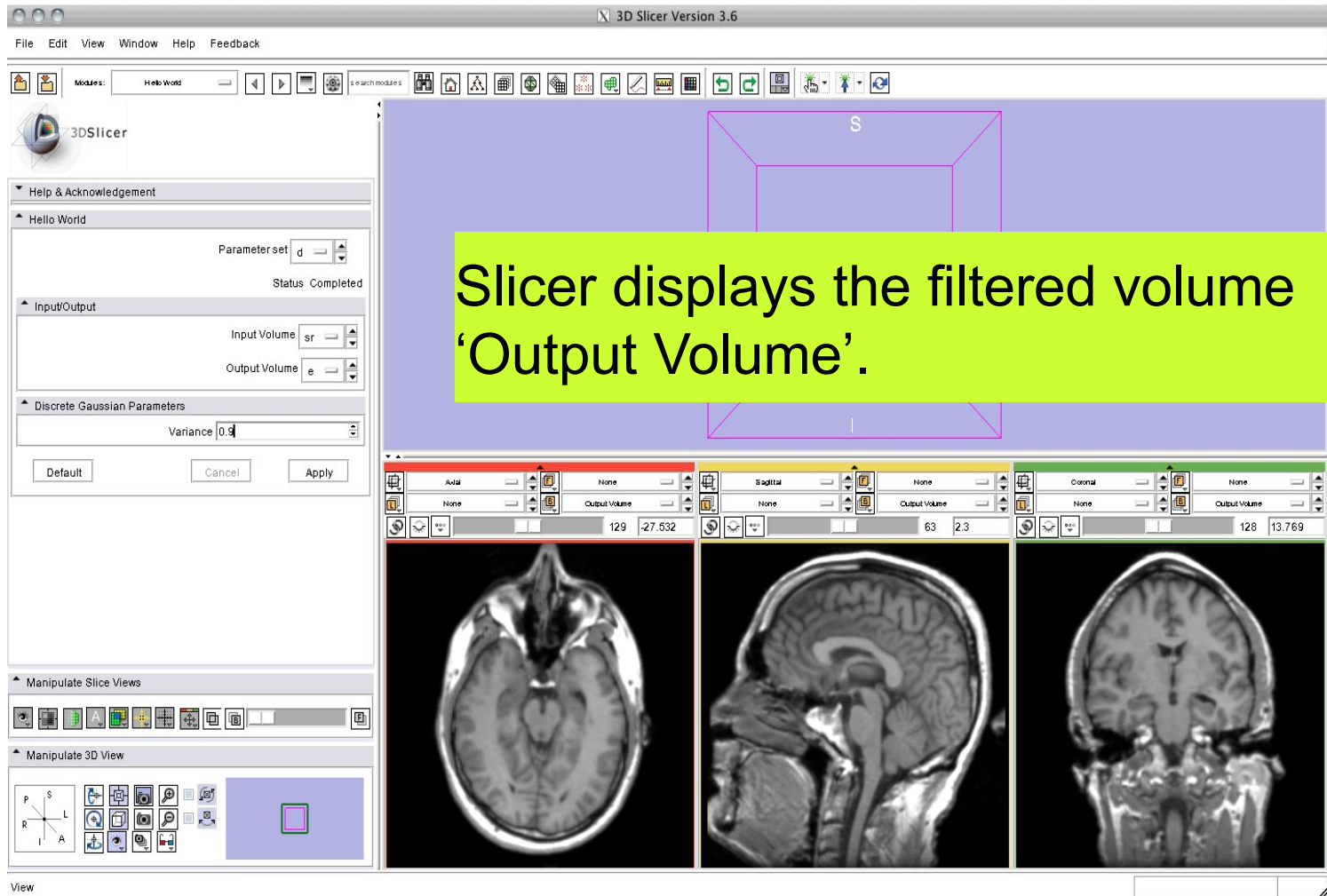
# Running the Filter



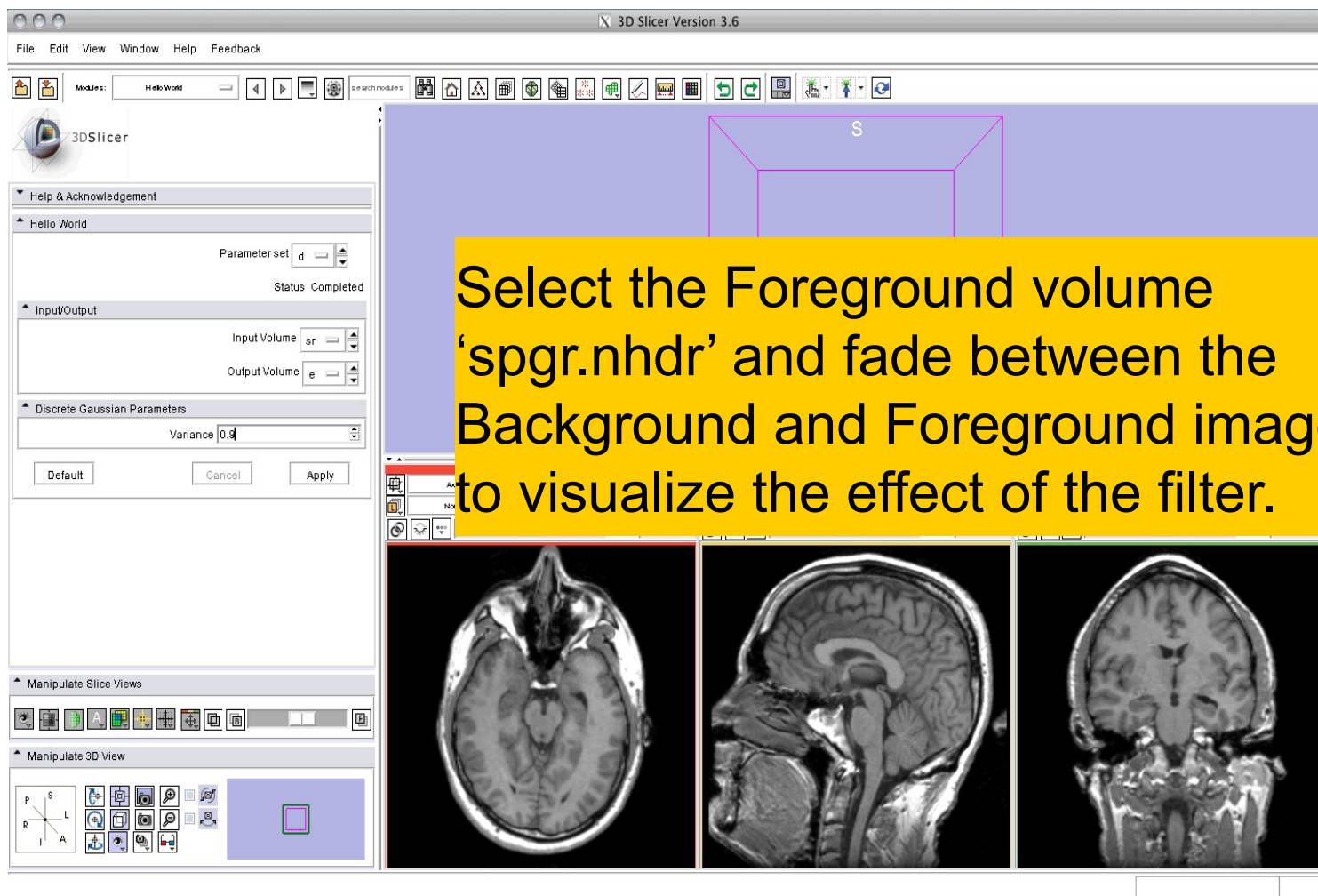
# Running the Filter

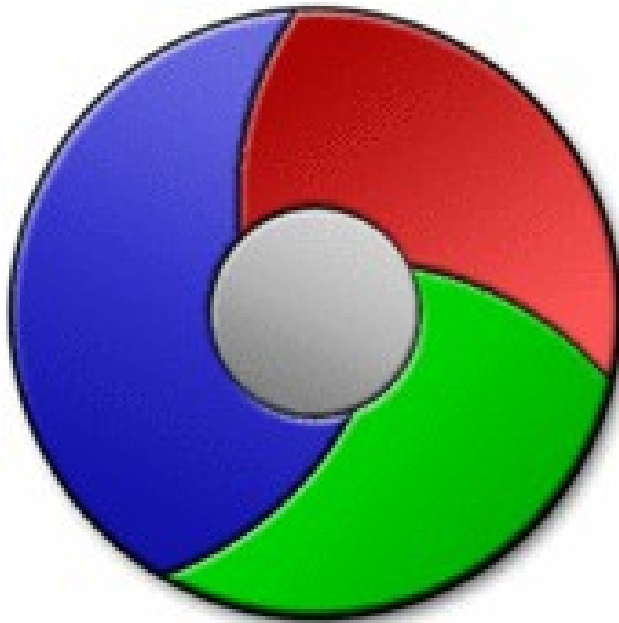


# Running the Filter



# Running the Filter





# Part C: Testing

- This section describes a [simple example for testing](#) that the 'help' functionality of our newly implemented module 'HelloWorld' works correctly.
- [CTest](#) is a core element of Slicer3's quality control system for software development.

[http://www.cmake.org/Wiki/CMake\\_Testing\\_With\\_CTest](http://www.cmake.org/Wiki/CMake_Testing_With_CTest)

- The goal of 'HelloWorldTest1' is to test the following command:

```
./HelloWorld --help
```



# HelloWorld Test 1

---

To implement the test **HelloWorldTest1**, add the following lines to the **CMakeLists.txt** file located in the **HelloWorld\** directory:

```
set (SLICER_EXE ${Slicer3_HOME}/Slicer3)
set(BUILD_SUBDIR "")
if(WIN32)
