

# Getting Started with ITK

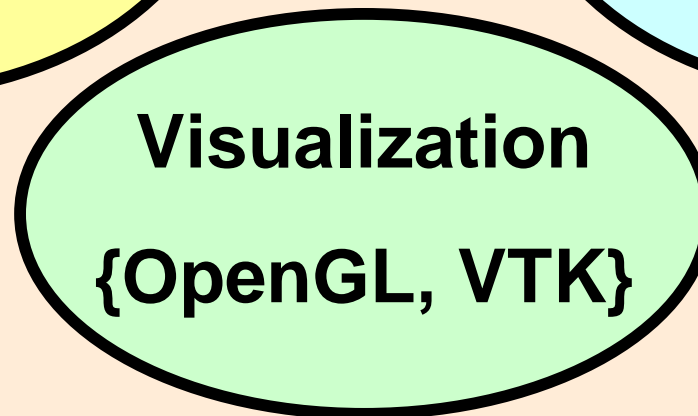
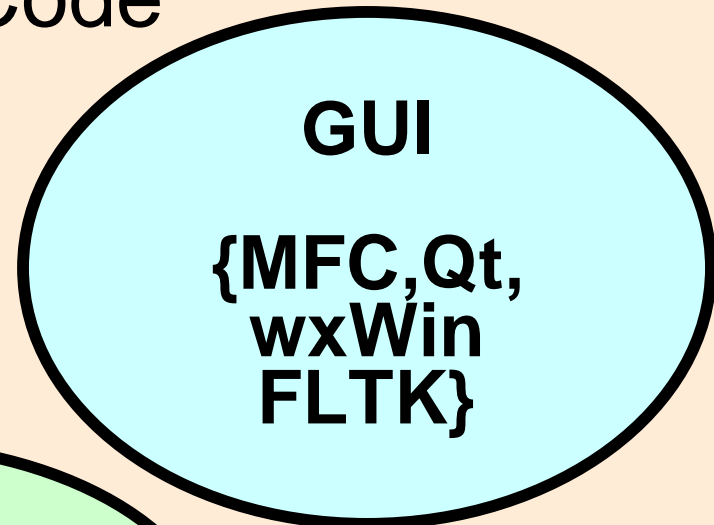
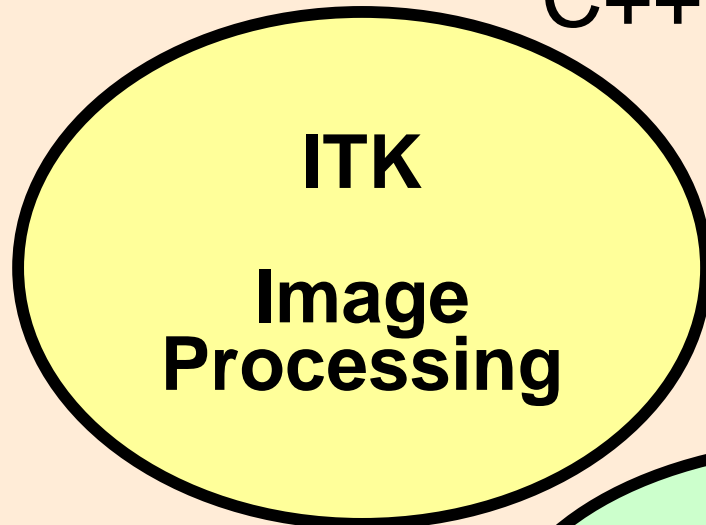
Luis Ibáñez  
Will Schroeder  
Insight Software Consortium

# What is ITK

- Image Processing
- Segmentation
- Registration
- No Graphical User Interface (GUI)
- No Visualization

# How to Integrate ITK in you application

C++ Glue Code



# What do I need ?

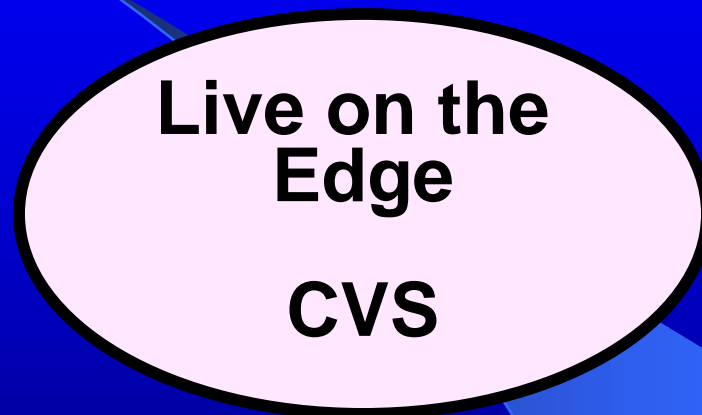
## **C++ Compiler**

**gcc 2.95 – 3.1**  
**Visual C++ 6.0**  
**Visual .NET**  
**Intel 5.0**  
**IRIX CC**  
**Borland 5.0**  
**Mac - gcc**

