



VEKTORIA-MANUAL DEVICES



Inhalt der Vorlesung



/// **VERTIEFUNG**

/// **IO-DEVICES**

/// **TASTATUR**

/// **MAUS & CURSOR**

/// **GAME CONTROLLER**



■■■ VERTIEFUNG

2 ■■■

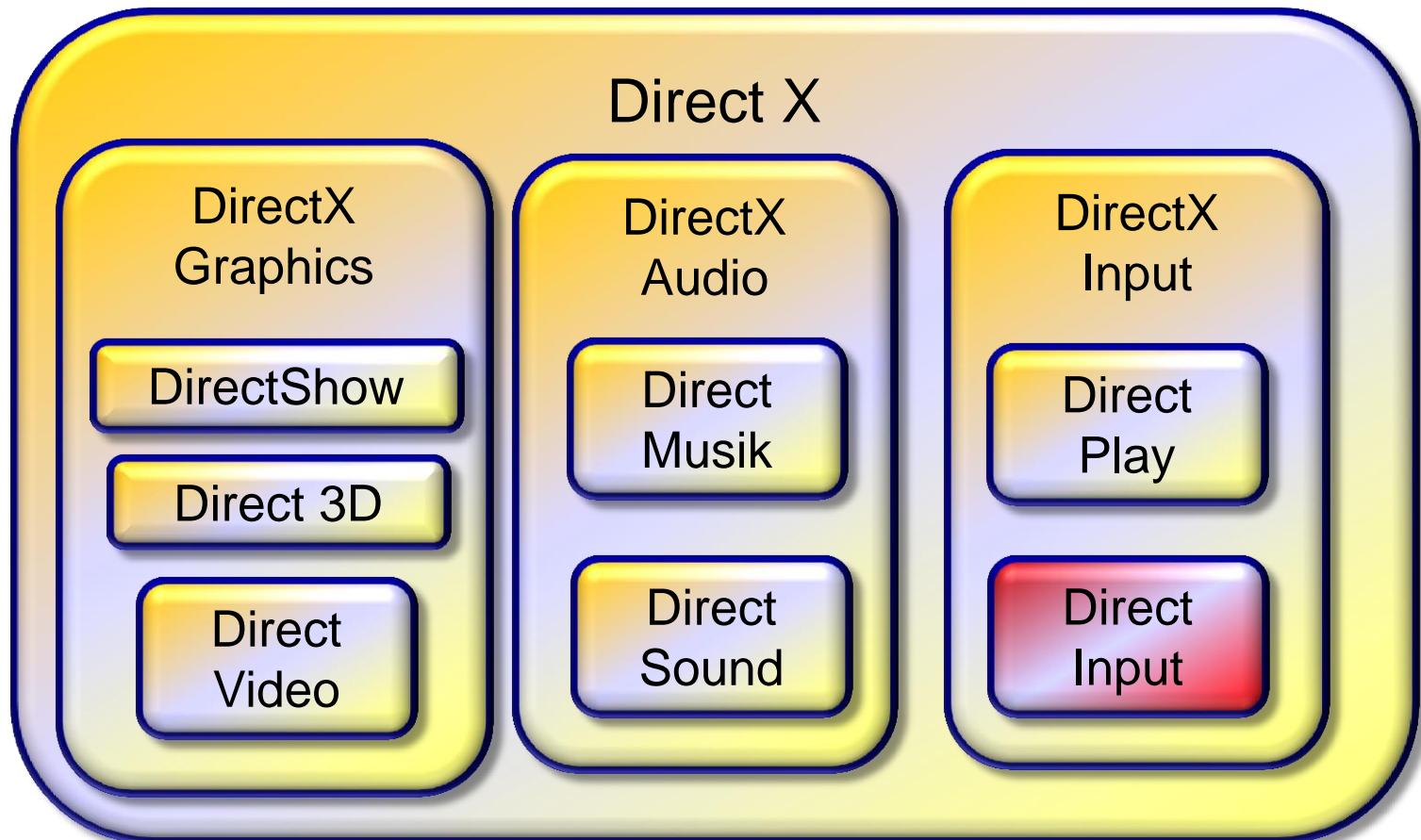
3 ■■■

4 ■■■

5 ■■■



Direct Input



Kapitel 2

Kapitel 2



/// **IO-DEVICES**



Knotenobjekte der Szeneraphen Devices

Devices sind Ein- und Ausgabegeräte

Synonyme in anderen Szeneagrafen:

Input Devices, Inputs

Beispiele:

