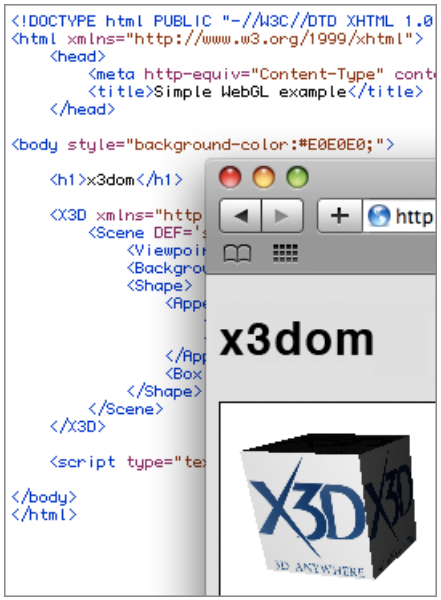
Deklarative 3D-Grafik im Webbrowser

X3DOM – Deklaratives (X)3D in HTML5



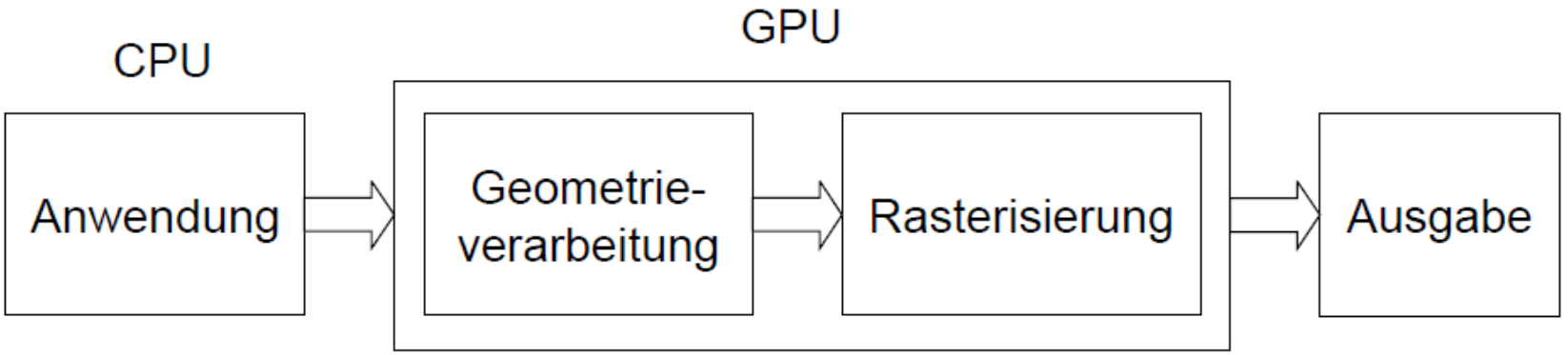
Yvonne Jung
Abt. Virtuelle und Erweiterte Realität
Fraunhofer IGD
Fraunhoferstraße 5
64283 Darmstadt

Tel.: +49 6151 155 290
yvonne.jung@igd.fraunhofer.de
www.igd.fraunhofer.de

3D-Grafikprogrammierung bisher

- Voraussetzungen
  - Gute Programmierkenntnisse in C/C++
  - Erfahrung mit hardwarenahen Grafik-Programmierschnittstellen
    - Z.B. OpenGL oder Direct3D
  - Mathematik
  - Mathematik
  - Mathematik
  - …
- Problem
  - Programmierung erfordert einschlägiges Studium und Berufserfahrung

Exkurs: Echtzeit-Rendering
Die 3D-Grafik-Pipeline

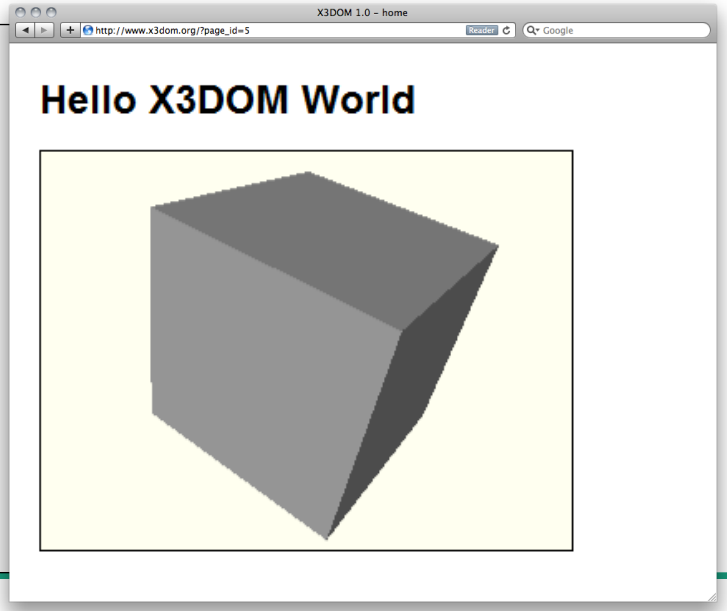


- Drei Hauptstufen: Anwendung (auf CPU), Geometrie, Rasterisierung (auf GPU, d.h. Grafikkarte)
- Anwendungsebene: Repräsentation der 3D-Daten (z.B. als Szenengraph), Kollisionsbehandlung, View Frustum Culling,…
- Geometriestufe: Modell-/ Kameratransformation, Projektion, Clipping,…
- Rasterisierung: Triangle-Setup, Scankonvertierung, Pixel-Shading (z.B. Texturierung), Per-Fragment Operationen (z.B. Depth-Test)

X3DOM: (Deklaratives) 3D in HTML5

- HTML erlaubt deklarative Beschreibung von Inhalten ohne Programmierung – bisher allerdings nur für Text und 2D-Multimedia
  - Webtechniken sind einfach zu lernen, gut dokumentiert, weit verbreitet usw.
- Das X3DOM-System erweitert diese Techniken noch um 3D-Elemente
  - Erlaubt die nahtlose Integration von 3D-Inhalten in HTML-Seiten

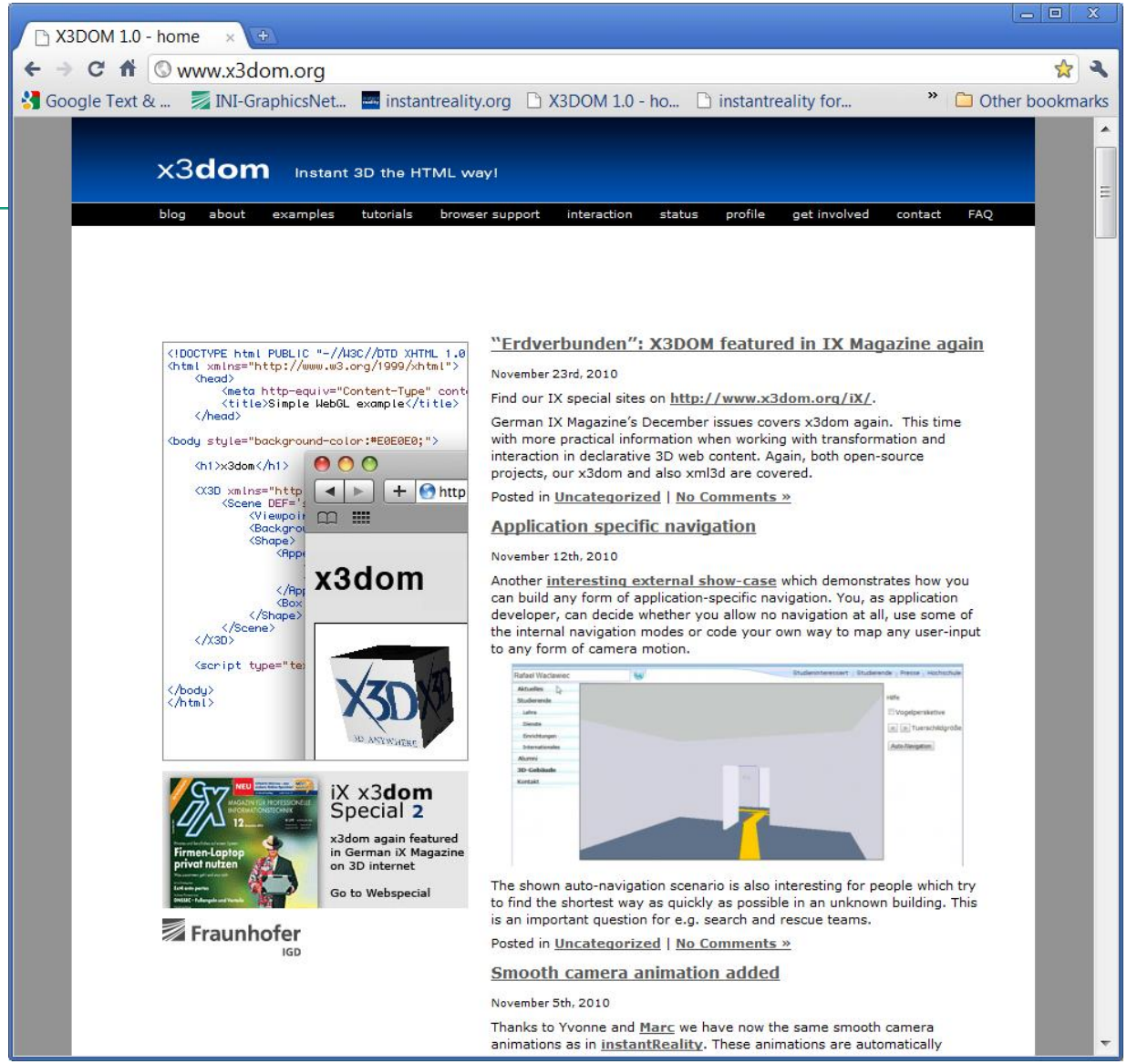
```
<html>
  <body>
    <h1>Hello X3DOM World</h1>
    <x3d>
      <scene>
        <shape>
          <box></box>
        </shape>
      </scene>
    </x3d>
  </body>
</html>
```



X3DOM

www.x3dom.org

- 3D im Webbrowser
  - Verwendet intern u.a. JavaScript und WebGL
- System-Voraussetzungen
  - Gamer-Grafikkarte (z.B. von NVidia)
  - Aktueller Browser
    - Firefox Beta
    - Chrome Beta