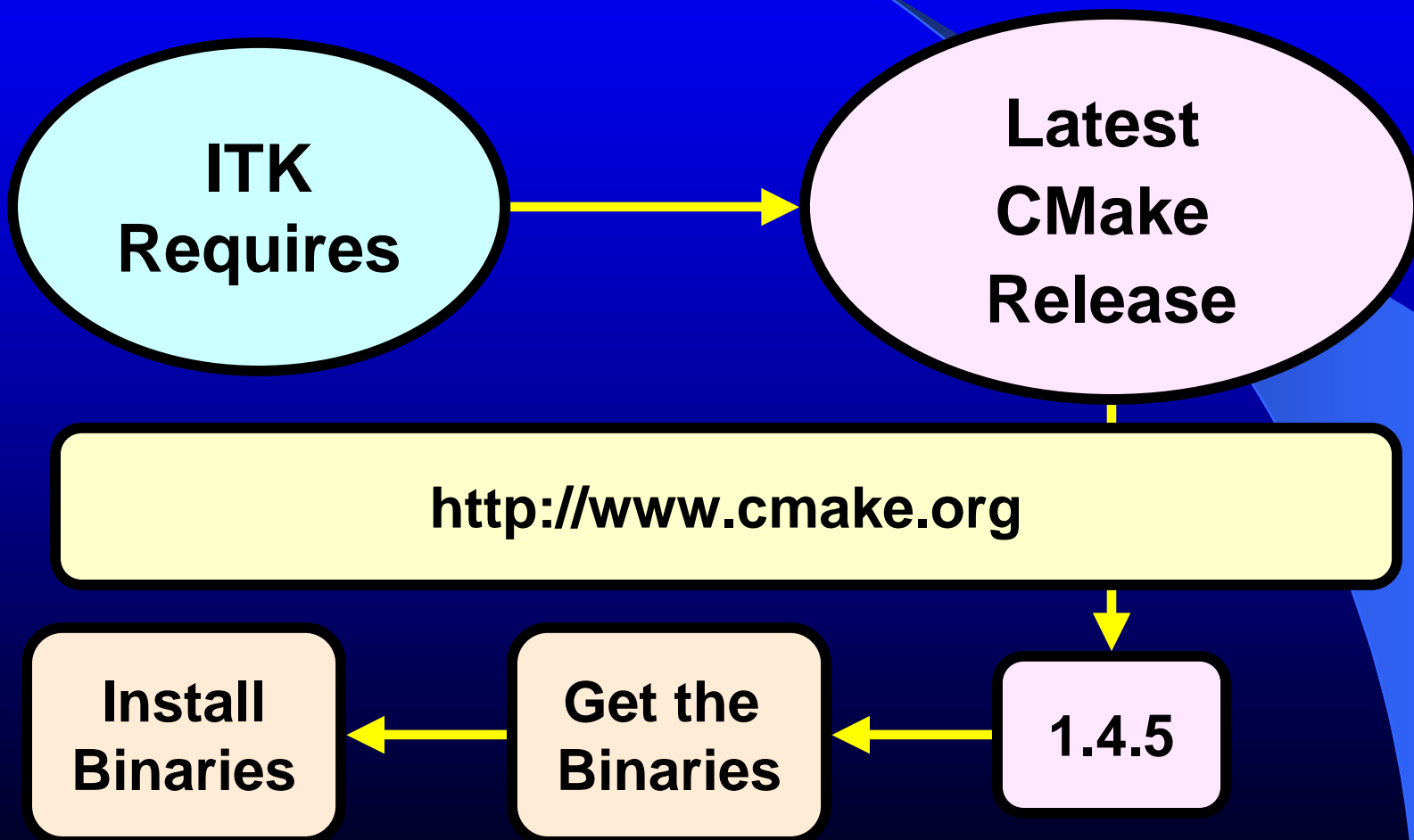
**CMake**

**[www.cmake.org](http://www.cmake.org)**

# Step 1. Download ITK

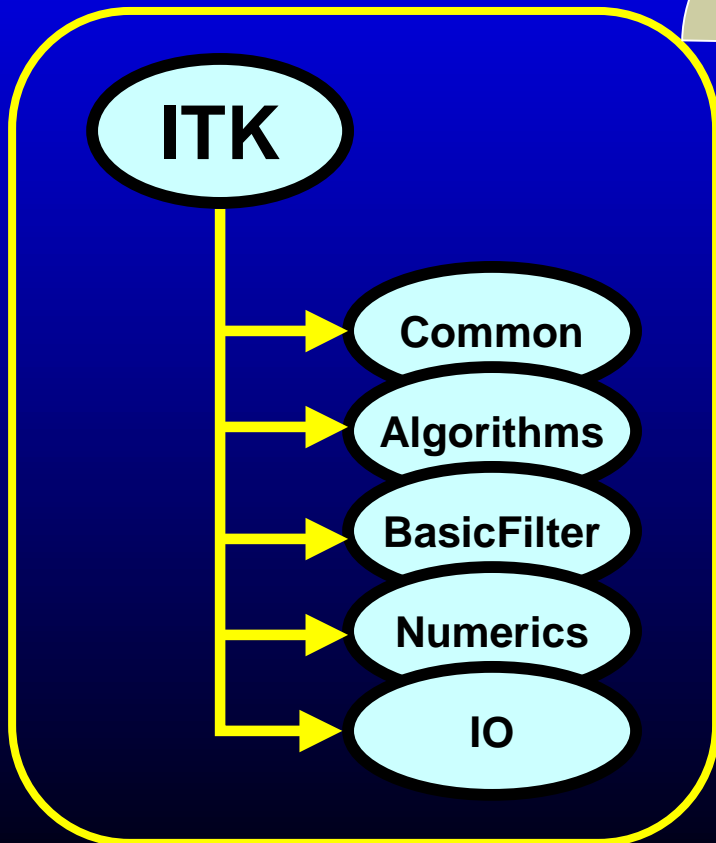


## Step 2. Download CMake



# Step 3. Configure ITK

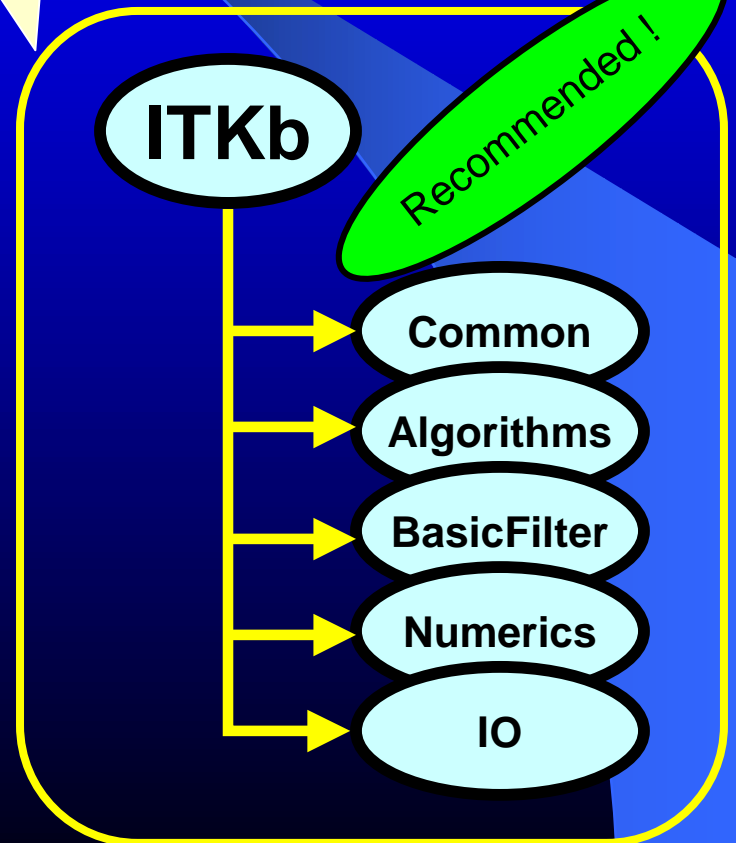
## Source Tree



Out  
Source Build

In  
Source  
Build

## Binary Tree



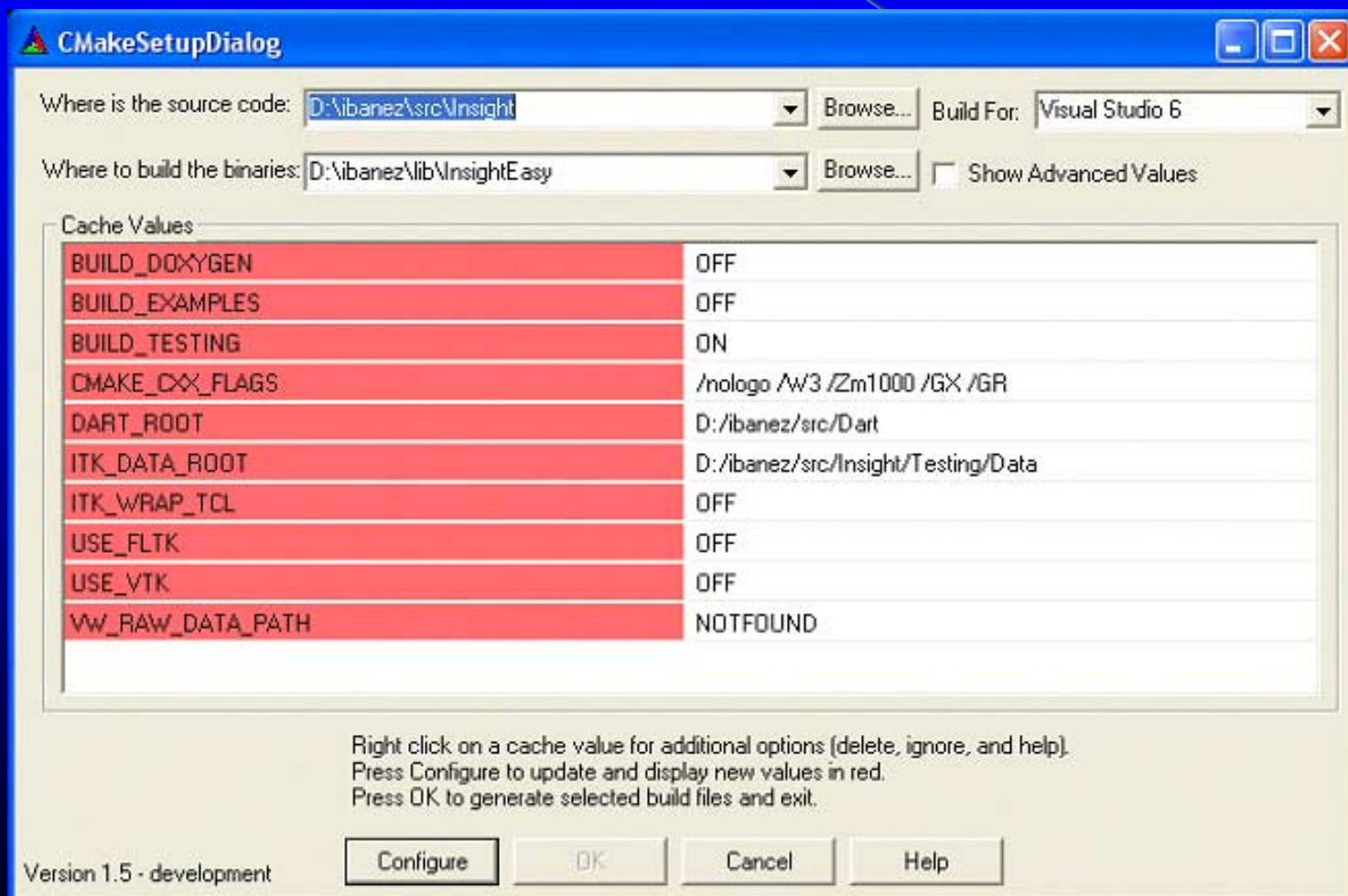
# Step 3. Configure - Easy Start

- Run CMake
- Select the SOURCE directory
- Select the BINARY directory
- Select the COMPILER
- Select CONFIGURE and then OK buttons

*(Note: CMake works iteratively. As new options are enabled, new CMake variables show up in red. Continue to select CONFIGURE until no red appears (no changes) then finally select OK to produce workspaces / makefiles)*



# Step 3. Configure - Easy Start



# Step 3. Configure - Easy Start

- Disable BUILD\_DOXYGEN
- Disable BUILD\_EXAMPLES
- Enable BUILD\_TESTING
- Disable USE\_FLTK
- Disable USE\_VTK
- Disable ITK\_WRAP\_TCL

*(Note: BUILD\_TESTING can be disabled; this will speed the build process but produce no executables (as referred to in Step 5 in the next slides))*

# Step 3. Configure - Easy Start

- Ignore CMAKE\_CXX\_FLAGS
- Ignore DART\_ROOT
- Ignore ITK\_DATA\_ROOT
- Ignore VW\_RAW\_DATA\_PATH