    set(BUILD_SUBDIR Debug)
endif(WIN32)
add_test>HelloWorldTest1 ${SLICER_EXE} --launch ${Slicer3_INSTALL_PLUGINS_BIN_DIR}/${BUILD_SUBDIR}/${CLP} --help
```

# Building HelloWorld

---

## Mac/Linux

Run **'make'** in the directory **HelloWorld-build/**

## Windows

Select **Build→Build Solution** to build the solution **HelloWorld.sln** located in **HelloWorld-build/**

# Testing HelloWorld

---

## Mac/Linux

- In the directory **/HelloWorld-build/** run the following command:

**/path/to/Slicer/build/Slicer3-lib/CMake-build/bin/ctest -R HelloWorldTest1**

## Windows

- In the directory **/HelloWorld-build/** run the following command:

**/path/to/Slicer/build/Slicer3-lib/CMake-build/bin/ctest.exe -R HelloWorldTest1**



# Running HelloWorldTest1

When the module successfully passes the test, the output below is generated:

```
xterm
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$
bash-3.2$ pwd
/Users/stuartwallace/Desktop/HelloWorld_Plugin/HelloWorld-build
bash-3.2$ /Users/stuartwallace/Desktop/Slicer3.6/Slicer3-lib/CMake-build/bin/ctest -R HelloWorldTest
1
Test project /Users/stuartwallace/Desktop/HelloWorld_Plugin/HelloWorld-build
  Start 1: HelloWorldTest1
1/1 Test #1: HelloWorldTest1 ..... Passed    2.00 sec

100% tests passed, 0 tests failed out of 1

Total Test time (real) =  2.00 sec
bash-3.2$ █
```

# Conclusion

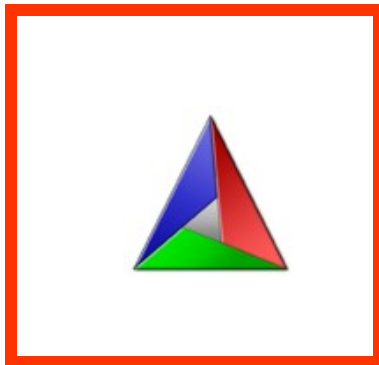
---

- This course described functionalities for **integrating**, **developing** and **testing** an external program within Slicer3.
- The **Execution Model** of Slicer3 provides a simple mechanism for incorporating command line programs as Slicer modules.
- The pipeline guided you through **6 components** of the NA-MIC kit.



# Slicer Programming Course

---





# Acknowledgments

---



**National Alliance for Medical Image Computing**

NIH U54EB005149



**Neuroimage Analysis Center**

NIH P41RR013218