- Tastatur
- 3D Maus

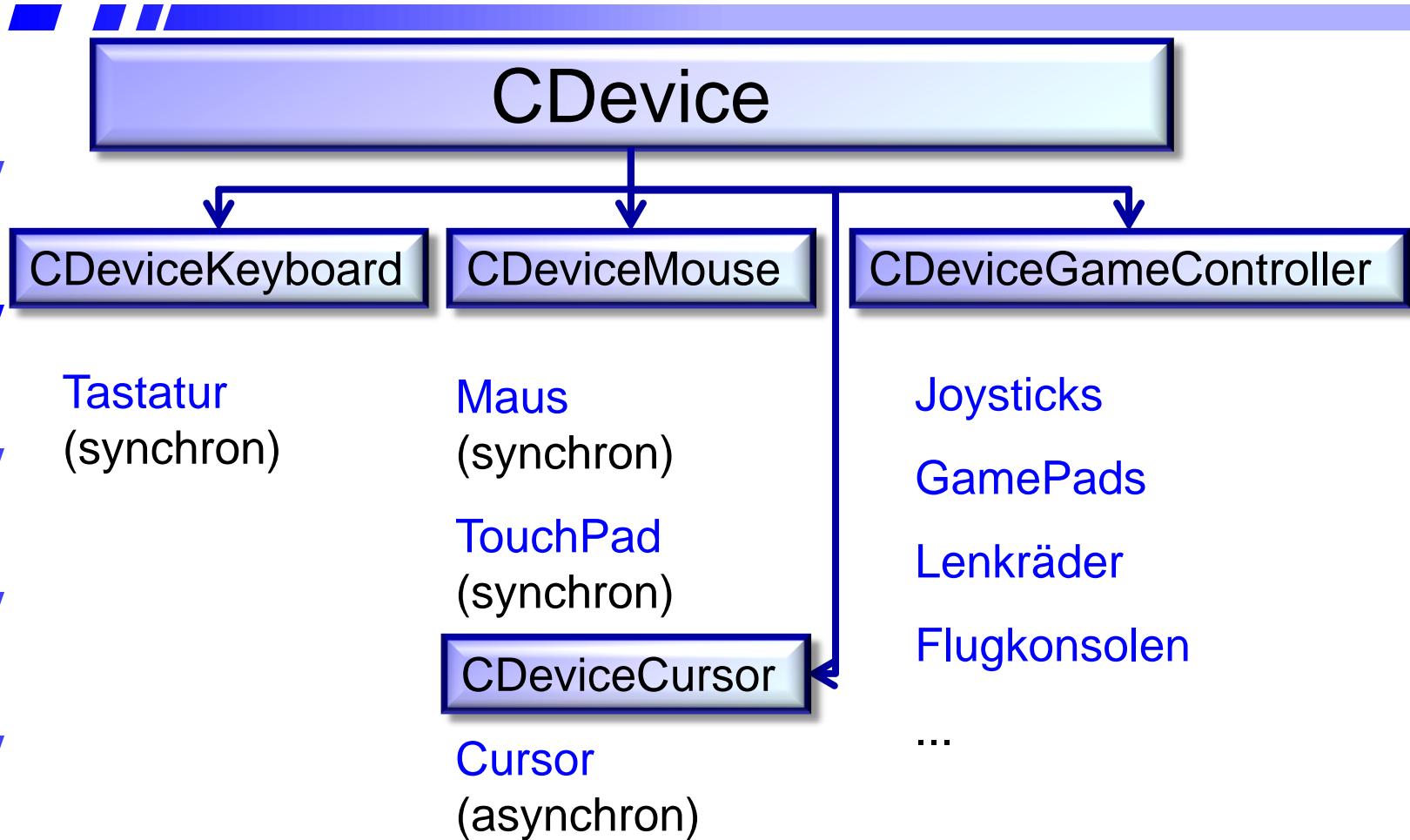
- Maus
- Cursor (nicht zu verwechseln mit Maus)

- TouchPad
- Lenkrad
- Joystick mit und ohne Force Feedback

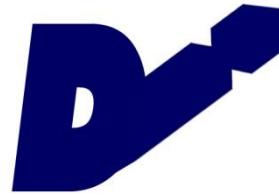
- Game Controller
- Motion Base
- ...



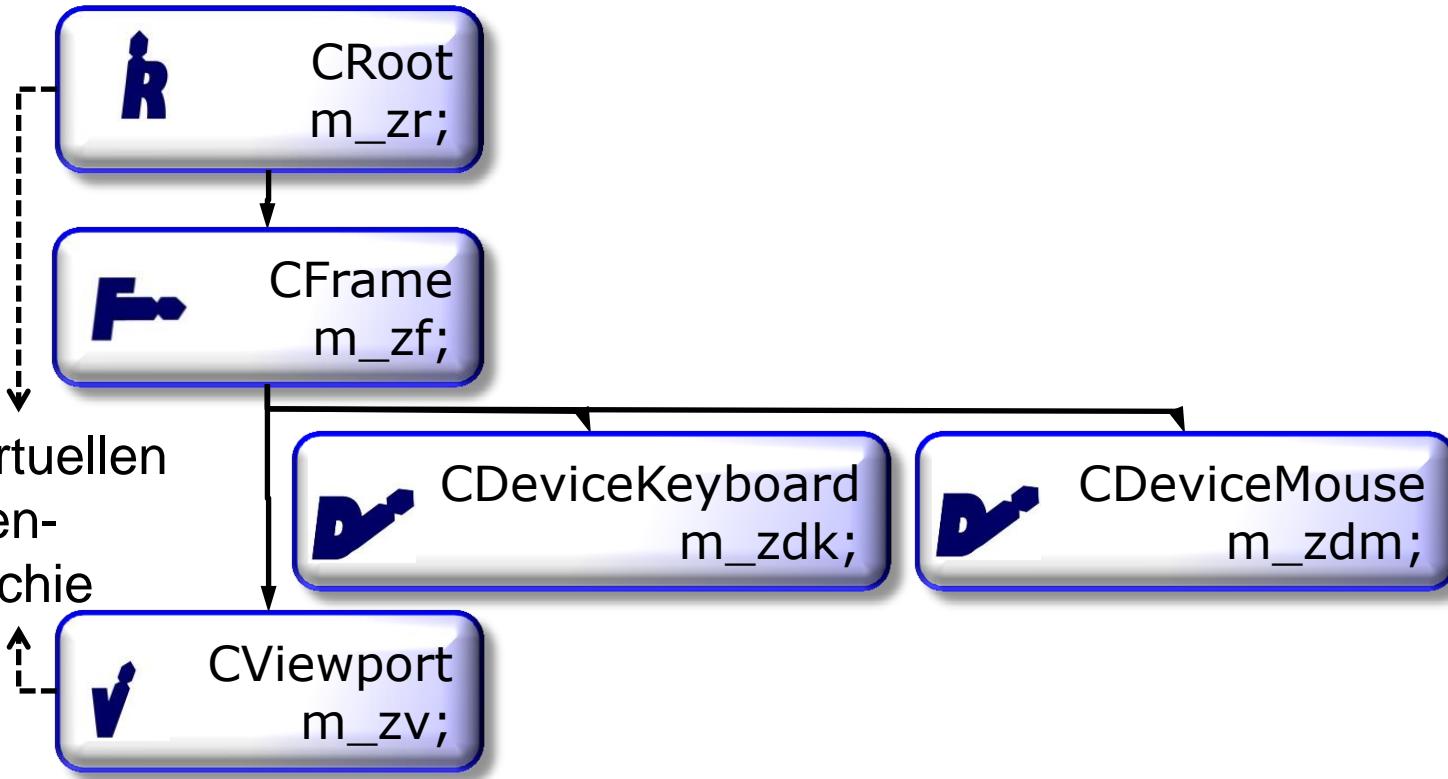
Abgeleitete Klassen von CDevice



Beispiel für Devicehierarchie



- 1 // / / /
- 2 // / / /
- 3 // / / / Zur virtuellen Szenenhierarchie
- 4 // / / /
- 5 // / / /



