How to 

in 16h

James Mayr

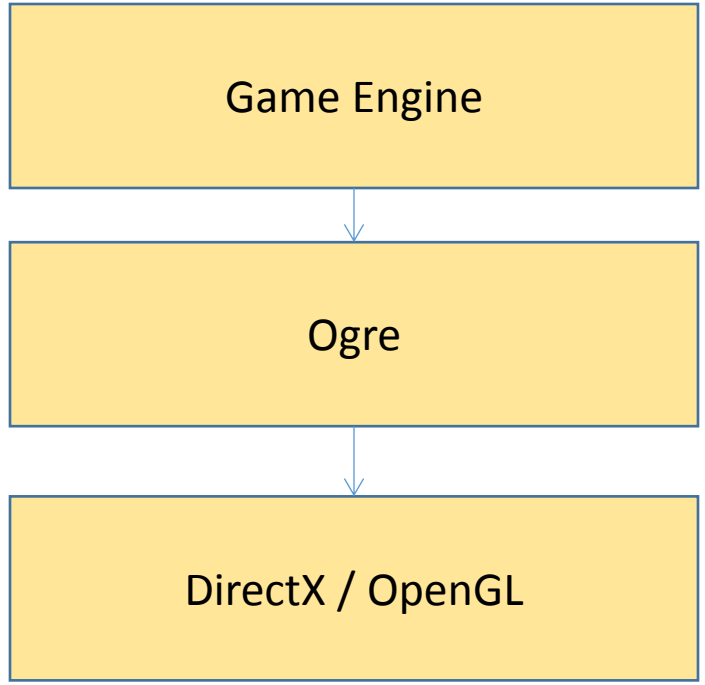
Inhalt

- Was ist ogre?
- Vergleich Unity
- Erste Schritte
- Minimal Ogre
- Implementationsdetail
- Fazit / Fragen
- Demonstration

Was ist Ogre?

Object-Oriented Graphics Rendering Engine

- MIT Lizenz (kostenlos, Quelloffen, auch für kommerzielle Projekte)
- Plattformunabhängig
- Keine Game-Engine!
- In C++ geschrieben / in IDE entwickeln
- Nutzt OpenGL / DirectX
- Wrapper für Java /.Net



Vergleich Unity

Was Ogre **nicht** ist: Game Engine

Kein:

- Global Illumination / Baked light
- WheelCollider, Triggers, ect.
- Collision detection
- Kein Szenen Editor


You're supposed to eat grass, not smoke it.


File
Edit
View
Editors
Camera
Renderer
Engine
Tools
DEV
Help
Scene Entity List
Lights
Default - Light Directional
Characters
Knightress_153
Knightress_154
Knightress_152
Knightress_151
Knightress_154
Scene
City Wall
Fence
Misc
Rocks
Rock2_2
Rock1B_247
Rock4A_12
Rock2_46
Rock2_29
Rock2_75
Rock2_40
Rock2_8
Rock4A_120
Rock2_6
Rock2_26
Skyline Console
Finished Updating Billboards
Finished Updating Billboards
Finished Updating Billboards
LOADING XML: Terrain Road Settings
LOADING XML: CAMERA Settings
LOADING XML: Load Bookmarks
LOADING XML: Post Effects
LOADING XML: SHADOW Settings
LOADING XML: Navmesh Settings
******* XML :: MISSING FILE:
FileName = E:/03 - Ogre/01 - Projects/aRa_PRO
file:E:/03 - Ogre/01 - Projects/aRa_PROJECT-SK
LOAD LEVEL COMPLETED [XML]
Clear Log
Enter Commands Here....
World Space
Cam Speed:
1.00
Terrain Editor
Creation
Height Edit
Paint
Vegetation
Terrain Generator
Terrain Setup
Terrain Size (Resolution)
512
World Size (Size in World Units
1000
Texture Blend Map Size:
1024
Tiles X
0
Tiles Y
0
Generate
Terrain Options:
Generate
Remove
Apply Physics
Import Options:
Export Options:
Load Terrain file
Export Terrain file
Load HeightMap
Export HeightMap
Terrain Properties
Heightmap Multiplier
75
Use Surface Names
Update Surfaces
Open AM Heightmaps
Open
Lightmap
Material
Brush Size
x 1
Brush Strength
Scene Settings
Terrain Editor
Particle Editor