# Step 4. Build Project

- Open ITK.dsw in the BINARY Directory (assuming MSVC compiler)
- Select ALL\_BUILD project
  - Select configuration
    - Debug (*recommended initially*)
    - Release
    - RelWithDebugInfo
    - MinSizeRel
- Build...it will take about 1 hour ...
  - (*if BUILD\_TESTING is disabled, less than 15 minutes*)

# Step 4. Build Project

- Most of ITK classes are C++ Templates
- Basic libraries are small—they only contain non-templated classes
- Basic libraries are built in about 15 min

# Step 5. Verify the Build

- Libraries and test Executables will be found in

ITK\_BINARY / bin / { Debug, Release }

- *The actual location depends on the configuration chosen in MSVC compiler*
- *(Executables will be present only if BUILD\_TESTING was enabled in CMake)*

# Step 5. Verify the Build

The following libraries should be found

- ITKCommon
- ITKBasicFilters
- ITKAlgorithms
- ITKNumerics
- ITKFEM
- ITKIO
- ITKStatistics
- VXLNumerics
- itkpng
- itkzlib
- ITKMetaIO

# Step 5. Verify the Build

The following executables should be found

- itkCommonTests
- itkBasicFiltersTests
- itkAlgorithmsTests
- itkNumericsTests
- itkIOTests



# Step 5. Verify the Build

The following executables should be found

- itkSpatialObjectTests
- itkFEMTests
- itkStatisticsTests
- vnlTests

# Step 5. Verify the Build

The following executables should be found

- itkCommonHeaderTest
- itkBasicFiltersHeaderTest
- itkAlgorithmsHeaderTest
- itkNumericsHeaderTest
- itkIOHeaderTest
- itkSpatialObjectHeaderTest

# Step 5. Verify the Build

- Run ONE of the tests
- Murphy's Law guarantees that if there is only ONE test failing, it will be the one you randomly select !

# Step 5. Verify the Build

- *Assuming that BUILD\_TESTING was enabled:*
  - The test organization reflects the source tree structure
  - For example, in order to test the GradientImageFilter execute  
`itkBasicFiltersTests.exe itkGradientImageFilter`
  - Preferred way to execute tests:
    - `cd` into BINARY directory (as set in CMake)
    - `ctest -R itkGradientImageFilter`  
(-R says any test matching this string is executed;  
without -R all tests are executed)
    - `ctest` is a companion program to CMake

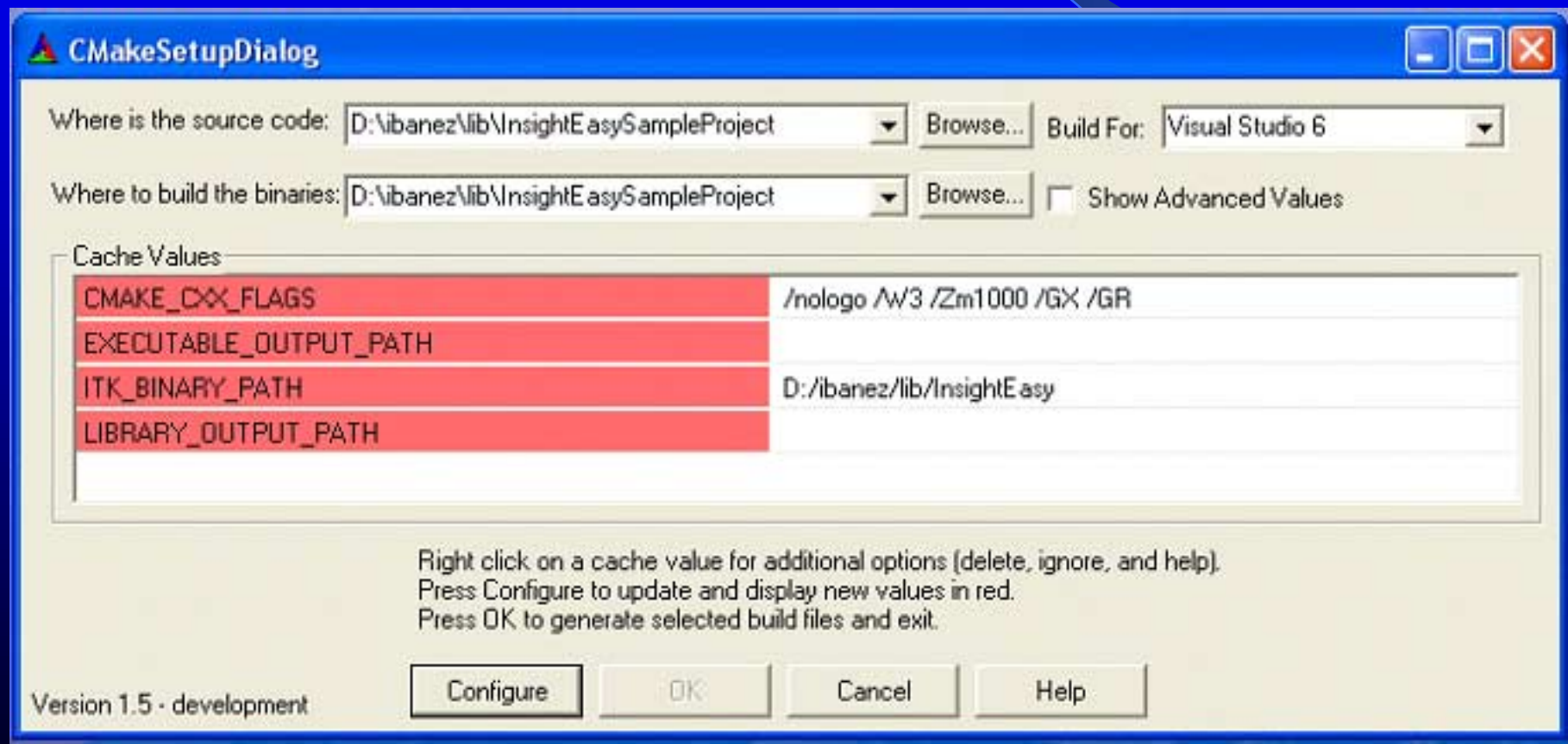
## Step 6. Use ITK from an external Project