Add-Methoden des Frames für Eingabe

Hat man den Frame für die Eingabe initialisiert, muss noch das jeweilige Device angehangen werden, dafür gibt es folgende vier Methoden in CFrame:

1 // / / /
2 // / / /
3 // / / /
4 // / / /
5 // / / /

```
void AddDeviceKeyboard  
    (CDeviceKeyboard * pdevicekeyboard);  
void AddDeviceMouse  
    (CDeviceMouse * pdevicemouse);  
void AddDeviceGameController  
    (CDeviceGameController * pdevicegamecontroller);  
void AddDeviceCursor  
    (CDeviceCursor * pdevicecursor);
```



Kapitel 3

Kapitel 3



1 // / / /

2 // / / /

/// **TASTATUR**



4 // / / /

5 // / / /





Tastatureingabearten

Tastatureingabearten

Assynchron (bisher kennengelernt)

Warte, bis eine Taste gedrückt ist, und stelle eine Meldung in die Message-Queue für die Applikation!
Geeignet für Texteingabe

Synchron (Direct Input)

Meldet der Applikation sofort, wenn eine Taste gedrückt ist.

Geeignet für Games und VR-Interaktion



CDeviceKeyboard::PlaceWASD

bool PlaceWASD(CPlacement & zp, float & fTimeDelta,
bool bEarth);

1 //
2 //
3 //
4 //
5 //

Diese Methode der Klasse CInputX implementierte eine komplette WASD-Steuerung. Sehr praktisch!

fTimeDelta ist diejenige Zeit, die seit dem letzten Frame vergangen ist.

Gibt **true** aus, wenn Leertaste gedrückt wurde. (z.B. für Shooter)

Placement der Kamera, wird von der Funktion automatisch jedes Frame befüllt

true, wenn Szene auf der Erde spielt (Horizont bleibt dann horizontal), im Weltall **false**.





Tastatursteuerung

CDeviceKeyboard::PlaceWASD



bool PlaceWASD(CPlacement & zp, float & fTimeDelta,
 bool bEarth);



Tasten der WASD-Steuerung:

A: nach links

W: nach vorne

S: nach hinten

D: nach rechts

R: nach oben

F: nach unten

Rechtspfeil: Rechtsdrehung (yaw right)

Linkspfeil: Linksdrehung (yaw left)

Aufpfeil: Hochdrehung (pitch up)

Abpfeil: Runterdrehung (pitch down)

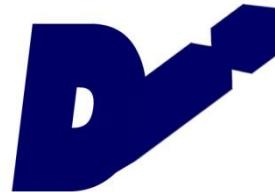
Leertaste: Programmabhängige Sonderaktion





Tastatursteuerung

CDeviceKeyboard, WASD-Hilfsmethoden



void SetWASDTranslationSensitivity (float fTranslationSensitivity);
float GetWASDTranslationSensitivity ();

Setzt bzw. holt die Verschiebungs-Empfindlichkeit
(Translationsgeschwindigkeit) bei den WASD-Tasten in
Einheiten pro Sekunde für die Methode PlaceWASD

void SetWASDRotationSensitivity (float fRotationSensitivity);
float GetWASDRotationSensitivity();

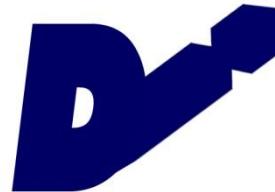
Setzt bzw. holt die Drehempfindlichkeit
(Rotationsschnelligkeit) bei den Pfeil-Tasten in Bogenmaß
pro Sekunde für PlaceWASD





Tastatursteuerung

CDeviceKeyboard, WASD-Hilfsmethoden



Erzeugt Kameraminimalhöhenlevel für die Erdsteuerung,
die Kamera kann nie unter fyLevelMin untertauchen.

void SetWASDLevelMin(float fyLevelMin);
void SetWASDLevelMax(float fyLevelMax);

Erzeugt Kameramaximalhöhenlevel für die Erdsteuerung,
die Kamera kann nie über fyLevelMax steigen.

float GetWASDLevelMin();
float GetWASDLevelMax();

Und natürlich gibt's dazu noch die
entsprechenden Hol-Methoden.





CDeviceKeyboard::KeyPressed



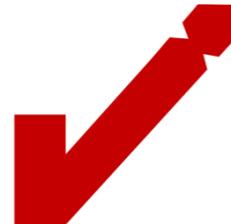
```
bool KeyPressed(int iKey);
```

true, falls Taste iKey (Tastaturnummer gemäß DirectX) gedrückt wurde, ansonsten false

Beispiel:

```
if(keyPressed(DIK_K))
{
    // geht hier rein, wenn Taste „K“ gedrückt wurde
}
if(keyPressed(DIK_K))
{
    // geht hier rein, wenn Taste „K“ gedrückt wurde
}
```





Tastatursteuerung Übung



- 1 // / / /
- Programmieren Sie eine WASD-Steuerung für ihre Erde!

2 // / / /

Freiwillige Zusatzaufgabe für die schnellen Nerds:

- 3 // / / /
- Nach Drücken der Leertaste soll sich die Erde gemäß WASD bewegen, nochmaliges Drücken bewegt wieder die Kamera.