Setup

- Online sehr gut dokumentiert
- Ogre Bibliothek herunterladen/installieren
- Direct X installieren (SDK for Windows 8.1)
- Visual Studio 2012 installieren
- Herunterladen Tutorial Framework
- Projekt einrichten (Linker/Compiler/PreCompiled Headers/Libraries)


TutorialApplication Render Window
FPS: 148
Average FPS: 153.7
Best FPS: 166.8
Worst FPS: 148.9
Triangles: 17568
Batches: 28
OGRE

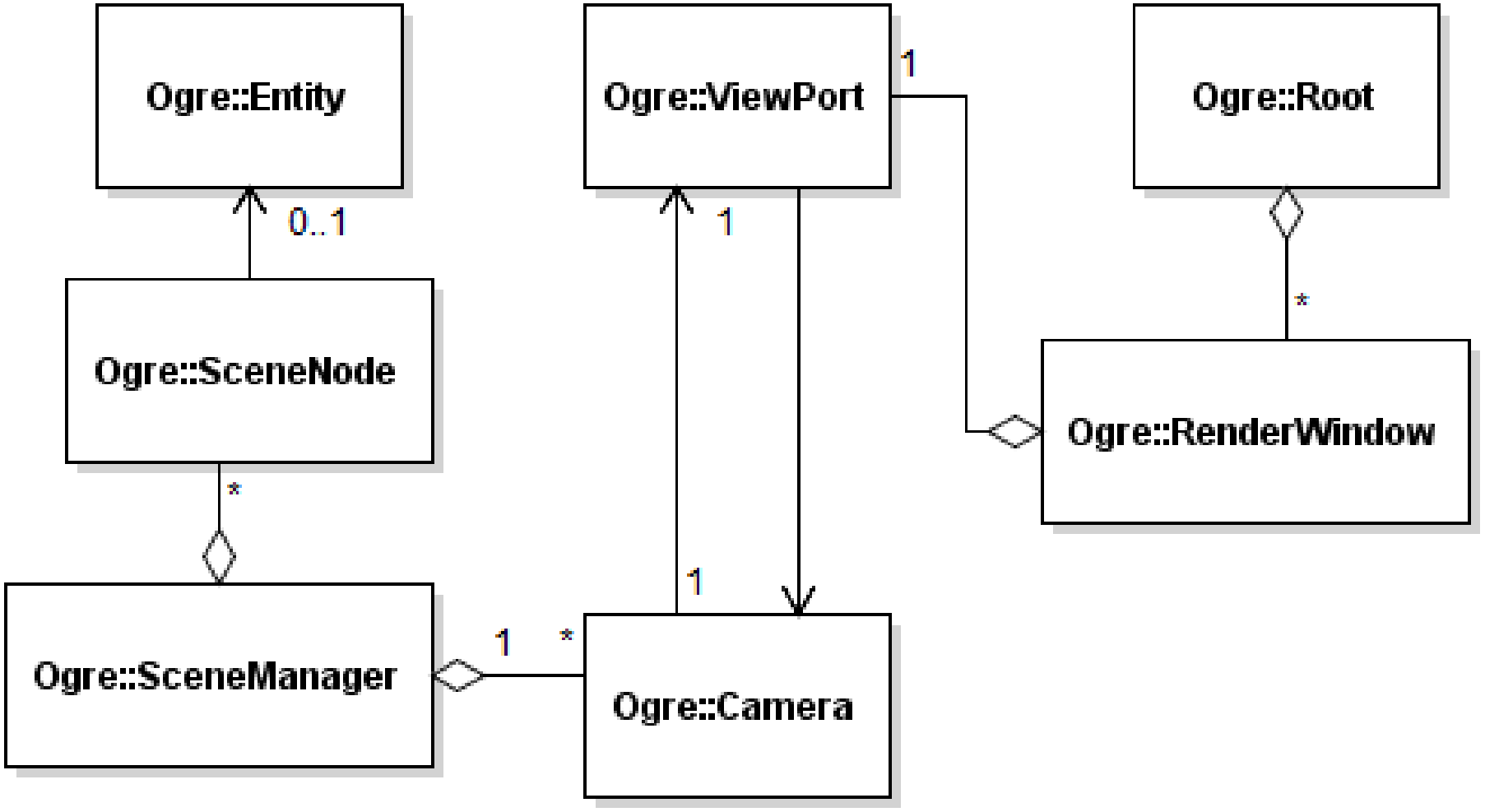
Erste Schritte

```
mSceneMgr->setAmbientLight(Ogre::ColourValue(0.5, 0.5, 0.5));
```

```
Ogre::Entity* ogreEntity3 = mSceneMgr->createEntity("ogrehead.mesh");

Ogre::SceneNode* ogreNode3 = mSceneMgr->getRootSceneNode()->createChildSceneNode();
ogreNode3->setPosition(0, 104, 0);
ogreNode3->setScale(2, 1.2, 1);
ogreNode3->attachObject(ogreEntity3);
```