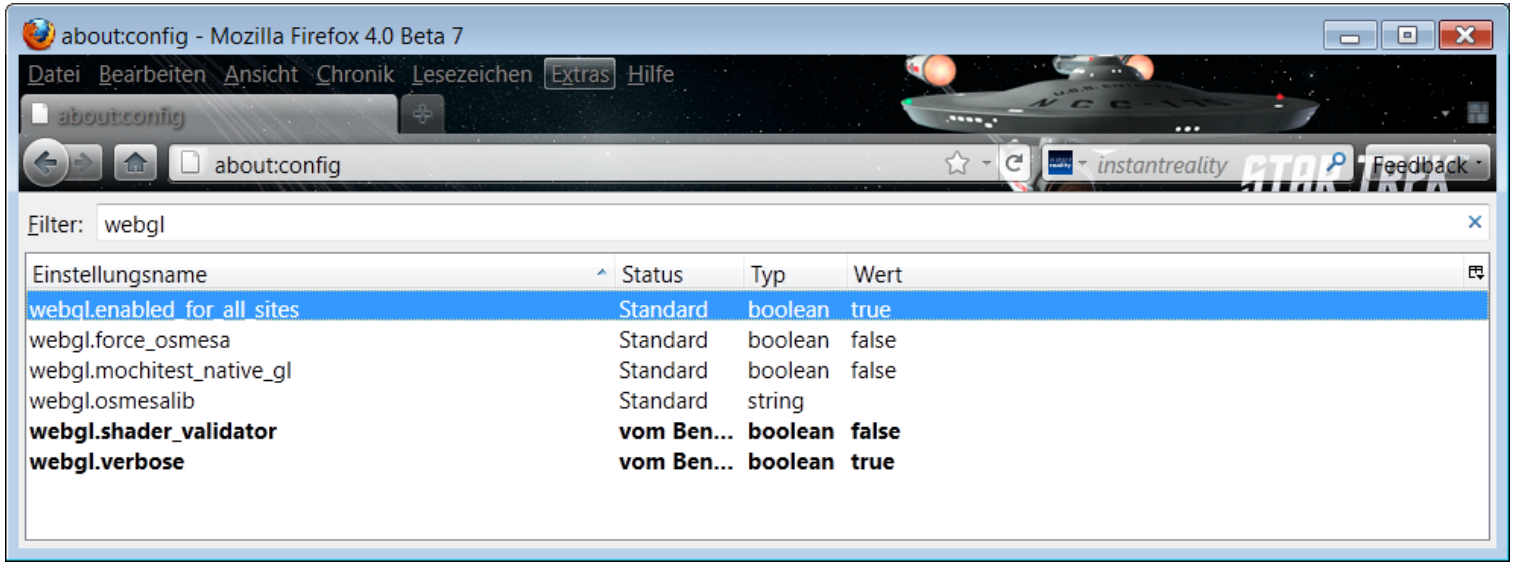
Erster Schritt
Den richtigen Webbrowser installieren

- 3D im Internet funktioniert derzeit nur in ganz aktuellen Browsern
  - → In den Beta-Versionen von Firefox 4 und Google Chrome
  - Achtung: Beide Browser befinden sich noch in der Entwicklung, d.h. es kann sich jederzeit noch etwas verändern!
    - Deshalb immer auf aktuelle Versionen achten (Browser und X3DOM)

- Firefox 4 Beta (empfohlen) gibt es zum Download hier:
  - http://www.mozilla.com/de/firefox/beta/
  - Beim ersten Starten in der Adressleiste “about:config” eingeben und dort unter Filter “webgl” eintippen
    - Darauf achten, dass der Wert von “webgl.enabled_for_all_sites” auf “true” steht (evtl. Ändern durch Doppelclick)
    - Außerdem “webgl.shader_validator” noch auf “false” setzen

Erster Schritt

Den richtigen Webbrowser installieren

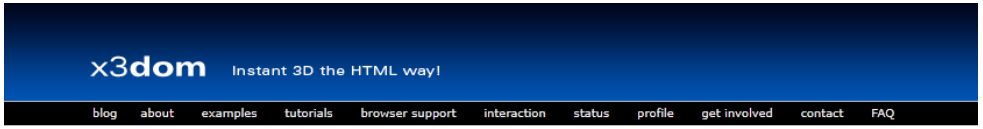


- Anschließend Firefox 4 Beta sicherheitshalber neu starten

- Google Chrome Beta gibt es zum Download hier
  - http://www.google.com/chrome/eula.html?extra=betachannel
  - Hier muss zunächst unter Eigenschaften (Rechtsklick) unter “Ziel” noch folgendes hinzugefügt werden: *--enable-webgl --use-gl=desktop*

Zweiter Schritt
Testen, ob es geht

- Auf x3dom.org gehen und im Menü “about” anklicken
  - Oder direkt diesem Link folgen: http://www.x3dom.org/?page_id=2
- Die neue Webseite sollte dann etwa so aussehen:



- **Hinweise zur Navigation**
  - Mit gedrückt gehaltener Maustaste kann man die 3D-Szene bewegen (innerhalb des Rechtecks)
  - Drehen mit linker Maustaste, verschieben mit mittlerer, zoomen mit rechter Taste

Fraunhofer IGD

about

X3DOM (pronounced X-Freedom) is an experimental open source framework and runtime to support the **ongoing discussion** in the Web3D and W3C communities how an integration of HTML5 and declarative 3D content could look like. It tries to fulfill the current HTML5 specification for **declarative 3D content** and allows including **X3D** elements as part of any HTML5 DOM tree.

The goal here is to have a live X3D scene in your HTML DOM, which allows you to manipulate the 3D content by only adding/ removing or changing DOM elements. No specific plugin or plugin interface (like **SAI**) are needed. It also supports some of the HTML events (like “onclick”) on 3D objects. The whole integration model is still evolving and open for discussions.

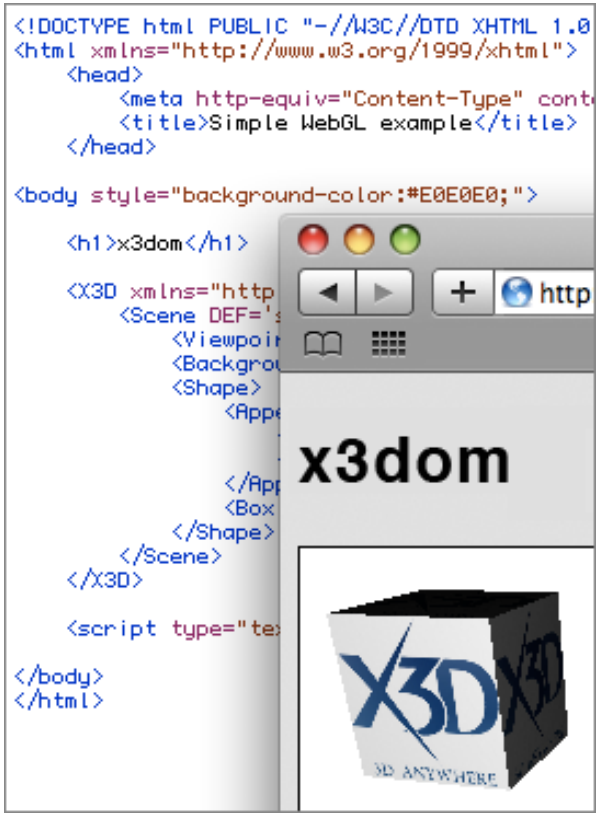
We hope to trigger a process similar to how the SVG in HTML5 integration evolved:

- Provide a vision and runtime today to experiment with and furthermore develop an integration model for declarative 3D in HTML5
- Get the discussion in the HTML5 and X3D communites going and evolve the system and integration model
- Finally it would be part of the HTML5 standard and supported by every major browser natively

More architectural and background information can be found in the **X3DOM-paper** (published at the Web3D symposium 2009).

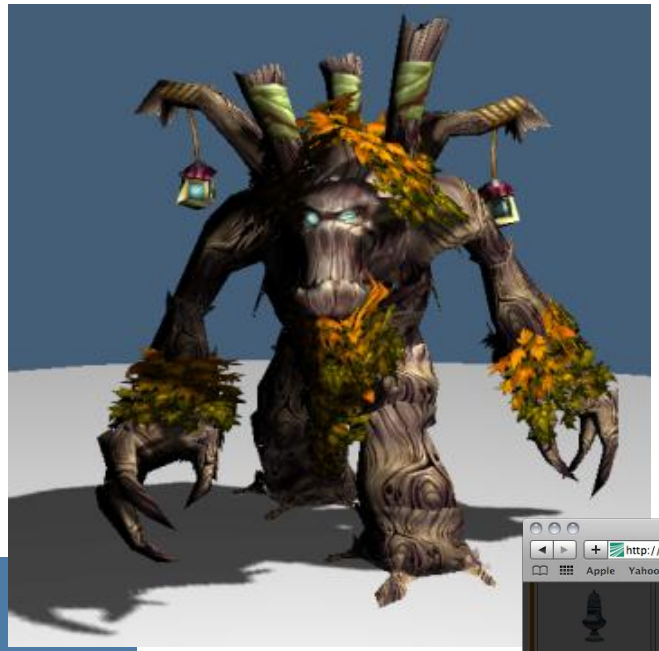
Alternatively you, as web-developer, can also just utilize the system today to build web-pages and applications, which include declarative (X)3D content that will be rendered hardware accelerated (thanks to **WebGL**) without the need for using any plugin.

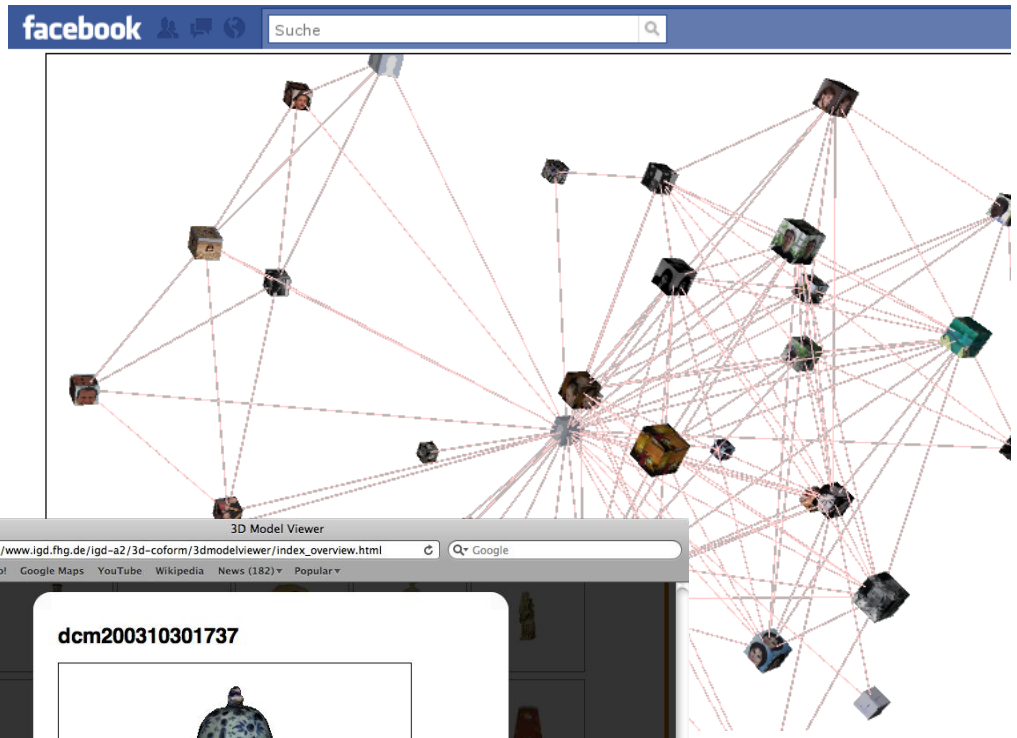
3D im Internet

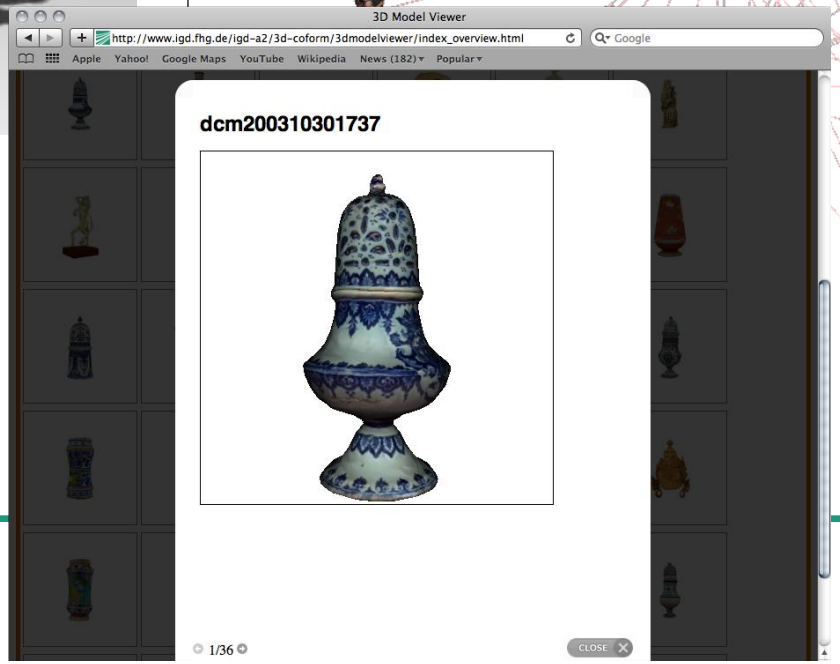


ANWENDUNGSBEISPIELE

Beispiel 1: Soziale Netzwerke, (eigene) Avatare, antike Gegenstände oder andere Informationen visualisieren