4 // / / /

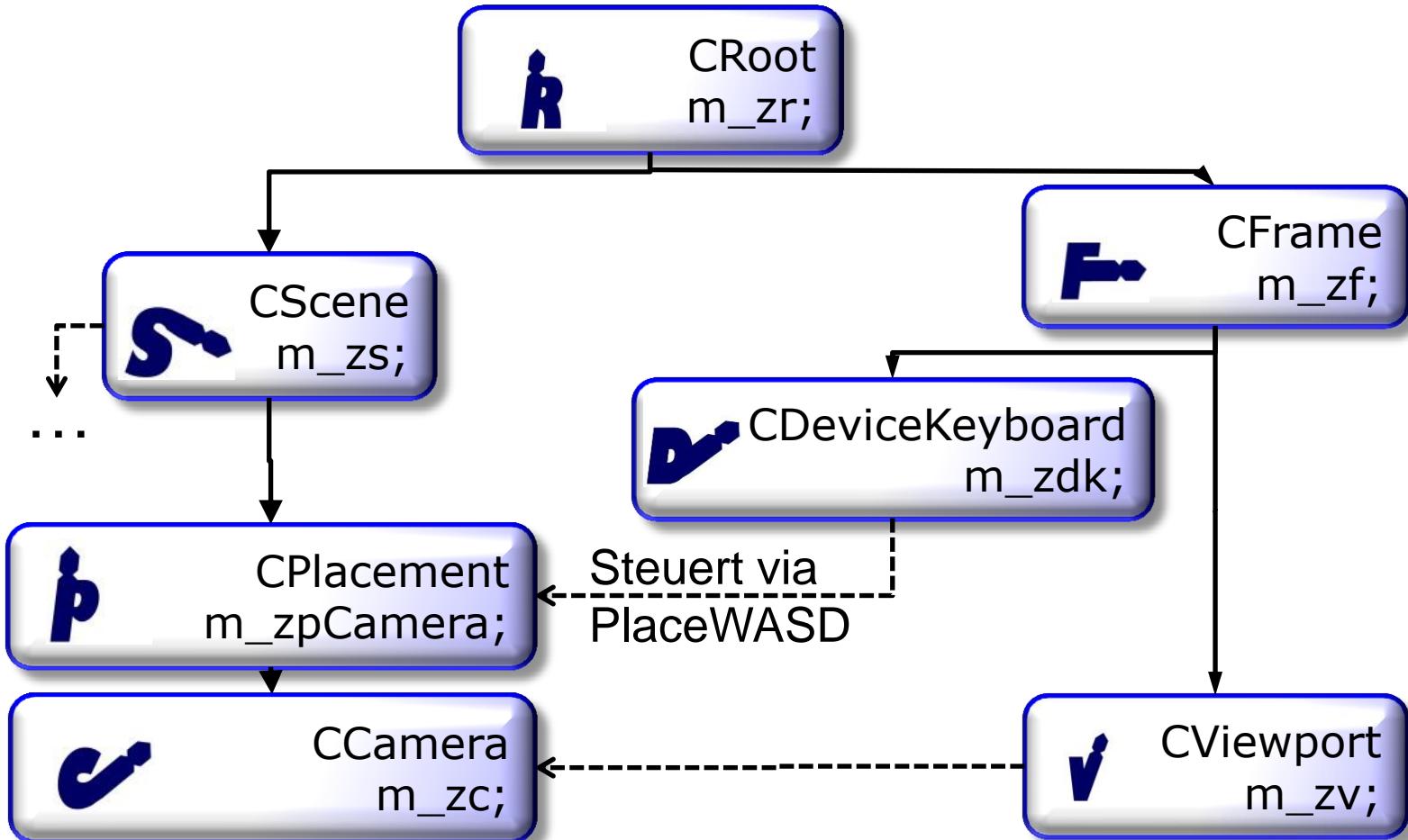
5 // / / /





Lösung Tastatursteuerung

1 // / / /
2 // / / /
3 // / / /
4 // / / /
5 // / / /





Lösung Tastatursteuerung (2/5)

Vektoria-Objekte in CGame.h



1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /

```
CRoot m_zr;  
CScene m_zs;  
CPlacement m_zpCamera;  
CPlacement m_zpSphere;  
CGeoSphere m_zgSphere;  
CFrame m_zf;  
CViewport m_zv;  
CCamera m_zc;  
CParallelLight m_zl;  
CMaterial m_zm;  
CDeviceKeyboard m_zdk;
```





Lösung Tastatursteuerung (3/5)

Init-Methode in CGame.cpp



```
void CGame::Init(HWND hwnd,
                  HWND hwndDX, RECT rectWnd)
{
    m_zf.Init(hwnd, rectWnd.right, rectWnd.bottom,
              eRenderApi_DirectX11_shadermode150,
              eInputApi_DirectInput);
    m_zv.InitFull(&m_zc);
    m_zr.AddFrameHere(&m_zf);
    m_zf.AddViewport(&m_zv);
    m_zf.AddDeviceKeyboard(&m_zdk);
    m_zm.MakeTextureImage
        ("textures\\earth_image.jpg");
    m_zl.Init(CHVector(1,0,1),CColor(1,1,1));
    ...
}
```

1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /





Lösung Tastatursteuerung (4/5)

Init-Methode in CGame.cpp



```
...
m_zr.AddScene(&m_zs);
m_zs.AddPlacement(&m_zpCamera);
m_zs.AddParallelLight(&m_zl);
m_zs.AddPlacement(&m_zpSphere);
m_zpCamera.AddCamera(&m_zc);
m_zpSphere.Translate(CHVector(0,0,-3));
m_zpSphere.AddGeo(&m_zgSphere);
m_zgSphere.Init(2.0F,&m_zm);
}
```





Lösung Tastatursteuerung (5/5)

Tick-Methode in CGame.cpp



```
void CGame::Tick(float fTimeDelta)
{
    m_zdk.PlaceWASD(m_zpCamera, fTimeDelta);
    m_zr.Tick(fTimeDelta);
}
```