Komponenten



Game Cycle

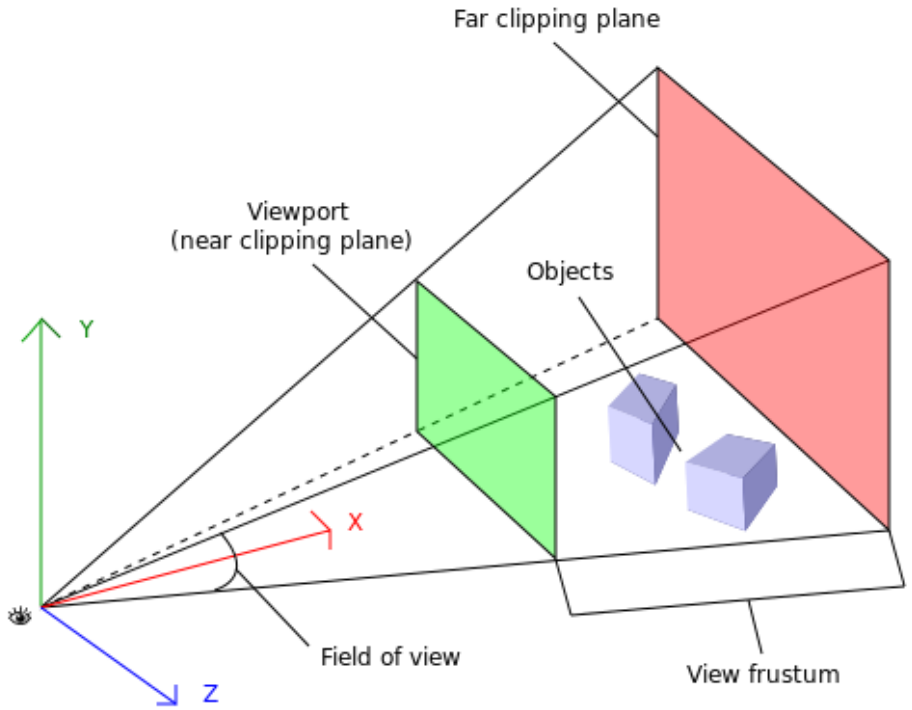
Interfaces für Listener

```
class BaseApplication : public Ogre::FrameListener,
    public Ogre::WindowEventListener,
    public OIS::KeyListener,
    public OIS::MouseListener,
    OgreBites::SdkTrayListener
{
protected:
    // Ogre::FrameListener
    virtual bool frameRenderingQueued(const Ogre::FrameEvent& evt);

    // OIS::KeyListener
    virtual bool keyPressed( const OIS::KeyEvent &arg );
    virtual bool keyReleased( const OIS::KeyEvent &arg );
    // OIS::MouseListener
    virtual bool mouseMoved( const OIS::MouseEvent &arg );
    virtual bool mousePressed( const OIS::MouseEvent &arg, OIS::MouseButtonID id );
    virtual bool mouseReleased( const OIS::MouseEvent &arg, OIS::MouseButtonID id );
};
```