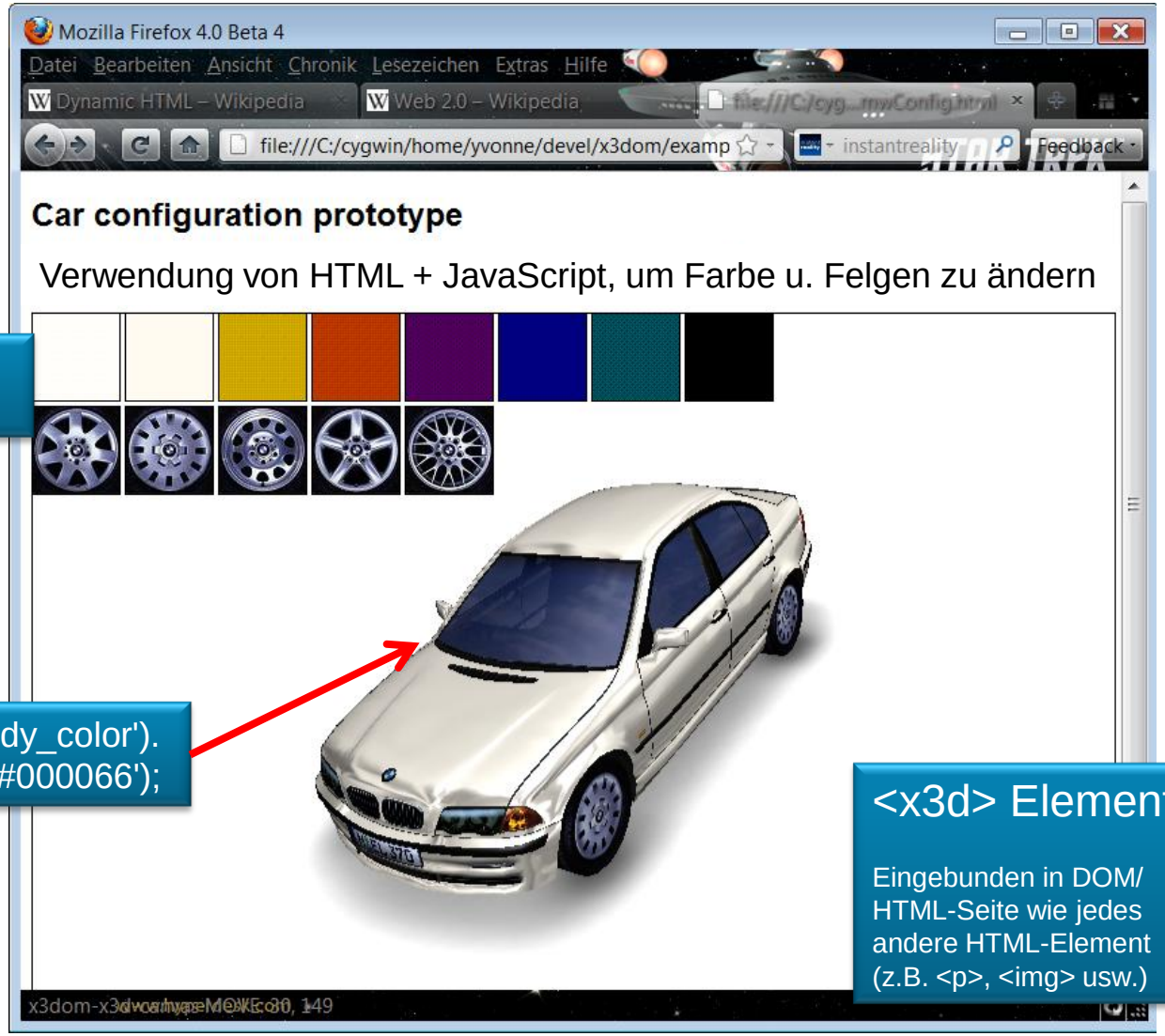




Fraunhofer
IGD

Beispiel 2: Einfacher interaktiver Car-Konfigurator

Interaktion über Standard-Webtechnologien (z.B. JavaScript-Events usw.)




```
<img src="felge1_64.jpg“
  onclick="..." style="…">
```

Klick auf <img> Element…

```
document.getElementById('body_color').
  setAttribute("diffuseColor", '#000066');
```

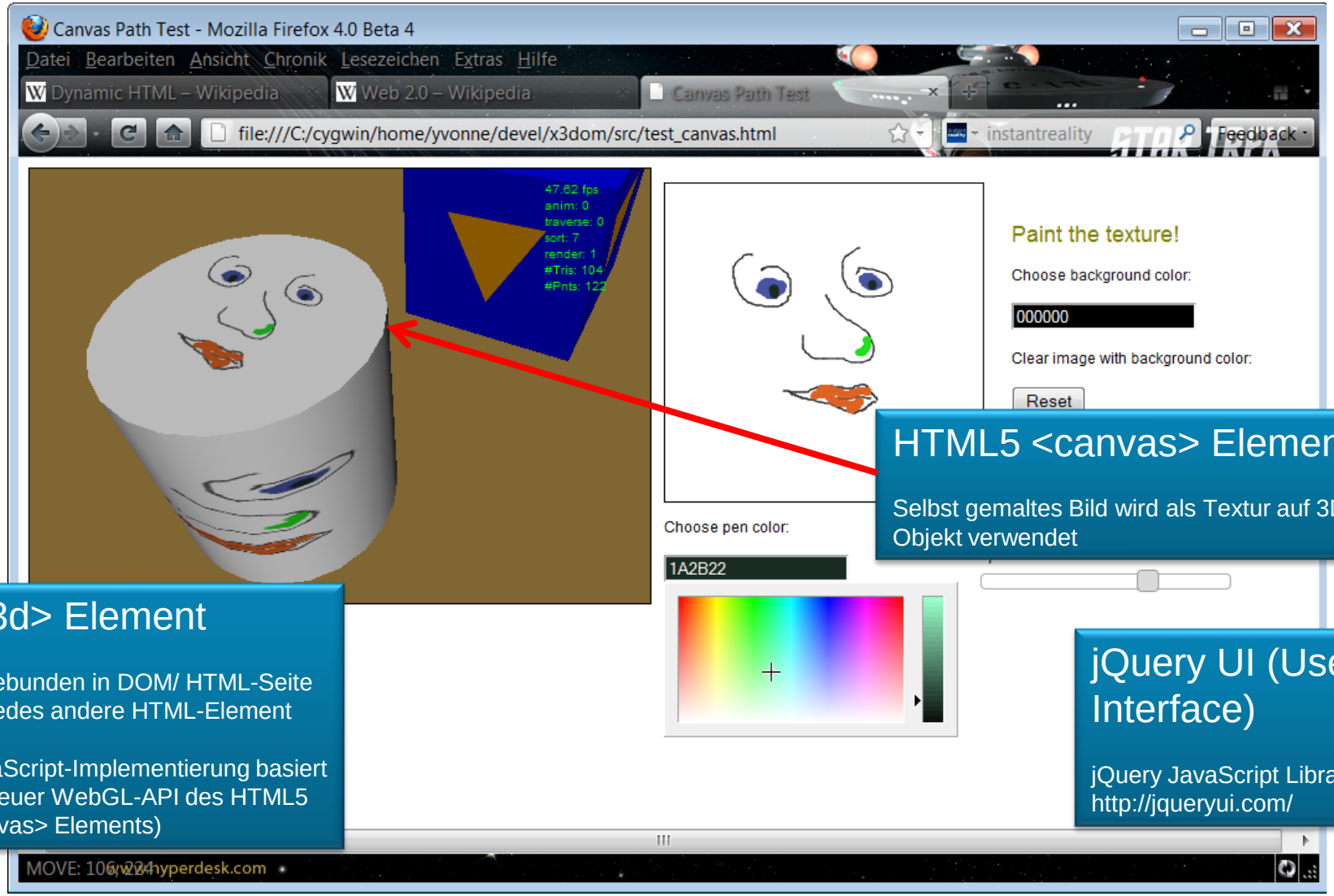
…verursacht Attribut-änderung von <texture> (d.h., andere Felgen sind nun zu sehen)

<x3d> Element

Eingebunden in DOM/ HTML-Seite wie jedes andere HTML-Element (z.B. <p>, <img> usw.)

Fraunhofer IGD

Beispiel 3: Texturen von 3D-Objekten selbst malen



HTML5 <canvas> Element

Selbst gemaltes Bild wird als Textur auf 3D-Objekt verwendet

<x3d> Element

Eingebunden in DOM/ HTML-Seite wie jedes andere HTML-Element

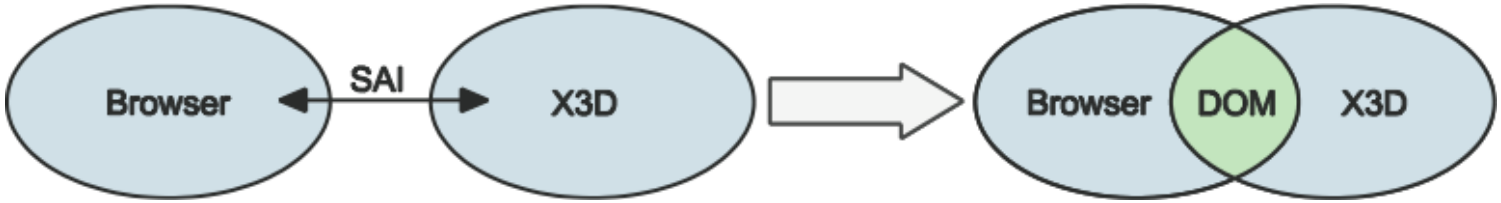
(JavaScript-Implementierung basiert auf neuer WebGL-API des HTML5 <canvas> Elements)

jQuery UI (User Interface)

jQuery JavaScript Library: http://jqueryui.com/

Fraunhofer IGD

Ein DOM-basiertes Modell für beschreibendes 3D in HTML5



X3DOM (X3D + DOM)

Exkurs: X3D (Extensible 3D)

- X3D ist ein offener ISO-Standard, der Format und Laufzeitverhalten von dynamischen, interaktiven 3D-Inhalten definiert
  - Nutzt sog. Szenengraph zur Beschreibung der 3D-Szene
  - Nachfolger von VRML – hat aber zusätzlich XML-Encoding
- Offizielle X3D-Spezifikation (auf Englisch), inklusive Knotendokumentation, Scripting-API usw.: http://www.web3d.org/x3d/specifications/
- Weitere Informationen (ebenfalls auf Englisch) gibt es auch hier: http://doc.instantreality.org/documentation/
- Empfehlenswerte Bücher über VRML und X3D:

