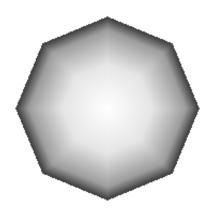
VTK/Examples/Cxx/Utilities/Animation

From KitwarePublic < VTK | Examples | Cxx

This example demonstrates how to create a simple animation. A timer is used to move a sphere across a scene.

Animation.cxx

```
#include <vtkSmartPointer.h>
#include <vtkSphereSource.h>
#include <vtkPolyData.h>
#include <vtkPolyDataMapper.h>
#include <vtkActor.h>
#include <vtkCommand.h>
#include <vtkRenderer.h>
#include <vtkRenderWindow.h>
#include <vtkRenderWindowInteractor.h>
class vtkTimerCallback2 : public vtkCommand
  public:
    static vtkTimerCallback2 *New()
      vtkTimerCallback2 *cb = new vtkTimerCallback2;
      cb->TimerCount = 0;
      return cb;
    virtual void Execute(vtkObject *caller, unsigned long event
                         void * vtkNotUsed(callData))
      if (vtkCommand::TimerEvent == eventId)
        ++this->TimerCount;
      std::cout << this->TimerCount << std::endl;</pre>
      actor->SetPosition(this->TimerCount, this->TimerCount,0);
      vtkRenderWindowInteractor *iren = vtkRenderWindowInteract
      iren->GetRenderWindow()->Render();
  private:
    int TimerCount;
  public:
    vtkActor* actor;
int main(int, char* [])
  // Create a sphere
  vtkSmartPointer<vtkSphereSource> sphereSource =
    vtkSmartPointer<vtkSphereSource>::New();
  sphereSource->SetCenter(0.0, 0.0, 0.0);
  sphereSource->SetRadius(5.0);
  sphereSource->Update();
  // Create a mapper and actor
  vtkSmartPointer<vtkPolyDataMapper> mapper =
    vtkSmartPointer<vtkPolyDataMapper>::New();
  mapper->SetInputConnection(sphereSource->GetOutputPort());
  vtkSmartPointer<vtkActor> actor = vtkSmartPointer<vtkActor>:
  actor->SetMapper(mapper);
```



```
// Create a renderer, render window, and interactor
vtkSmartPointer<vtkRenderer> renderer =
 vtkSmartPointer<vtkRenderer>::New();
vtkSmartPointer<vtkRenderWindow> renderWindow =
 vtkSmartPointer<vtkRenderWindow>::New();
renderWindow->AddRenderer(renderer);
vtkSmartPointer<vtkRenderWindowInteractor> renderWindowIntera
  vtkSmartPointer<vtkRenderWindowInteractor>::New();
renderWindowInteractor->SetRenderWindow(renderWindow);
// Add the actor to the scene
renderer->AddActor(actor);
renderer->SetBackground(1,1,1); // Background color white
// Render and interact
renderWindow->Render();
// Initialize must be called prior to creating timer events.
renderWindowInteractor->Initialize();
// Sign up to receive TimerEvent
vtkSmartPointer<vtkTimerCallback2> cb =
  vtkSmartPointer<vtkTimerCallback2>::New();
cb->actor = actor;
renderWindowInteractor->AddObserver(vtkCommand::TimerEvent, c
int timerId = renderWindowInteractor->CreateRepeatingTimer(10)
std::cout << "timerId: " << timerId << std::endl;</pre>
// Start the interaction and timer
renderWindowInteractor->Start();
return EXIT_SUCCESS;
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.8)

PROJECT(Animation)

find_package(VTK REQUIRED)
   include(${VTK_USE_FILE})

add_executable(Animation MACOSX_BUNDLE Animation)

if(VTK_LIBRARIES)
    target_link_libraries(Animation ${VTK_LIBRARIES})
   else()
   target_link_libraries(Animation vtkHybrid vtkWidgets)
   endif()
```

Download and Build Animation

Click here to download Animation

(http://gitorious.org/vtkwikiexamplestarballs/vtkwikiexamplestarballs/raw/master:Animation.tar). and its *CMakeLists.txt* file.

Once the tarball *Animation.tar* has been downloaded and extracted,

```
cd Animation/build
```

<u> </u>	
■ If VTK is installed:	
cmake	
If VTK is not installed but compiled on your system, you will need to specify the path to your VTK build:	
cmake -DVTK_DIR:PATH=/home/me/vtk_build	 1 1 1
Build the project:	
make	
and run it:	
 ./Animation	 ! ! !

WINDOWS USERS PLEASE NOTE: Be sure to add the VTK bin directory to your path. This will resolve the VTK dll's at run time.

Color.

Retrieved from "http://www.vtk.org/Wiki/index.php? title=VTK/Examples/Cxx/Utilities/Animation&oldid=50264" Category: VTKMakeMoreInteresting

- This page was last modified on 10 December 2012, at 15:56.
- Content is available under Attribution 2.5 unless otherwise noted.