Kapitel 4

Kapitel 4



1 // / / /

2 // / / /

3 // / / /

/// MAUS 6 // CURSOR



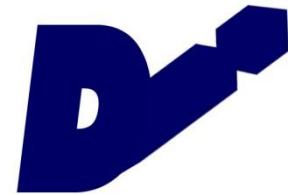
5 // / / /



PROF. DR. TOBIAS BREINER
HS KEMPTEN

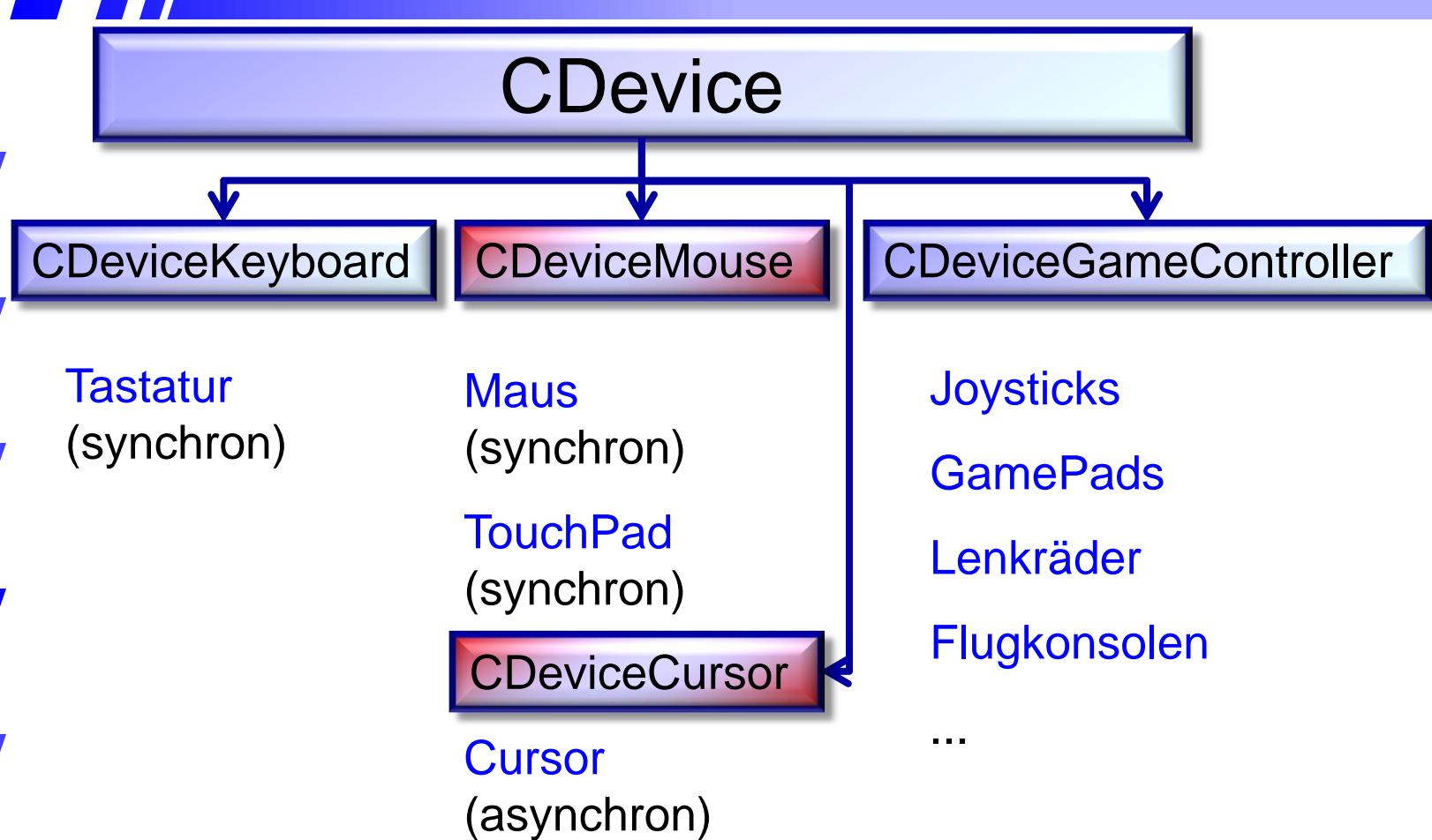
23 VON 49
DEVICES





Ein- und Ausgabegeräte

Maus und Cursor



Unterschied zw. Maus und Cursor



Achtung! Maus und Cursor werden häufig miteinander verwechselt.

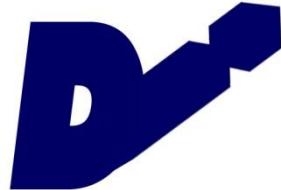
Während eine Maus jedoch ein reales Eingabegerät ist, stellt ein Cursor eine zeichnerische Interpretation der Mausbewegungen dar.



In Vektoria wird dieser Umstand durch zwei verschiedene Device-Klassen Rechnung getragen:
CDeviceMouse und **CDeviceCursor**

Wenn Sie absolute Cursorkoordinaten bezüglich des Fensters haben wollen, dann verwenden Sie CDeviceCursor
Wenn es auf schnelle relative Mausbewegungen ankommt, verwenden Sie CDeviceMouse





Mausbewegungslesemethoden

```
float GetRelativeX(void);
// Gibt relative Mausbewegung in X-Richtung aus
float GetRelativeY(void);
// Gibt relative Mausbewegung in Y-Richtung aus
float GetRelativeZ(void);
// Gibt relative Mausradbewegung (Z-Richtung) aus

float GetAbsoluteX(void);
// Gibt die absolute Mausbewegung auf der X-Achse
aus (nicht zu verwechseln mit Cursorpos!)
float GetAbsoluteY(void);
// Gibt die absolute Mausbewegung auf der y-Achse
aus (nicht zu verwechseln mit Cursorpos!)
```



Synchr. Mausklicklesemethoden

```
bool ButtonPressed(int a_iButton);
// true, falls Taste iButton (Tastennummer gemäß
// DirectX) gedrückt wurde, ansonsten false
(synchron)
```





Parametrisierungsmethoden

1 //
void SetSensitivity(float fSensitivity=0.0005f);
// Setzt die Empfindlichkeit

2 //
float GetSensitivity(void);
// Gibt die Empfindlichkeit aus

3 //
void SetInvertedYAxisOn();
// Y-Achse invertieren

4 //
void SetInvertedYAxisOff();
// Y-Achse nicht invertieren

5 //





Übung Mauseingabe



- 1 // / / /
- Lassen Sie die Erdkugel zusätzlich durch die Maus drehen!

2 // / / /

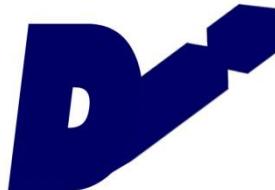
Freiwillige Zusatzaufgabe für die schnellen Nerds:

- 3 // / / /
- Implementieren Sie eine progressive Maussteuerbewegung, d.h. erst bewegt sich die Erde langsam, hat sie aber einen gewissen Drall, dann immer schneller!

- 4 // / / /
- Implementieren Sie eine progressive Maussteuerbewegung, d.h. erst bewegt sich die Erde langsam, hat sie aber einen gewissen Drall, dann immer schneller!

- 5 // / / /
- Implementieren Sie eine progressive Maussteuerbewegung, d.h. erst bewegt sich die Erde langsam, hat sie aber einen gewissen Drall, dann immer schneller!