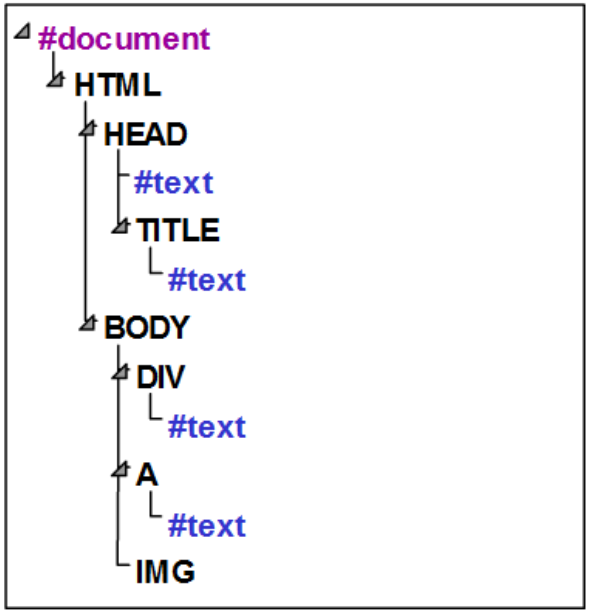
Das Einsteiger-seminar VRML

The Annotated VRML 97 Reference

X3D: Extensible 3D Graphics for Web Authors

Exkurs: DOM (Document Object Model)

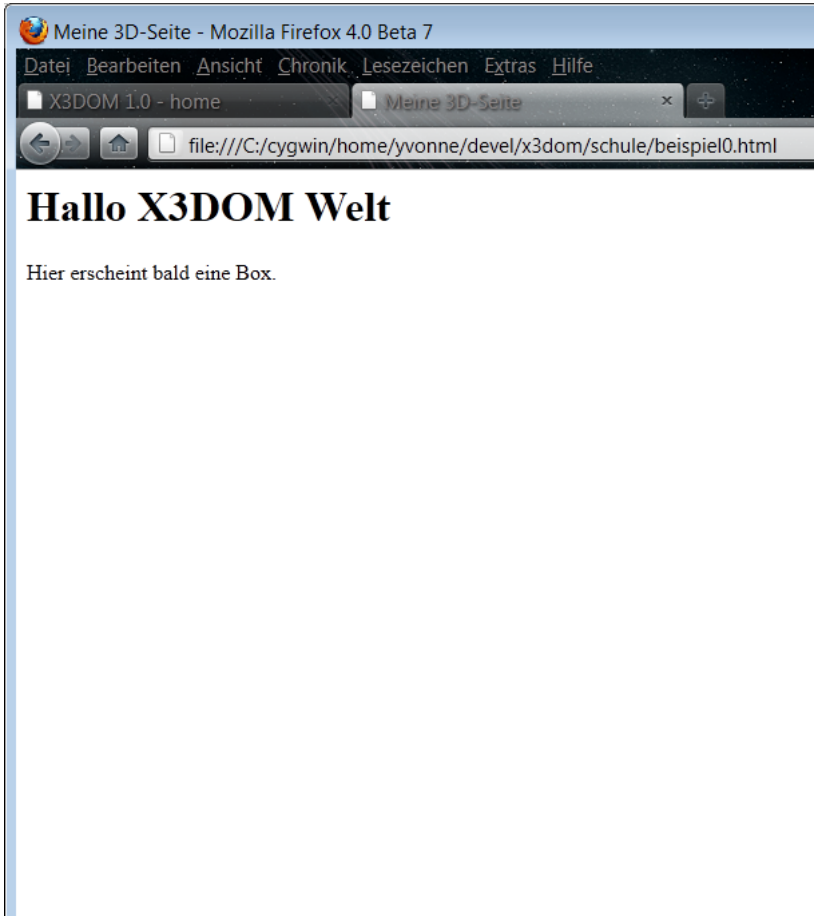
- Schnittstelle für den Zugriff auf (X)HTML- oder XML-Dokumente (vom W3C standardisiert)
- Baumstruktur mit unterschiedlichen Knoten (z.B. <h1>, <p>, <a>, <img>,…)
- HTML dient zur Beschreibung der Inhalte, DOM erlaubt Manipulation der Inhalte
  - Navigation über die einzelnen Knoten eines HTML-Dokuments (meist über JavaScript)
  - Knoten erzeugen, löschen, ändern
  - Auslesen und ändern von Attributen
- Tutorials und Beispiele zu HTML, CSS, JavaScript u. DOM-Scripting gibt es auf http://de.selfhtml.org/



DOM-Baumstruktur (Bsp.)

Kurze Wiederholung von HTML

```
<html>
    <head>
        <title>Meine 3D-Seite</title>
    </head>
    <body>
        <h1>Hallo X3DOM Welt</h1>
        <p>
            Hier erscheint bald eine Box.
        </p>
    </body>
</html>
```

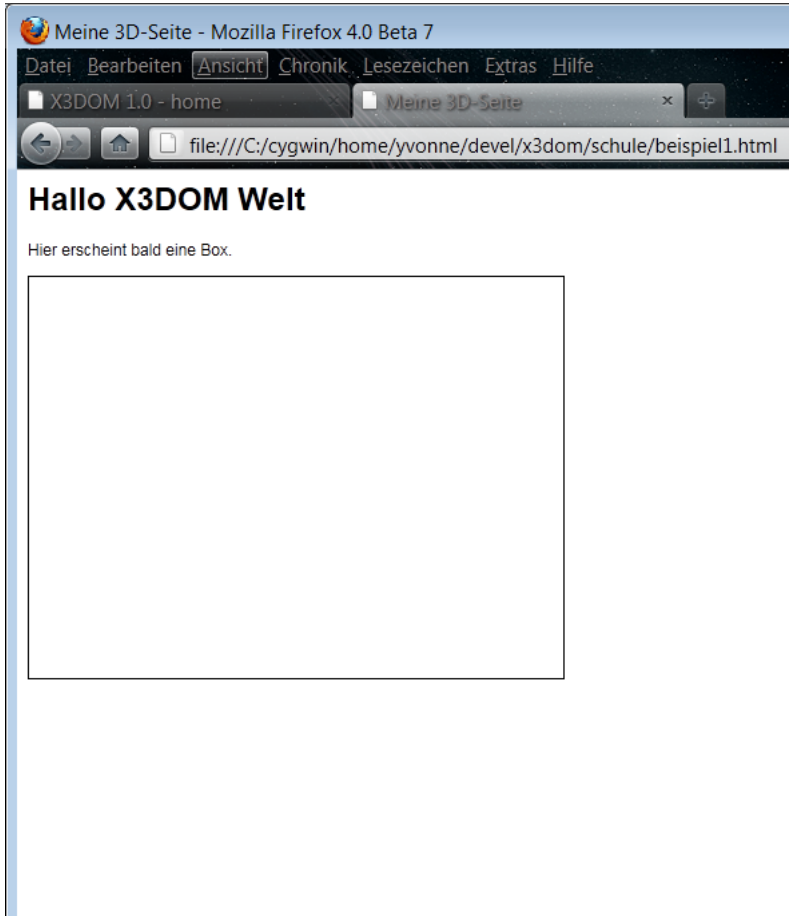


Jetzt muss HTML noch (X)3D lernen

```
<html>
  <head>
    <title>Meine 3D-Seite</title>
     <link rel="stylesheet" type="text/css"
          href="http://www.x3dom.org/x3dom/release/x3dom.css">
    </link>
    <script type="text/javascript"
          src="http://www.x3dom.org/x3dom/release/x3dom.js">
    </script>
  </head>
  …
```

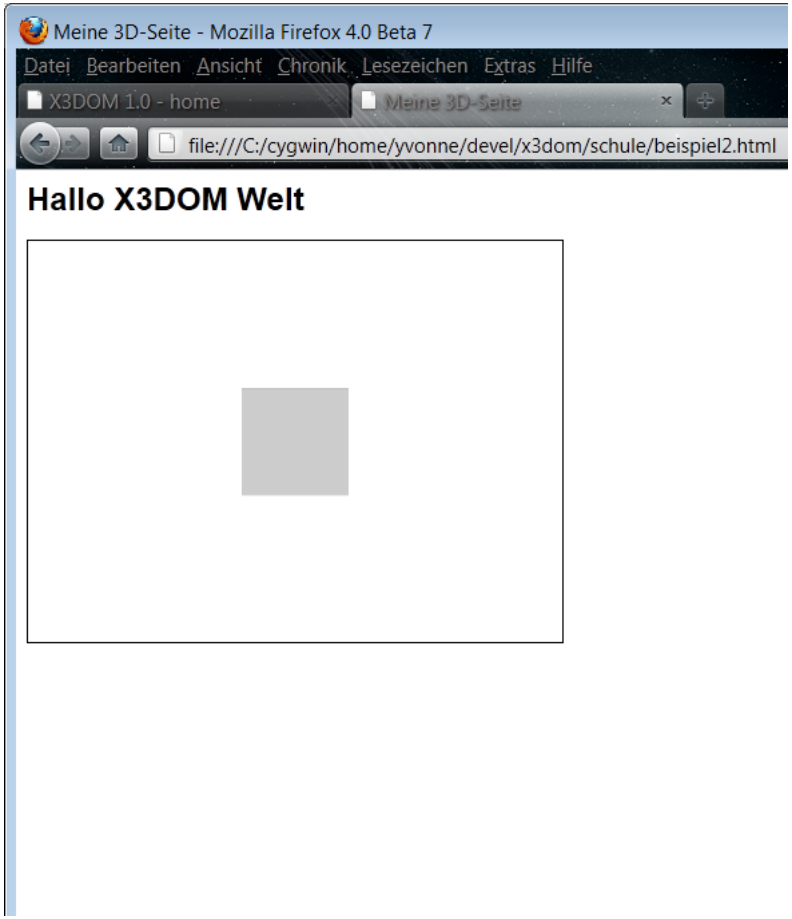
3D funktioniert nur innerhalb von <X3D>

```
…
    <body>
        <h1>Hallo X3DOM Welt</h1>
        <p>
            Hier erscheint bald eine Box.
        </p>
        <x3d width="400" height="300">
        </x3d>
    </body>
</html>
```



Alle 3D-Objekte hängen unter <scene>

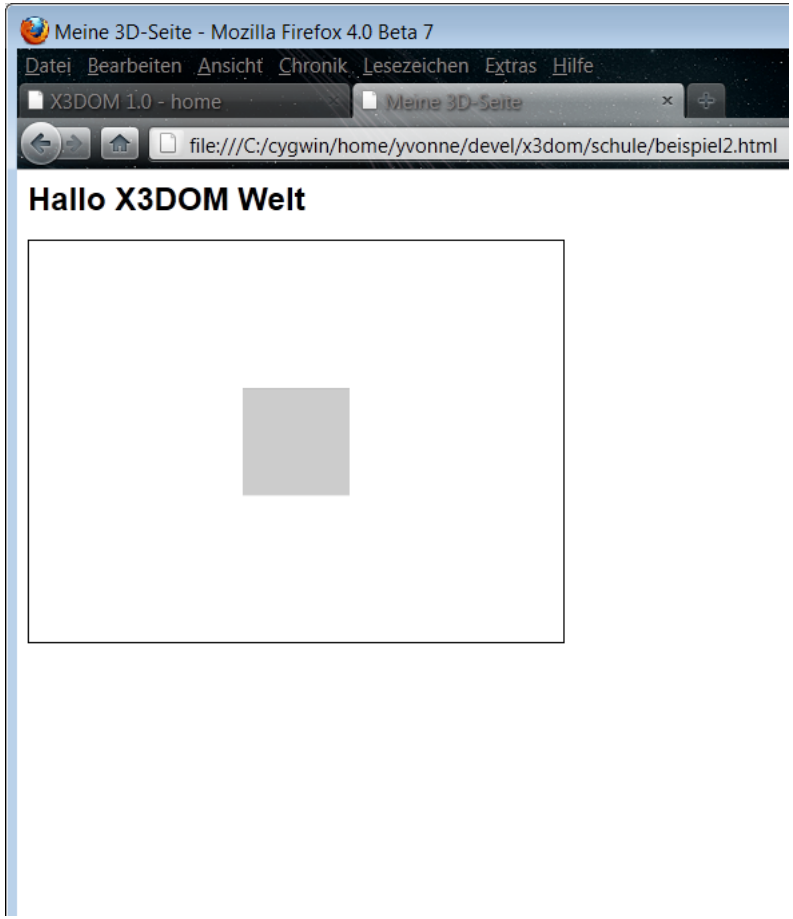
```
...
    <body>
        <h1>Hallo X3DOM Welt</h1>
        <x3d width="400" height="300">
            <scene>
                <shape>
                    <box></box>
                </shape>
            </scene>
        </x3d>
    </body>
</html>
```