**Copy  
“SampleProject”  
from the Examples  
Directory  
into another  
directory**

**Run  
CMake**

- **Select Source Dir**
- **Select Binary Dir**
- **Select Compiler**

# Step 6. Use ITK from an external Project



## Step 6. Use ITK from an external Project

- Ignore CMAKE\_CXX\_FLAGS
- Ignore EXECUTABLE\_OUTPUT\_PATH
- Ignore LIBRARY\_OUTPUT\_PATH
- Point `ITK_BINARY_PATH` to the binary directory where ITK was built

# Step 7. Build Sample Project

- Open SampleProject.dsw generated by CMake
- Select ALL\_BUILD project
- Build it  
...It will take about 10 seconds ...



# Step 8. Run the example

- Locate the file itkSampleProject.exe
- Run it...
- It should produce the message:  
*Test Passed !*

# Step 9. Start your own project

- Create a clean new directory
- Write a CMakeLists.txt file
- Write a simple .cxx file
- Configure with CMake
- Build
- Run

# Step 10. Writing CMakeLists.txt

```
PROJECT( myProject )
```

```
INCLUDE (${CMAKE_ROOT}/Modules/FindITK.cmake)
```

```
IF ( USE_ITK_FILE )
```

```
    INCLUDE(${USE_ITK_FILE})
```

```
ENDIF( USE_ITK_FILE )
```

```
ADD_EXECUTABLE( myProject myProject.cxx )
```

```
TARGET_LINK_LIBRARIES ( myProject
```

```
VXLNumerics ITKCommon ITKIO ITKMetaIO itkpng itkzlib )
```