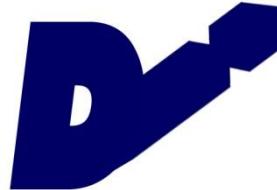




Cursorpositionslesemethoden

```
bool GetAbsolute(int & ix, int & iy,  
                 bool bHideCursor = false);  
// Gibt die zeigerpositionskoordinaten bezüglich  
// der linken oberen Ecke des Frames aus, gibt  
// true aus, wenn Cursor sich innerhalb des Frames  
// befindet  
bool GetFractional(float & frx, float & fry,  
                    bool bHideCursor = false);  
// Gibt die fraktionalen zeigerpositions-  
// koordinaten (Wertebereich jeweils 0..1)  
// bezüglich des Frames aus, gibt true aus, wenn  
// Cursor sich innerhalb des Frames befindet.  
// Ist HideCursor true, dann verschwindet er, wenn  
// er sich im Frame befindet => Man kann mit  
// Overlays seinen eigenen Cursor erzeugen
```





Assynchr. Mausklicklesemethoden

1 //
bool ButtonPressed(int iButton);
// true, falls Taste iButton gedrückt wurde,
ansonsten false

2 //
bool ButtonPressedLeft(); // true, falls linke
Maustaste gedrückt wird, ansonsten false
bool ButtonPressedRight(); // true, falls rechte
Maustaste gedrückt wird, ansonsten false
bool ButtonPressedMid(); // true, falls mittlere
Maustaste gedrückt wird, ansonsten false

3 //
4 //
5 //





Übung Cursor ersetzen



- 1 // / / /
- Ersetzen Sie den Windows-Cursor durch Ihren eigenen!

2 // / / /

Freiwillige Zusatzaufgabe für die schnellen Nerds:

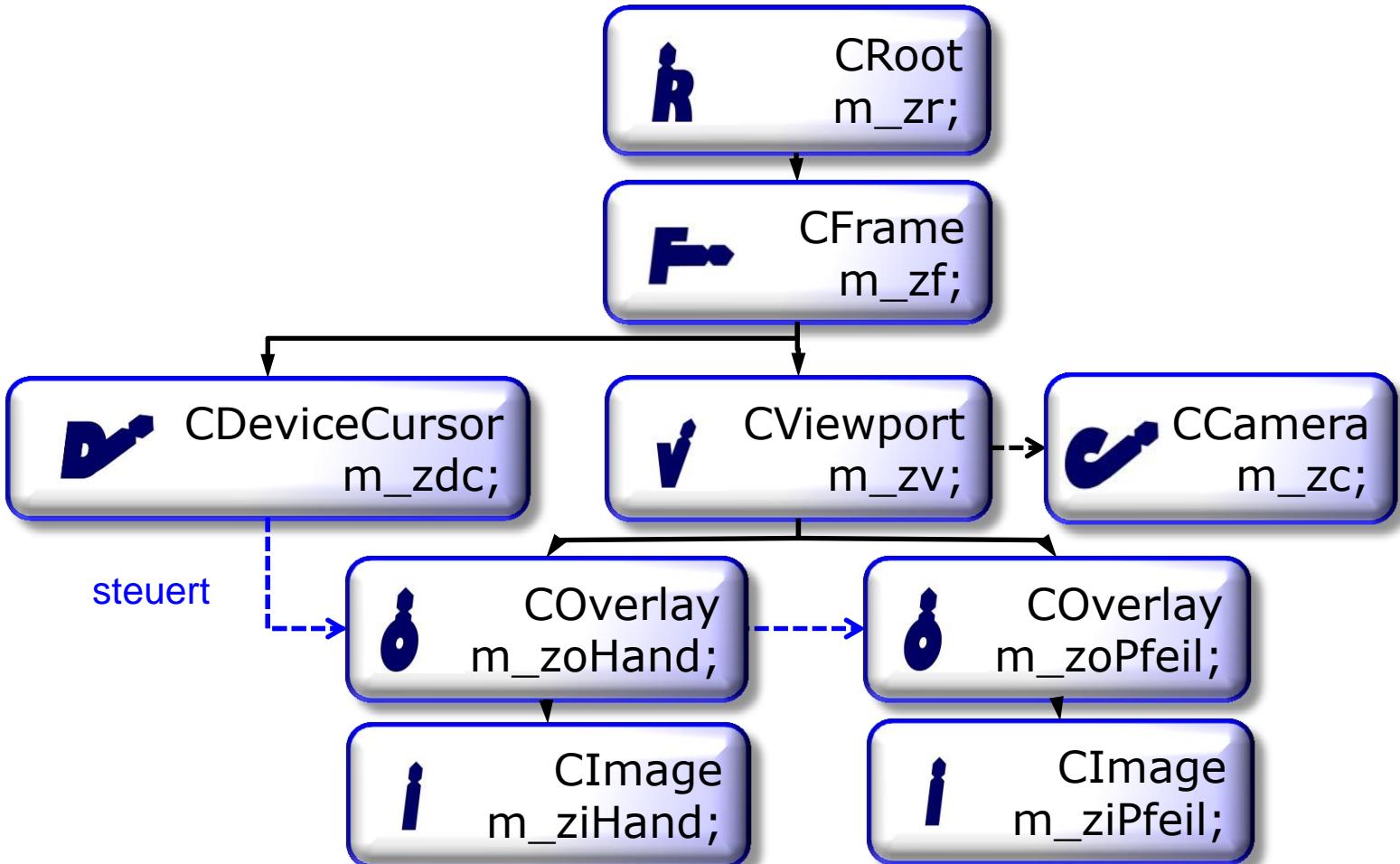
- 3 // / / /
- Wenn die linke Maustaste gedrückt ist, soll eine Hand erscheinen ansonsten ein Pfeil!

4 // / / /

5 // / / /



Lösung Cursor Ersetzen





Nerd-Lösung Cursor Ersetzen (2/5)

Vektoria-Objekte in CGame.h



```
CRoot m_zr;  
CFrame m_zf;  
CDeviceCursor m_zdc;  
CViewport m_zv;  
CCamera m_zc;  
CImage m_ziPfeil;  
CImage m_ziHand;  
COVERLAY m_zoPfeil;  
COVERLAY m_zoHand;
```

1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /





Nerd-Lösung Cursor Ersetzen (3/5)

Init-Methode in CGame.cpp



```
m_zf.Init(hwnd, rectwnd.right,  
          rectwnd.bottom);  
m_ziPfeil.Init("Pfeil.bmp");  
m_ziHand.Init("Hand.bmp");  
  
m_zoPfeil.Init(&m_ziPfeil,  
               CFloatRect(0.0F,0.0F,0.02F,0.02F),true);  
m_zoHand.Init(&m_ziHand,  
               CFloatRect(0.0F,0.0F,0.02F,0.02F),true);  
m_zv.InitFull(&m_zc);  
m_zr.AddFrameHere(&m_zf);  
m_zf.AddViewport(&m_zv);  
m_zf.AddDeviceCursor(&m_zdc);  
m_zv.AddOverlay(&m_zoPfeil);  
m_zv.AddOverlay(&m_zoHand);
```

Diese Bilder müssen natürlich noch mit Gimp oder Photoshop erzeugt und in den Projektordner gelegt werden.