Fraunhofer IGD

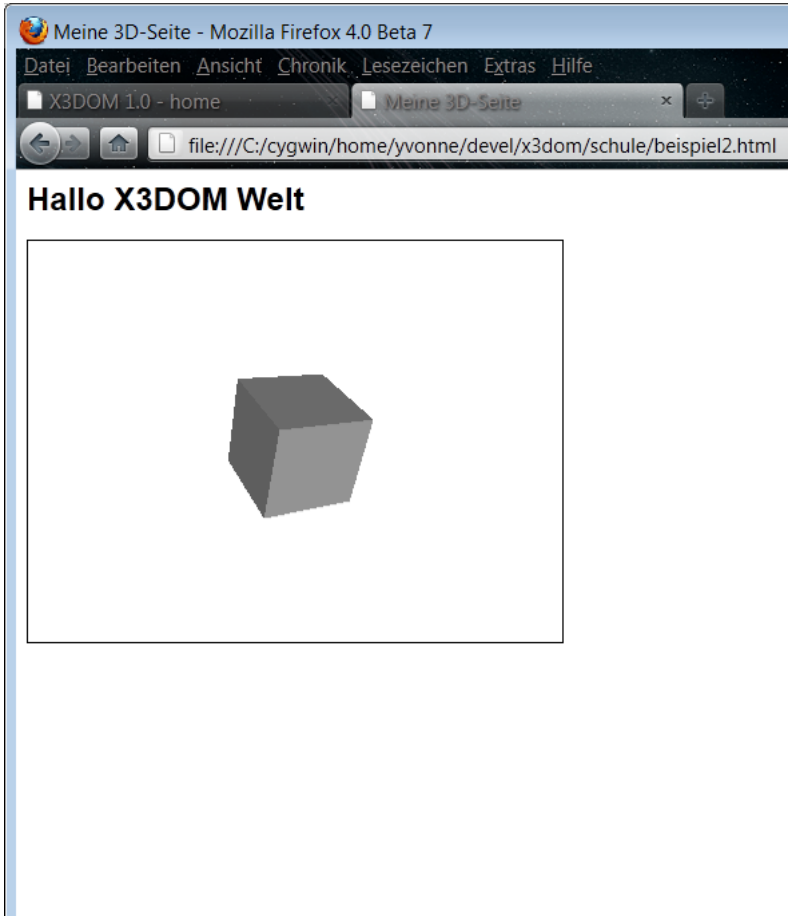
Jedes Objekt hat eine <shape> (Gestalt)

```
...
    <body>
        <h1>Hallo X3DOM Welt</h1>
        <x3d width="400" height="300">
            <scene>
                <shape>
                    <box></box>
                </shape>
            </scene>
        </x3d>
    </body>
</html>
```



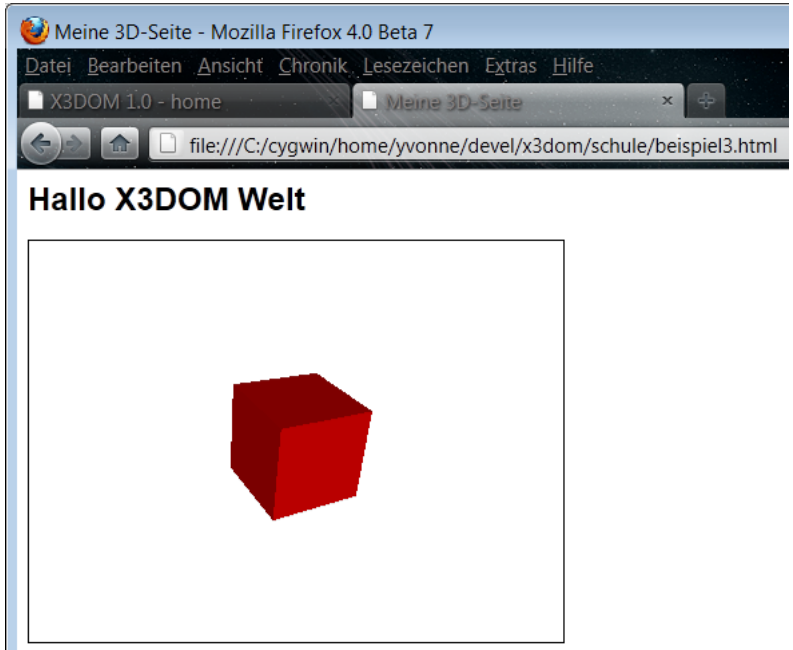
…und eine Geometrie, wie z.B. eine <box>

```
…
    <body>
        <h1>Hallo X3DOM Welt</h1>
        <x3d width="400" height="300">
            <scene>
                <shape>
                    <box></box>
                </shape>
            </scene>
        </x3d>
    </body>
</html>
```



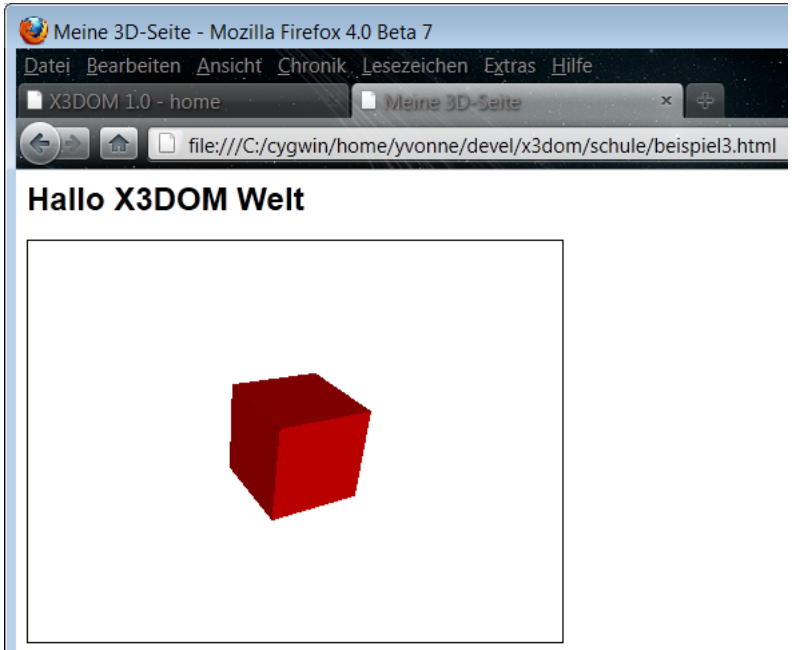
…sowie eine <appearance> (Aussehen)

```
<x3d width="400" height="300">
   <scene>
      <shape>
         <appearance>
            <material diffuseColor="red">
            </material>
         </appearance>
         <box></box>
      </shape>
   </scene>
</x3d>
```



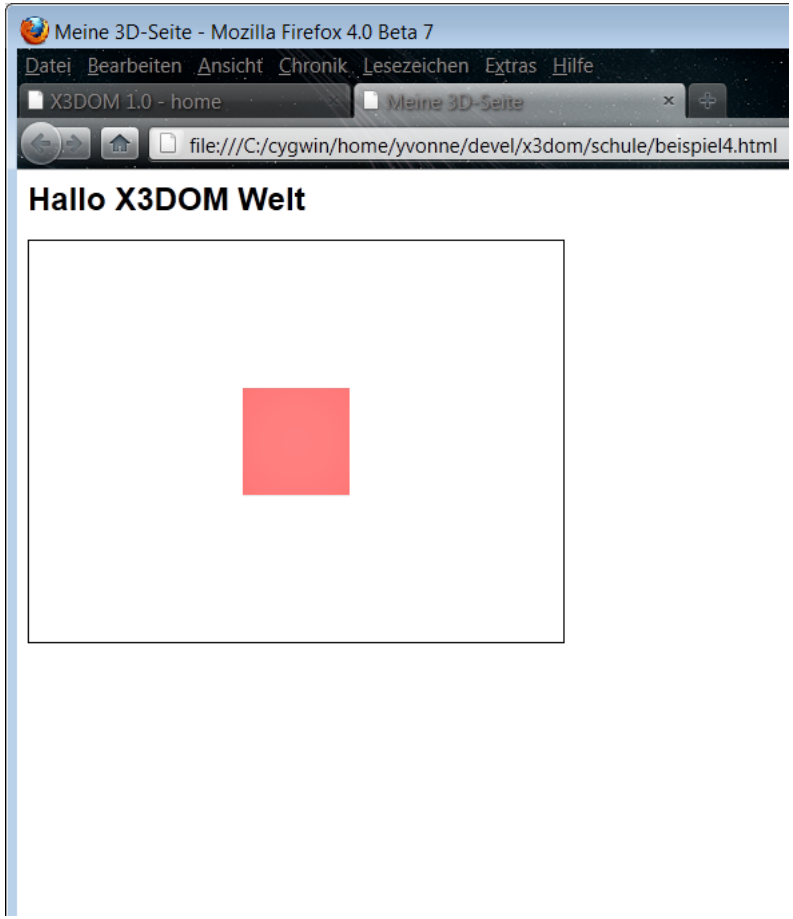
…mit einem (z.B. roten) <material>

```
<x3d width="400" height="300">
   <scene>
      <shape>
         <appearance>
            <material diffuseColor="red">
            </material>
         </appearance>
         <box></box>
      </shape>
   </scene>
</x3d>
```



Materialien mit Glanzlichtern versehen

```
<x3d width="400" height="300">
   <scene>
      <shape>
         <appearance>
            <material diffuseColor="red"
               specularColor="#808080">
            </material>
         </appearance>
         <box></box>
      </shape>
   </scene>
</x3d>
```