# Step 11. Writing myProject.cxx

```
#include "itkImage.h"
#include "itkImageFileReader.h"
#include "itkGradientMagnitudeImageFilter.h"

int main( int argc, char **argv ) {
    typedef itk::Image<unsigned short,2>           ImageType;
    typedef itk::ImageFileReader<ImageType>         ReaderType;
    typedef itk::GradientMagnitudeImageFilter<
                                                ImageType,ImageType> FilterType;

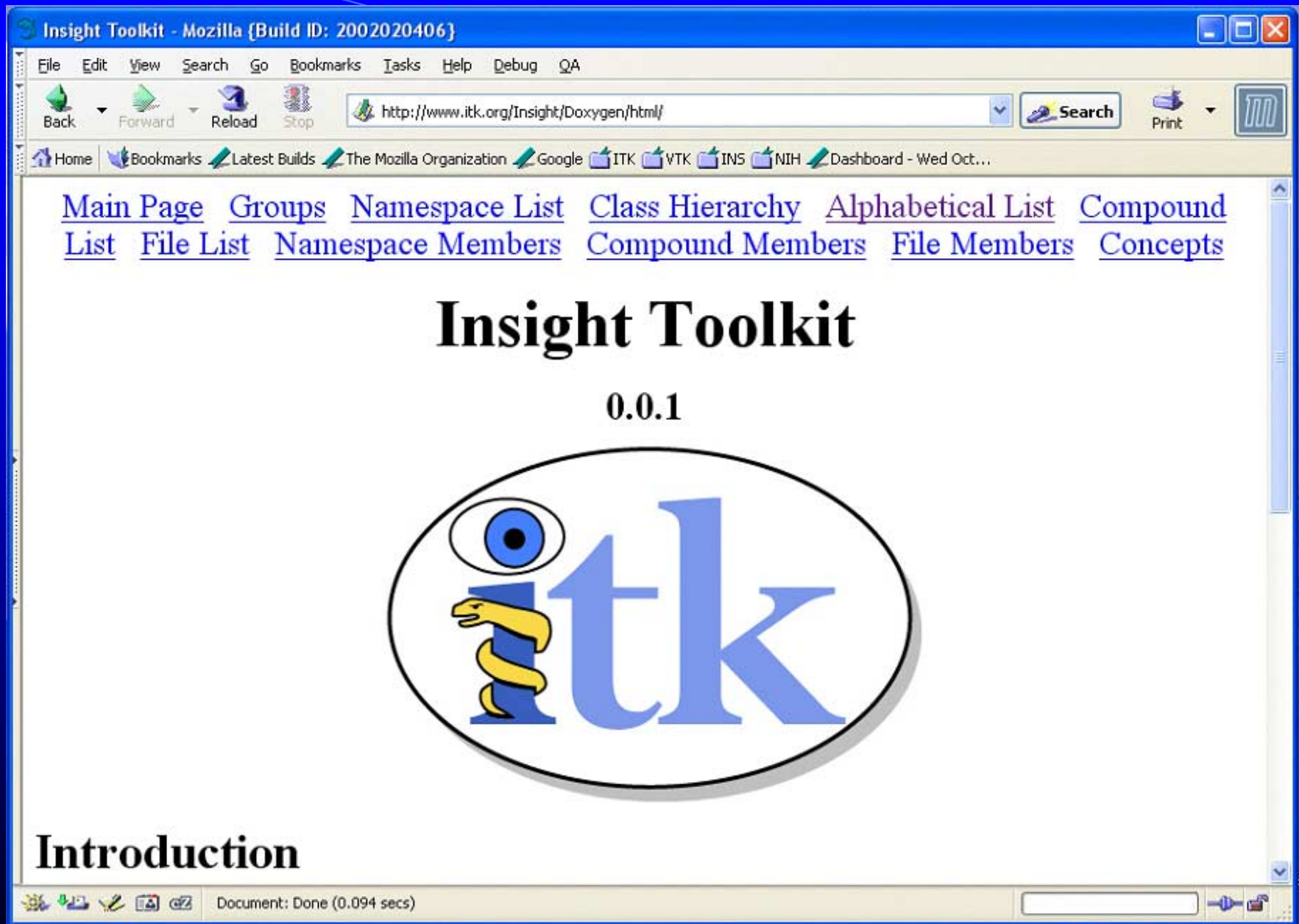
    ReaderType::Pointer reader = ReaderType::New();
    FilterType::Pointer filter = FilterType::New();

    reader->SetFileName( argv[1] );
    filter->SetInput( reader->GetOutput() );
    filter->Update();
    return 0;
}
```

# Step 12. How to find what you need

<http://www.itk.org/Doxygen/html/index.html>

- Follow the link **Alphabetical List**
- Follow the link **Groups**
- Post to the **insight-users** mailing list



Module Index - Mozilla (Build ID: 2002020406)

File Edit View Search Go Bookmarks Tasks Help Debug QA

Back Forward Reload Stop <http://www.itk.org/Insight/Doxygen/html/modules.html> Search Print

Home Bookmarks Latest Builds The Mozilla Organization Google ITK VTK INS NIH Dashboard - Wed Oct...

[Main Page](#) [Groups](#) [Namespace List](#) [Class Hierarchy](#) [Alphabetical List](#) [Compound List](#) [File List](#) [Namespace Members](#) [Compound Members](#) [File Members](#) [Concepts](#)