Nerd-Lösung Cursor Ersetzen (4/5)

Tick-Methode in CGame.cpp



```
float fx, fy;
m_zdc.GetFractional(fx,fy,true);
if(m_zdc.ButtonPressedLeft())
{
    m_zoPfeil.Switchoff(); m_zoHand.Switchon();
    m_zoHand.SetRect(
        CFloatRect(fx-0.01F,fy-0.01F,0.02F,0.02F));
}
else
{
    m_zoPfeil.Switchon();m_zoHand.Switchoff();
    m_zoPfeil.SetRect(
        CFloatRect(fx-0.01F,fy-0.01F,0.02F,0.02F));
}
m_zr.Tick(fTimeDelta);
```

1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /





Nerd-Lösung Cursor Ersetzen (5/5)

ReSize-Methode in CGame.cpp



```
void CGame::WindowReSize  
    (int iNewwidth, int iNewHeight)  
{  
    m_zf.ReSize(iNewwidth, iNewHeight);  
}
```



Wichtig!

Man muss dem Frame in WindowReSize die Fenstergröße übermitteln, sonst wird der neue Cursor nicht angezeigt! Ursache: Der Frame gibt die Fenstergröße an den angehängten DeviceCursor weiter dieser berechnet daraus die Cursorposition, die er zum Anzeigen braucht.

1 // / / /

2 // / / /

3 // / / /

4 // / / /

5 // / / /



Picking

Picking allgemein

1 // / / / Picking bezeichnet das Auswählen von Objekten mit Hilfe des Cursors – in diesem Falle das Auswählen von 3D-Objekten.

2 // / / / Dazu geht man mit der Maus über ein Objekt und drückt auf eine Maustaste – normalerweise die linke Maustaste.

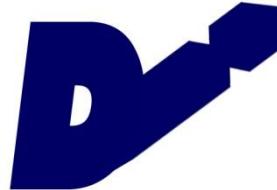
3 // / / / Picking wird nur von wenigen Szenegrafen und Engines unterstützt.

4 // / / /

5 // / / /



Picken des Viewports in Vektoria



In Vektoria findet man die Picking-Methoden in der Klasse CDeviceCursor. Zum Picken des Viewports gibt es folgende Methoden:

1 //
1 Cviewport * Pickviewport();

2 //
2 // Gibt erstes sichtbares Viewport aus, welches
2 // unter dem Cursor liegt, falls kein Viewport
2 // gefunden wurde, ist das Ergebnis NULL

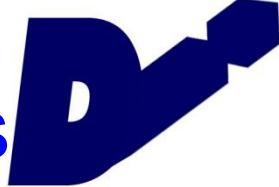
3 //
3 Cviewport * Pickviewport(float & frxviewport,
3 // float & fryviewport);

4 //
4 // Gibt erstes sichtbares Viewport aus, welches
4 // unter dem Cursor liegt, falls kein Viewport
4 // gefunden wurde, ist das Ergebnis NULL

5 //
5 // Zusätzlich werden die passenden
5 // Bildschirmkoordinaten frxviewport und fryviewport
5 // des Picking-Punktes mit ausgegeben



Picken von Overlays und Kameras



Zum Picken eines Overlays gibt es folgende Methode:

1 //|//|//|
coverlay * PickOverlay();
// Gibt oberstes sichtbares Overlay aus,
// welches unter dem Cursor liegt, falls kein
// Overlay gefunden wurde ist das Ergebnis NULL

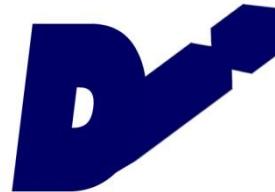
2 //|//|//|
Und zum Picken einer Kamera folgende Methode:

3 //|//|//|
CCamera * PickCamera();
// Gibt die Kamera aus, die mit dem sichtbaren
// Viewport verbunden ist, welches unter dem Cursor
// liegt, falls kein Viewport gefunden wurde ist das
// Ergebnis NULL



Picking

Picken einer Szene



Zum Picken einer Szene gibt es folgende Methode:

1 //|/|/| cScene * PickScene();
// Gibt die Szene aus, welche das sichtbare Viewport
// zeigt, welches unter dem Cursor liegt, falls
// keine Szene gefunden wurde ist das Ergebnis NULL

2 //|/|/|

3 //|/|/|

4 //|/|/|





Picking

Picken von Placements



Zum Picken eines oder mehrerer Placements gibt es folgende Methoden:



1 CPlacement * PickPlacement();

// Gibt dasjenige nächste Placement mit der
// niedersten Hierarchiestufe aus, welches unter dem
// Cursor liegt. Falls kein Placement gefunden wurde,
// ist das Ergebnis NULL.



2 CPlacements * PickPlacements();

// Gibt alle Placements aus, welche unter dem
// Cursor liegen. Falls kein Placement gefunden
// wurde, ist das Ergebnis NULL.



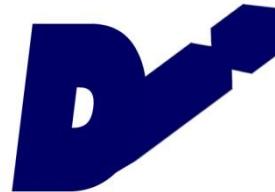
3 Die Placements werden anhand ihrer jeweiligen
Bounding-Box gefunden.





Picking

Picken von Geometrien



Zum Picken einer Geometrie gibt es folgende Methode:

1 //
CGeo * PickGeo(CHVector & vIntersection,
float & fDistanceSquare);

2 //
// Gibt dasjenige Geo mit der niedersten Hierarchie-
// stufe aus, welches unter dem Cursor liegt. Falls
// kein Geo gefunden wurde, ist das Ergebnis NULL.
// vIntersection ist der genaue Schnittpunkt,
// fDistanceSquare ist das Quadrat der Entfernung
// zwischen Strahlursprung und Schnittpunkt.