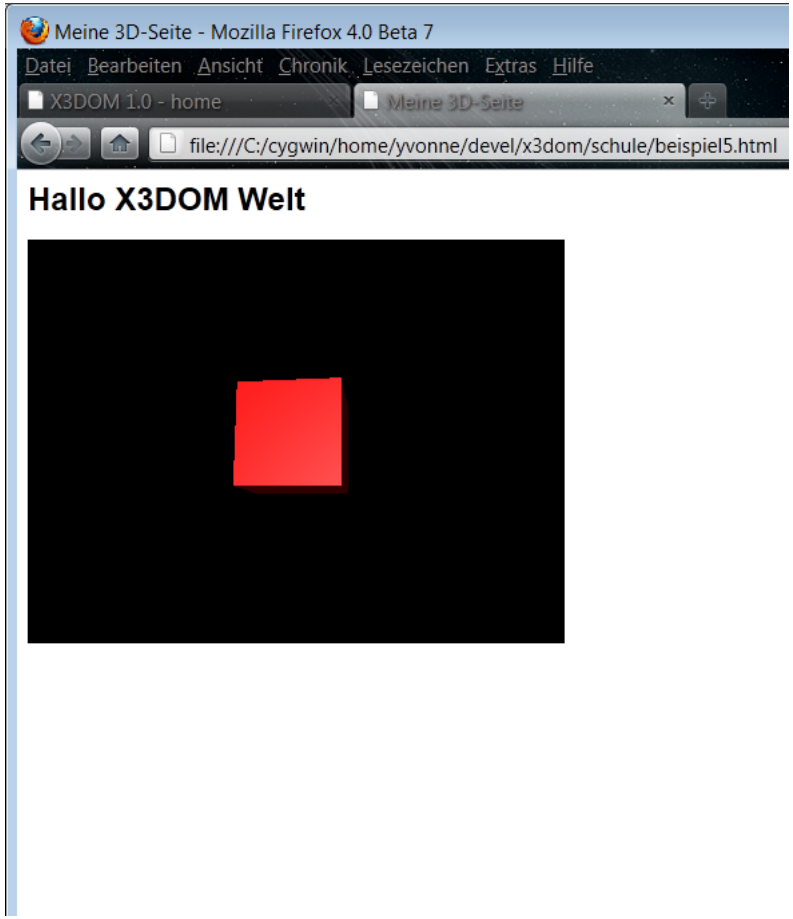


Fraunhofer IGD

Hintergrund ändern

Farbangabe in (R,G,B) mit Rot-/Grün-/Blau-Anteil ∈ [0,1]

```
<scene>
    <shape>
        <appearance>
            <material diffuseColor="red"
                specularColor="#808080">
            </material>
        </appearance>
        <box></box>
    </shape>
    <background skyColor="0 0 0">
    </background>
</scene>
```

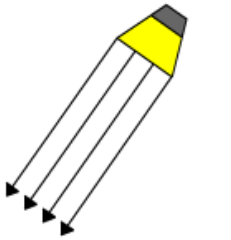


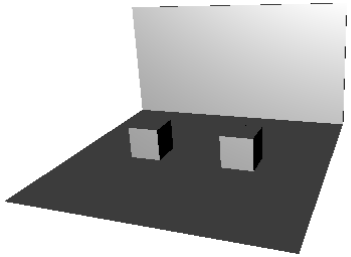
Fraunhofer IGD

Lichtquellen in X3DOM

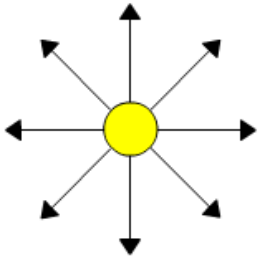
Sind Teil der <scene>

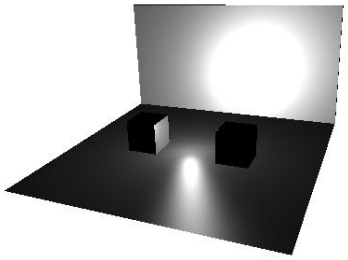
Direktionale Lichtquelle



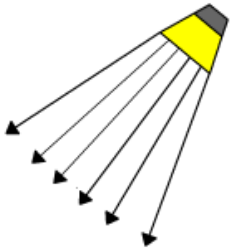


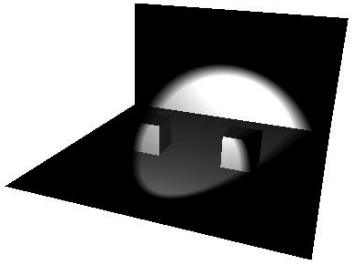
Punkt-Lichtquelle





Spot-Lichtquelle

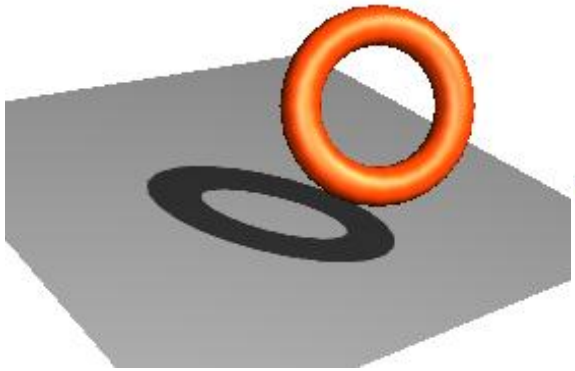




- <directionalLight direction='0 0 -1' intensity='1'> </directionalLight >
- <pointLight location='0 0 0' intensity='1'> </pointLight >
- <spotLight direction='0 0 -1' location='0 0 0' intensity='1'> </spotLight >

Weitere Rendering-Effekte

Schatten

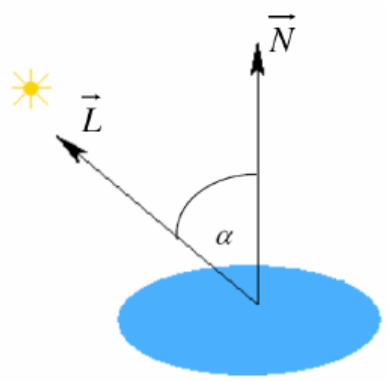


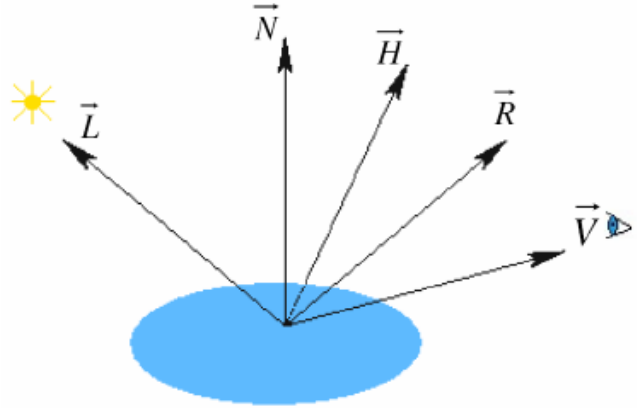
Nebel

Texturierung

- <directionalLight direction='0 0 -1' intensity='1' shadowIntensity='0.7'> </directionalLight >
  - Achtung: nur für das erste <directionalLight> in der Szene implementiert
- <fog visibilityRange= '1000'></fog>
- **<imageTexture** url="myTextureMap.jpg"**></ imageTexture>**
  - Achtung: wie <material> nur als Kindknoten von <appearance> möglich!

Exkurs: das Beleuchtungsmodell
(Physik: diffuse und spiegelnde Reflexion)





$\vec{L}$ = Lichtvektor
$\vec{N}$ = Normale
$\vec{H}$ = Halbvektor
(Winkel zw. $\vec{V}$ und $\vec{L}$)
$\vec{R}$ = Reflexionsvektor
$\vec{V}$ = Betrachtervektor (View)

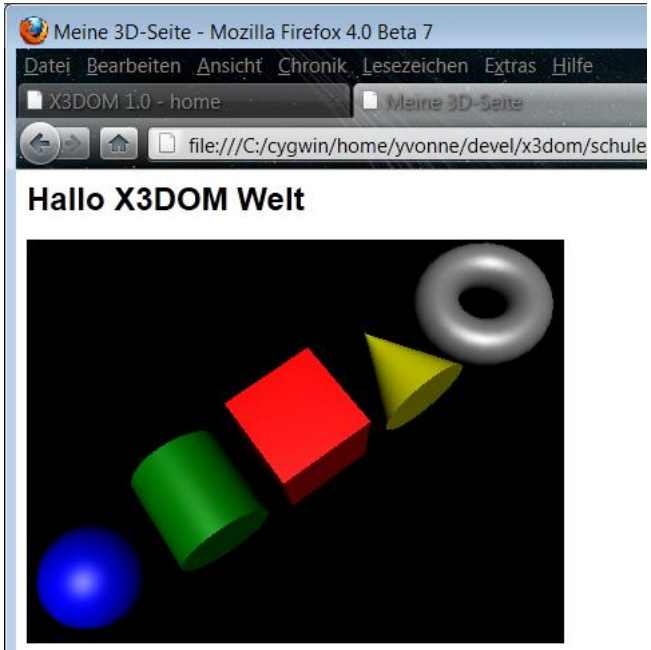
$$I_{diff} = \max(0, \vec{N} \bullet \vec{L}) = \max(0, \cos\alpha) \qquad I_{spec} = (\vec{N} \bullet \vec{H})^s = \cos^s \beta \qquad \vec{H} = (\vec{L} + \vec{V}) / |\vec{L} + \vec{V}|$$

Farbe *I* := ambientes Material + diffuses Material * *(N·L)* + spekulares Material * *(N·H)*

Bei mehreren Lichtquellen: $I_{ges} = a_{glob} \otimes m_{amb} + m_{em} + \sum_k c^k_{spot}(I^k_{amb} + d^k(I^k_{diff} + I^k_{spec}))$

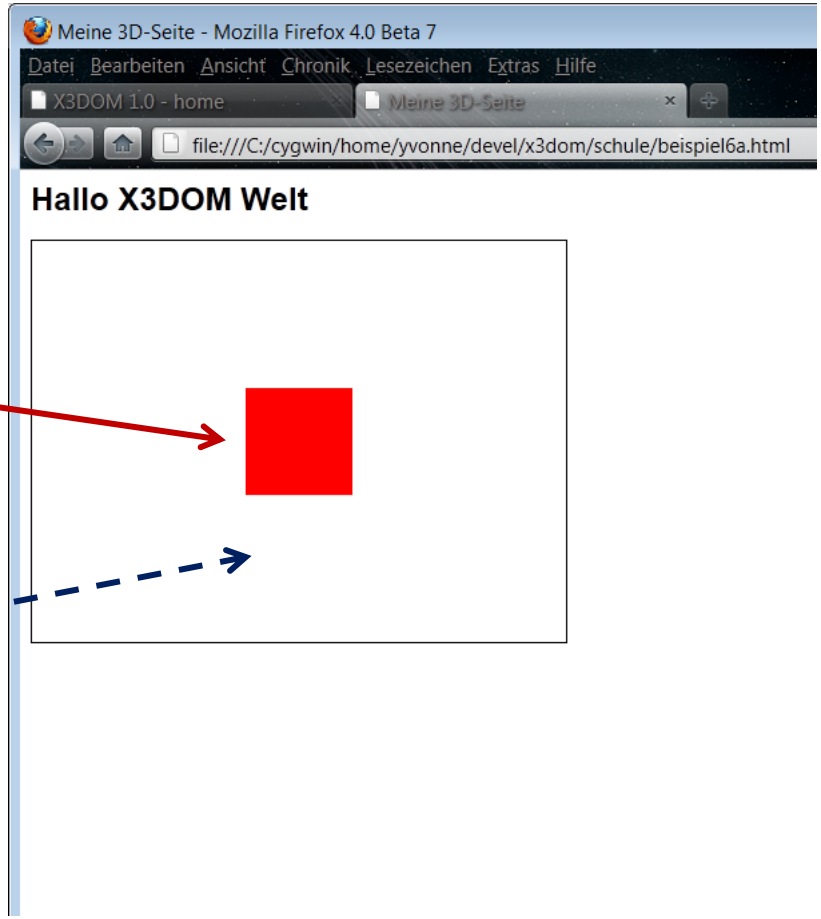
Geometrische Grundobjekte

- Siehe Screenshot – von links nach rechts:
- Kugel
  - <sphere radius="1.0">
- Zylinder
  - <cylinder radius="1.0" height="2.0">
- Würfel
  - <box size="2.0 2.0 2.0">
- Kegel
  - <cone bottomRadius="1.0" height="2.0">
- Ring
  - <torus innerRadius="0.5" outerRadius="1.0">



Zwei Objekte in einer Szene (?!)