Implementationsdetail

Flugzeug soll sich in Cursorrichtung drehen:



Gegeben: Koordinate des Viewport

Gesucht: Schnittpunkt Geraden “Kamera Viewport” mit zx Ebene

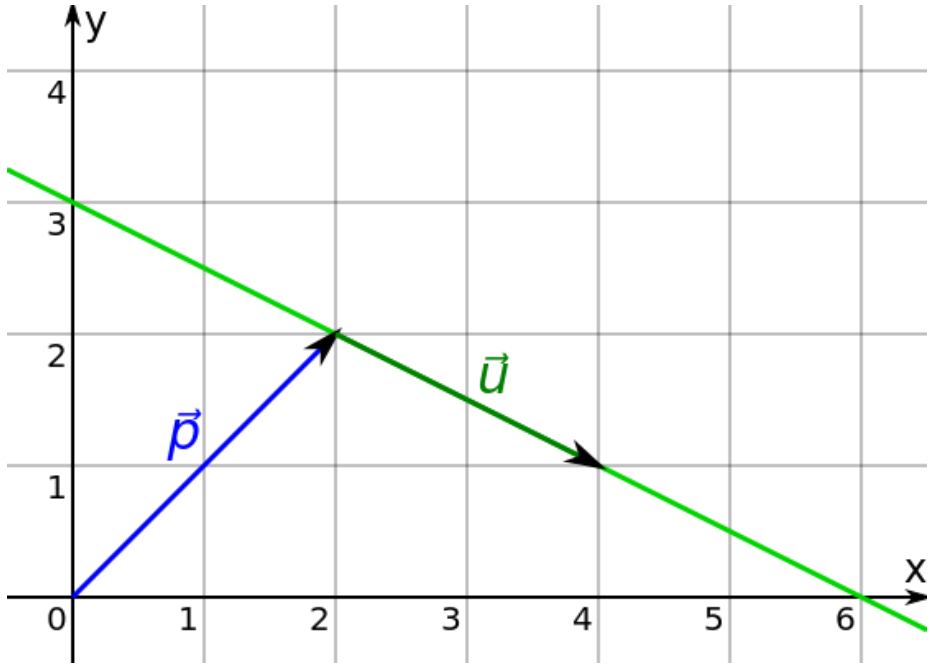
Implementationsdetail 2

```
Ogre::Viewport* vp = mSceneMgr->getCurrentViewport();
Ogre::Real y = (Ogre::Real)arg.state.Y.abs /vp->getActualHeight();
Ogre::Real x = (Ogre::Real)arg.state.X.abs /vp->getActualWidth();
Ogre::Ray mouseRay = mCamera->getCameraToViewportRay(x,y);

Ogre::Vector3 origin = mouseRay.getOrigin();
Ogre::Vector3 direction = mouseRay.getDirection();

//Calculate Lamda (distance)
//-> where the ray hit's the x-z plane
//origin.y + lamda * direction.y == 0
Ogre::Real lamda=0;
lamda = -origin.y/direction.y;

mCursor = mouseRay.getPoint(lamda);
```



Implementationsdetail 3

```
Ogre::Vector3 curPos = mNode->getPosition();
Ogre::Vector3 direction = curPos - mCursorPos;

Ogre::Vector3 src = mNode->getOrientation() * -Ogre::Vector3::UNIT_Z;
src.y = 0;
direction.y = 0;
src.normalise();
direction.normalise();

mNode->setPosition(mNode->getPosition()+mCurrentMoveVector);
Ogre::Quaternion quat = src.getRotationTo(direction);

mNode->rotate(quat);
```

Fazit

Pro

- wenig Ressourcen
- Plattformunabhängig
- Opensource / gute API Doc
- OO Design
- Spass in C++ zu entwickeln
- Plugin basiert/ erweiterbar

Kontra

- Kein Szenen Editor
- Modelle import
- Nur Render Engine
- Ineffiziente Entwicklung
- Hohe Einstiegshürde
- Orthonormale Perspektive unbrauchbar

Demonstration / Infos

SourceCode & Ogre Installationsanleitung

http://github.com/tscheims1/ogre