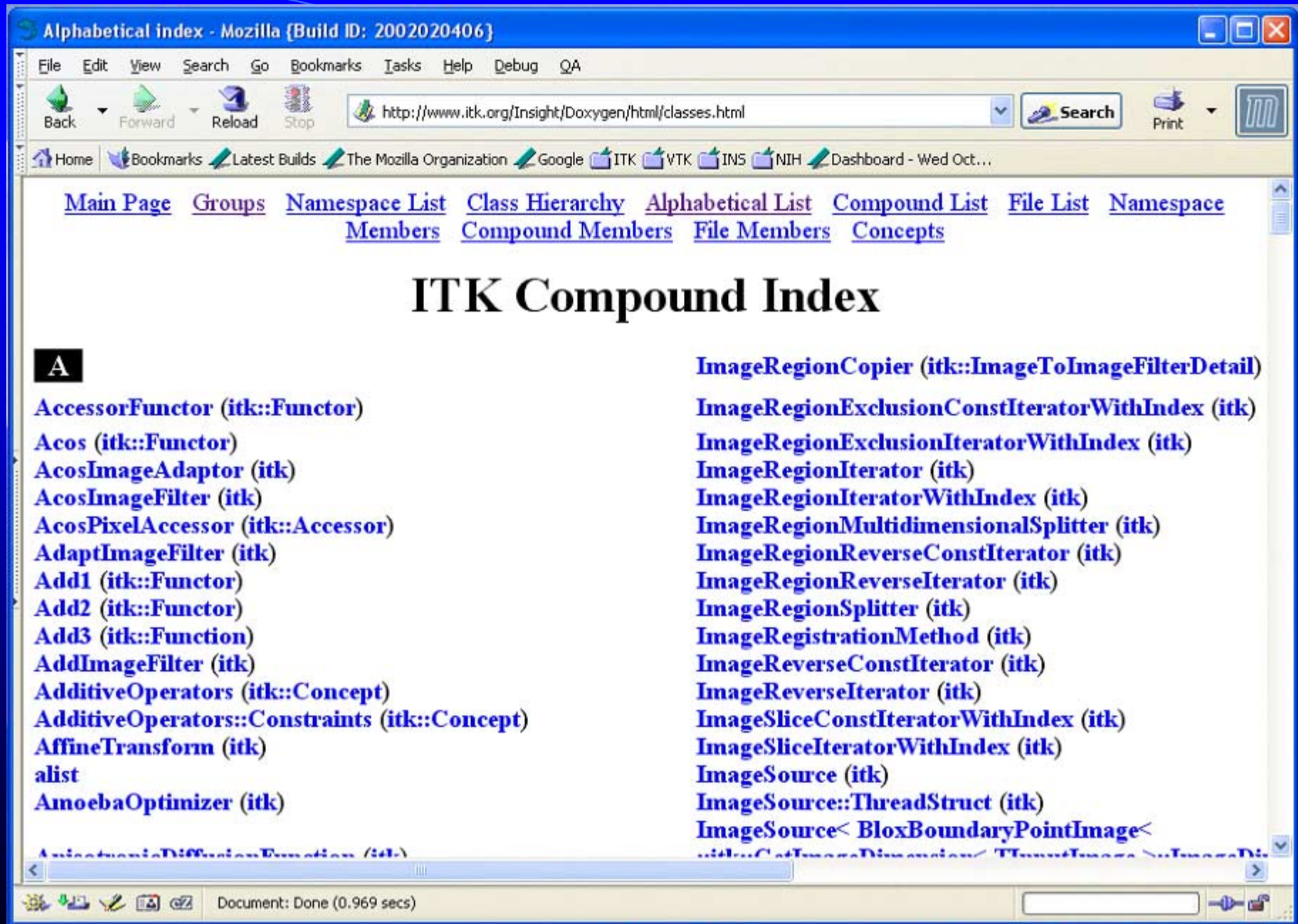
## ITK Modules

Here is a list of all modules:

- **Data Representation Objects**
  - ◊ **Image Representation Objects**
  - ◊ **Mesh Representation Objects**
  - ◊ **Geometry Representation Objects**
- **Data Access Objects**
  - ◊ **Image Access Objects**
  - ◊ **Mesh Access Objects**
  - ◊ **Iterators**
    - **Image Iterators**
- **Data Processing Objects**
  - ◊ **Filters**
    - **Image Filters**
      - **Intensity Image Filters**
        - **Mathematical Morphology Image Filters**
      - **Image Enhancement Filters**
      - **Image Feature Extraction Filters**
        - **Image Gradient Filters**
      - **Image Segmentation Filters**

Document: Done (0.14 secs)







# Additional Resources

- User Mailing List
  - <http://www.itk.org/mailman/listinfo/insight-users>
- Further Documentation
  - InsightDocuments (cvs checkout)

`cvs -d :pserver:anonymous@www.itk.org:/insight/cvsroot login`

with password "insight"...then get the source/documentation as follows:

`cvs -d :pserver:anonymous@www.itk.org:/insight/cvsroot checkout  
InsightDocuments`



**Enjoy ITK !**