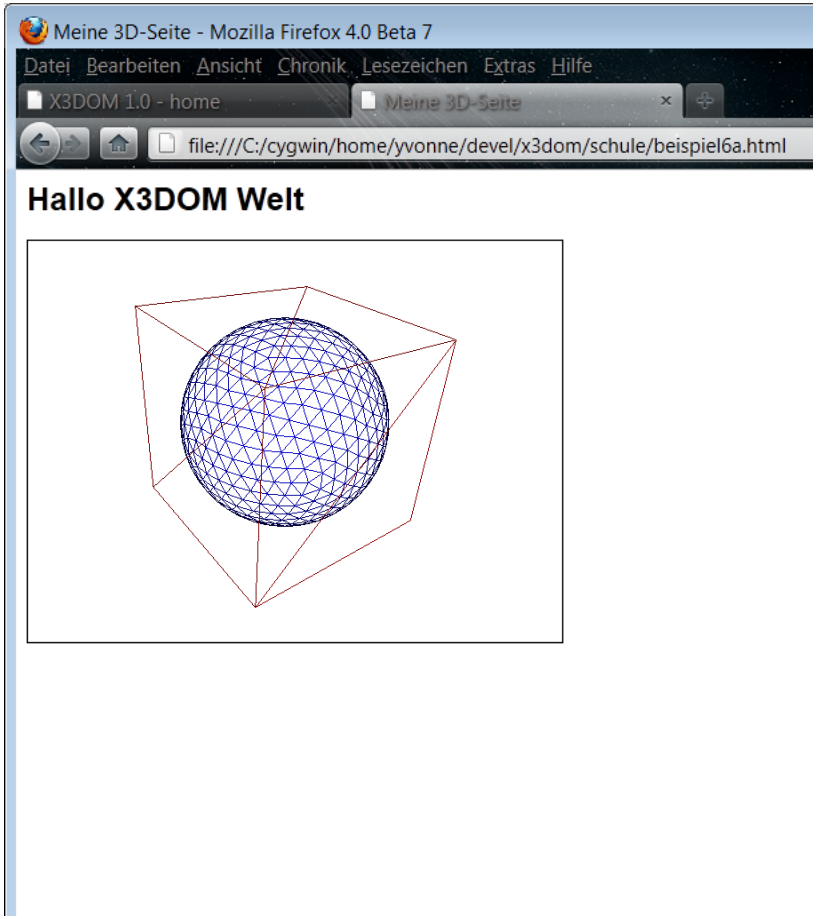
```
<scene>
    <shape>
      <appearance>
        <material diffuseColor='red'></material>
      </appearance>
      <box></box>
    </shape>
    <shape>
      <appearance>
        <material diffuseColor='blue'></material>
      </appearance>
      <sphere></sphere>
    </shape>
</scene>
…
```



Zwei Objekte in einer Szene

Problem: Beide erscheinen an der gleichen Position

```
<scene>
    <shape>
      <appearance></appearance>
      <box></box>
    </shape>
    <shape>
      <appearance></appearance>
      <sphere></sphere>
    </shape>
</scene>
```

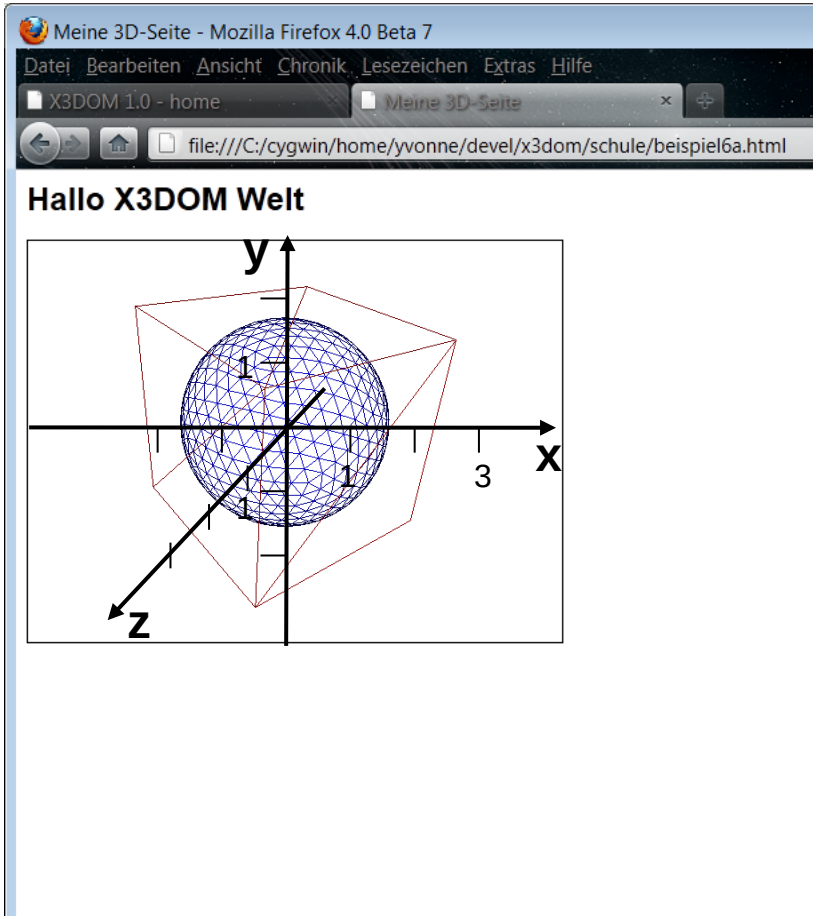


Fraunhofer IGD

Zwei Objekte in einer Szene

Problem: Beide erscheinen an der gleichen Position

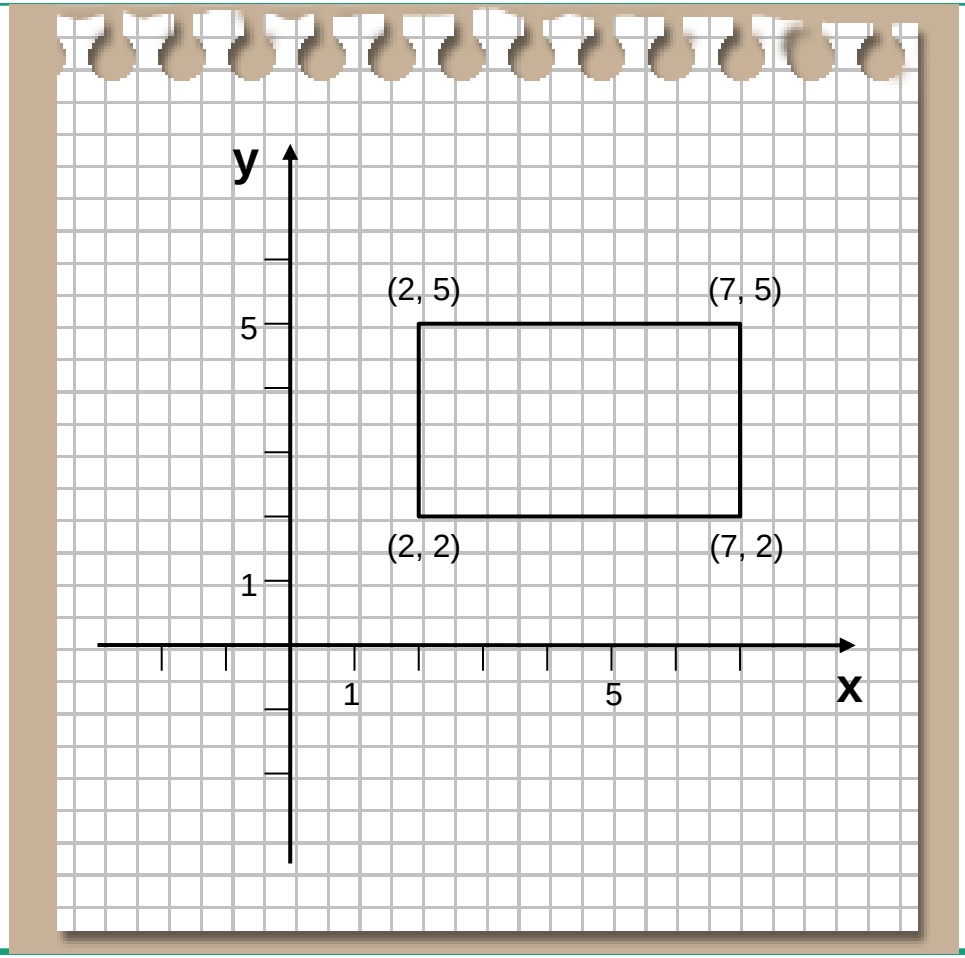
```
<scene>
    <shape>
      <appearance></appearance>
      <box></box>
    </shape>
    <shape>
      <appearance></appearance>
      <sphere></sphere>
    </shape>
</scene>
```



- **Ursache:** 3D-Objekte werden immer im Koordinatenursprung (Nullpunkt) erzeugt und müssen dann noch nachträglich positioniert werden

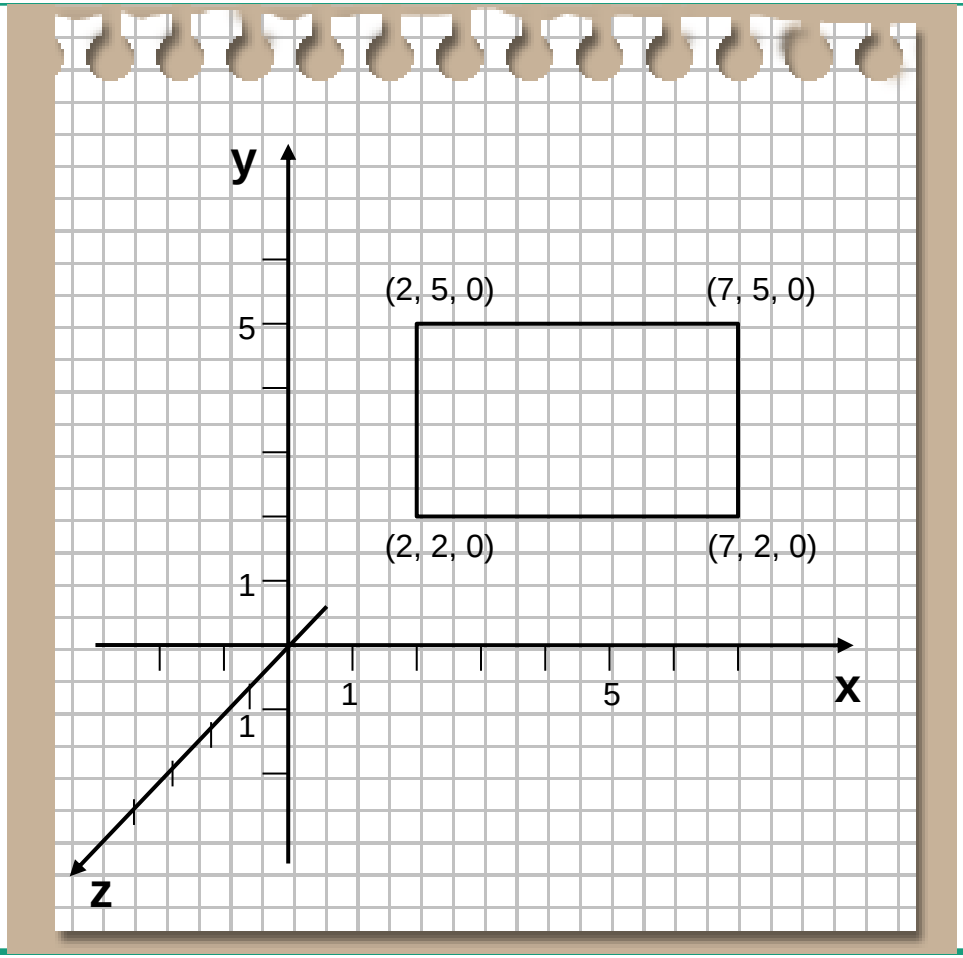
Exkurs: (2D-) Koordinatensysteme

Objektkoordinaten in der Bildebene (geg. durch x u. y)