3 //
4 //
5 //

Diese Methode dauert lange. Sie kann bei großen
Geometrien durchaus einige Millisekunden dauern.
Sie sollte daher nur dann angewandt werden, wenn exaktes
Picking zwingend notwendig ist, ansonsten sollte einfach das
darüberliegende Placement gepickt werden.



Kapitel 5

Kapitel 5



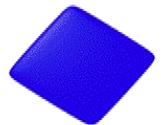
1 // / / /

2 // / / /

3 // / / /

4 // / / /

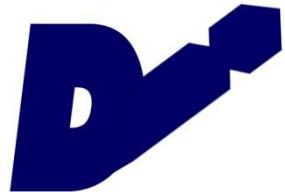
/// GAME /// CONTROLLER



PROF. DR. TOBIAS BREINER
HS KEMPTEN

44 VON 49
DEVICES





Game Controller

- 1 // / / /
- 2 // / / /
- 3 // / / /
- 4 // / / /
- 5 // / / /

Joysticks



JoyPads



Lenkräder



Tanzmatten



Lightguns



Motorad-
controller

Flugsteuerungen



CDeviceGameController



Lesemethoden

```
float GetRelativeX(void);
// Gibt relative Controllerbewegung
// in X-Richtung aus
float GetRelativeY(void);
// Gibt relative Controllerbewegung
// in Y-Richtung aus
float GetRelativeZ(void);
// Gibt relative Controllerbewegung
// in Z-Richtung aus

bool ButtonPressed(int a_iButton);
// true, falls Taste iButton (Tastennummer gemäß
// DirectX) gedrückt wurde, ansonsten false
```





Parametrisierungsmethoden

```
void SetSensitivity(float fSensitivity=0.0005f);  
// Setzt die Empfindlichkeit  
  
float GetSensitivity(void);  
// Gibt die Empfindlichkeit aus  
  
void SetInvertedYAxisOn();  
// Y-Achse invertieren  
  
void SetInvertedYAxisOff();  
// Y-Achse nicht invertieren
```



Übung Raumflugsteuerung



- Programmieren Sie eine intuitive Raumflugsteuerung für ein Game Controller Ihrer Wahl!

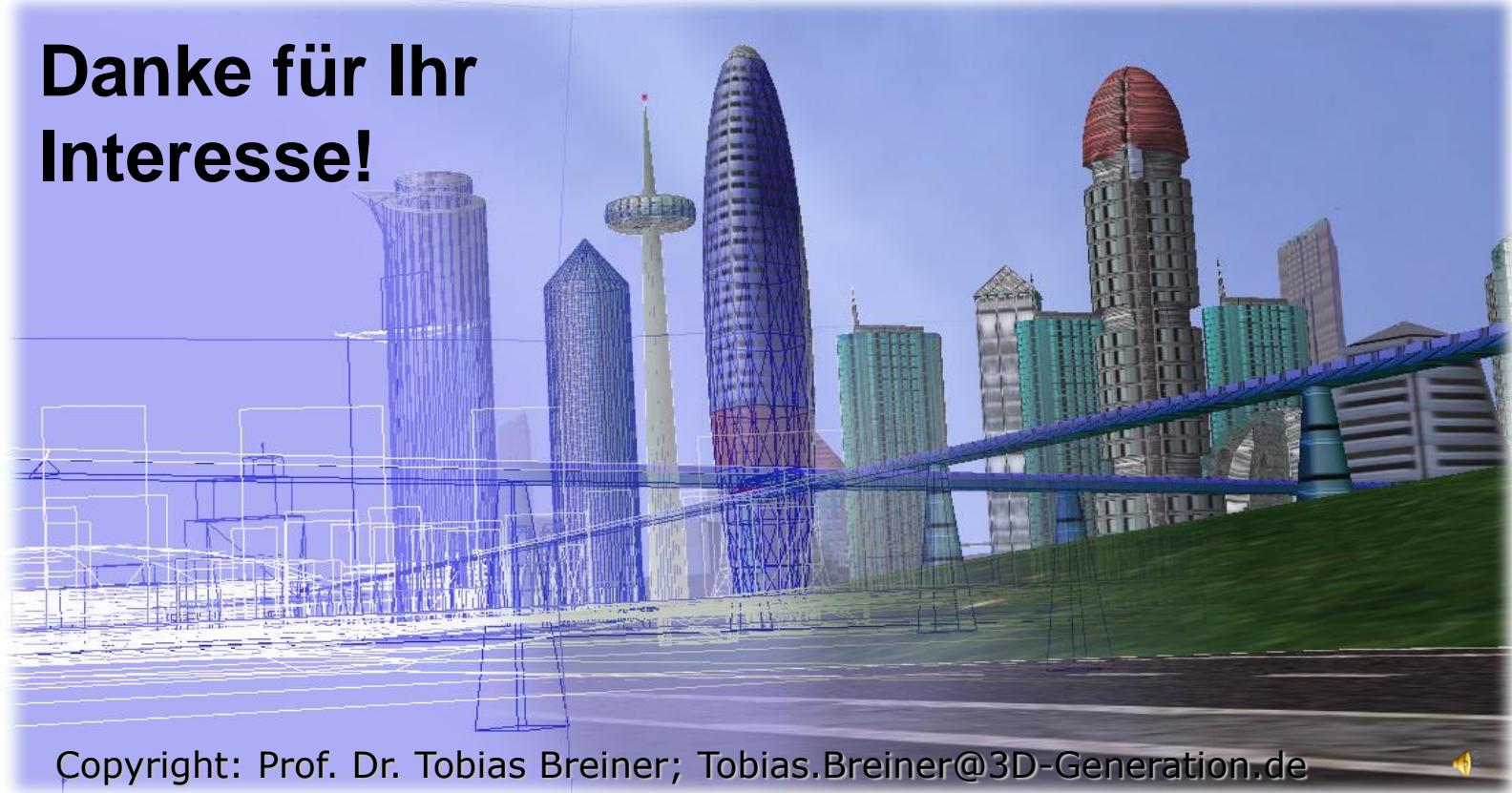
Freiwillige Zusatzaufgabe für die schnellen Nerds:

- Fügen Sie der Flugsteuerung Beschleunigung und Abbremsung hinzu!



|||||GAME OVER

Danke für Ihr
Interesse!



Copyright: Prof. Dr. Tobias Breiner; Tobias.Breiner@3D-Generation.de