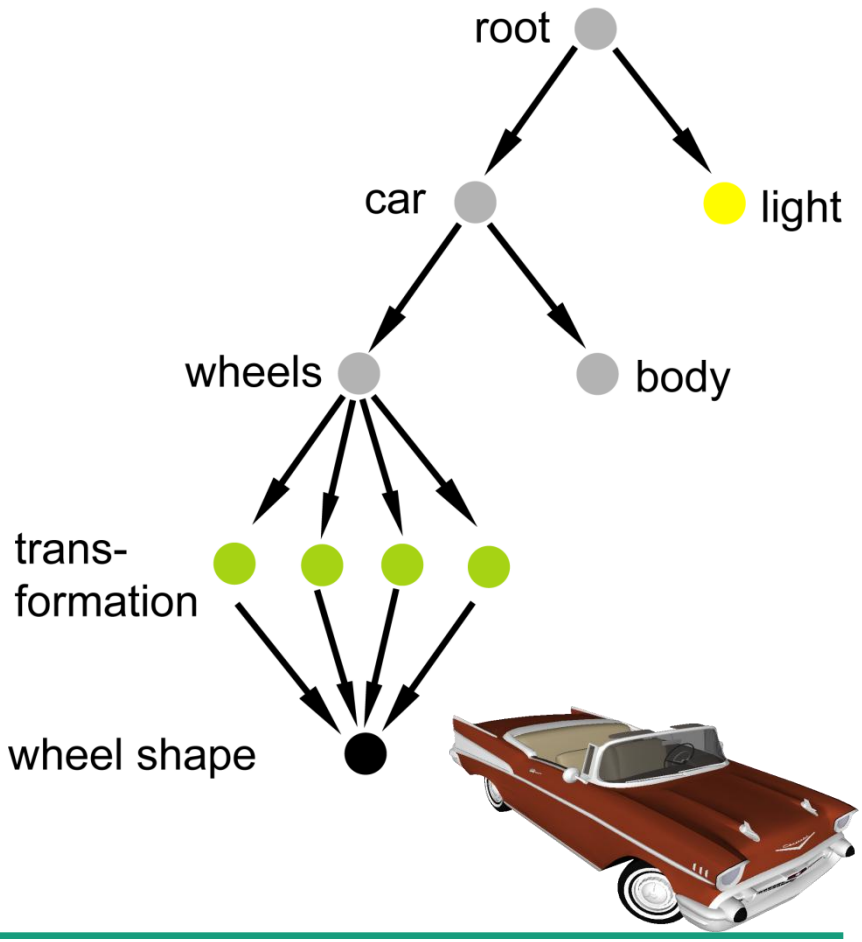
Exkurs: (3D-) Koordinatensysteme

Objektkoordinaten im Raum (z senkrecht auf x und y)



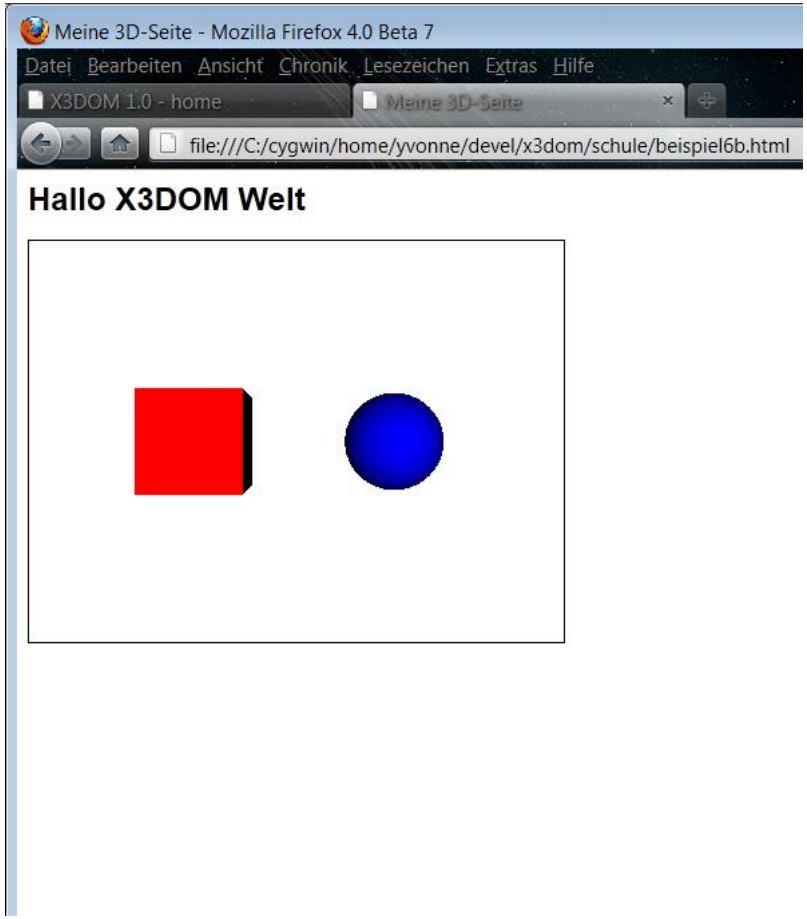
Der Szenengraph
Gruppierung und Transformationen

- 3D-Elemente werden hierarchisch angeordnet
  - Ausgehend vom Wurzelknoten (d.h. vom <scene> Element) werden alle 3D-Elemente (z.B. <shape>, <box> usw.) als Kind- bzw. Geschwisterelemente in den “Baum” (Szenegraphen) eingefügt
  - <transform> Elemente helfen dabei, Objekte zu gruppieren sowie zu positionieren:
    - <transform translation="0 0 0" rotation="0 1 0 0" scale="1 1 1">
      ...
      </transform>



Zwei Objekte in einer Szene
Diesmal mit Verschiebungen (d.h. translation)

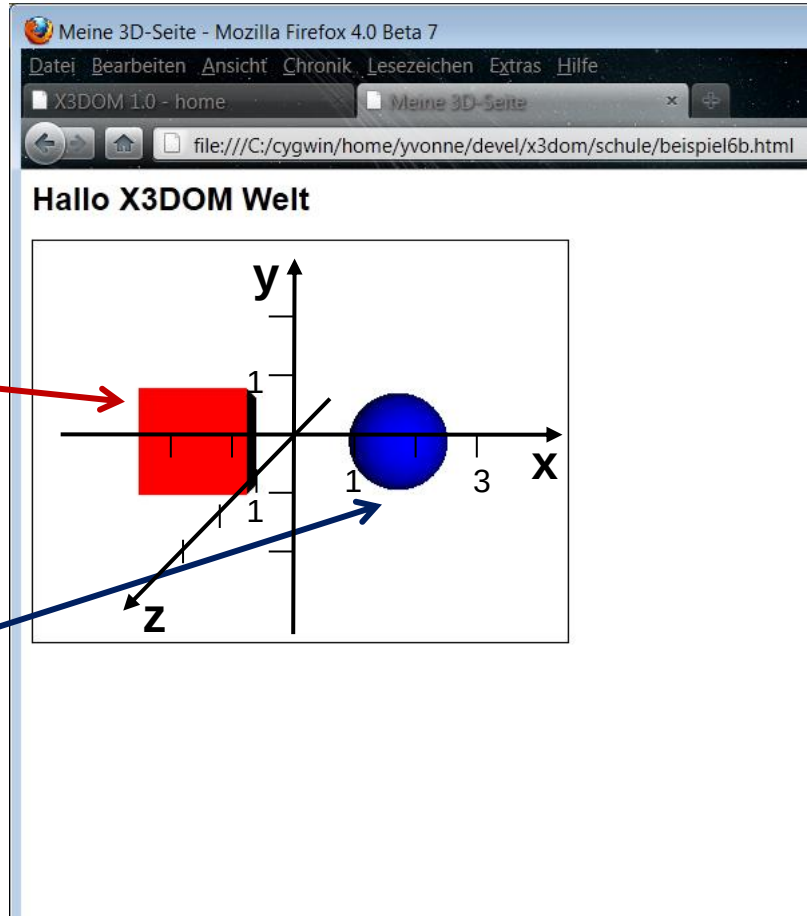
```
<transform translation="-2 0 0">
  <shape>
    <appearance>
      <material diffuseColor="red"></material>
    </appearance>
    <box></box>
  </shape>
</transform>
<transform translation="2 0 0">
  <shape>
    <appearance>
      <material diffuseColor="blue"></material>
    </appearance>
    <sphere></sphere>
  </shape>
</transform>
```



Zwei Objekte in einer Szene

Diesmal mit Verschiebungen (d.h. translation)

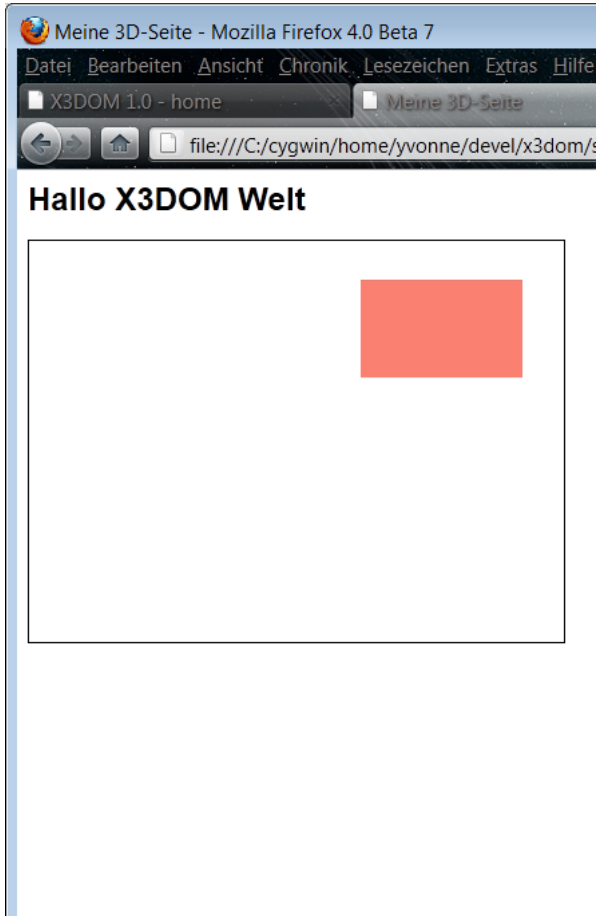
```
<transform translation="-2 0 0">
  <shape>
    <appearance>
      <material diffuseColor="red"></material>
    </appearance>
    <box></box>
  </shape>
</transform>
<transform translation="2 0 0">
  <shape>
    <appearance>
      <material diffuseColor="blue"></material>
    </appearance>
    <sphere></sphere>
  </shape>
</transform>
```



Eigene Geometrien gestalten

Bsp.: ein einfaches Rechteck mit <indexedFaceSet>

```
<scene>
    <shape>
        <appearance>
            <material diffuseColor="salmon">
            </material>
        </appearance>
        <indexedFaceSet coordIndex="0 1 2 3 -1">
            <coordinate point="2 2 0, 7 2 0, 7 5 0, 2 5 0">
            </coordinate>
        </indexedFaceSet>
    </shape>
    <viewpoint position="0 0 15"></viewpoint>
</scene>
```



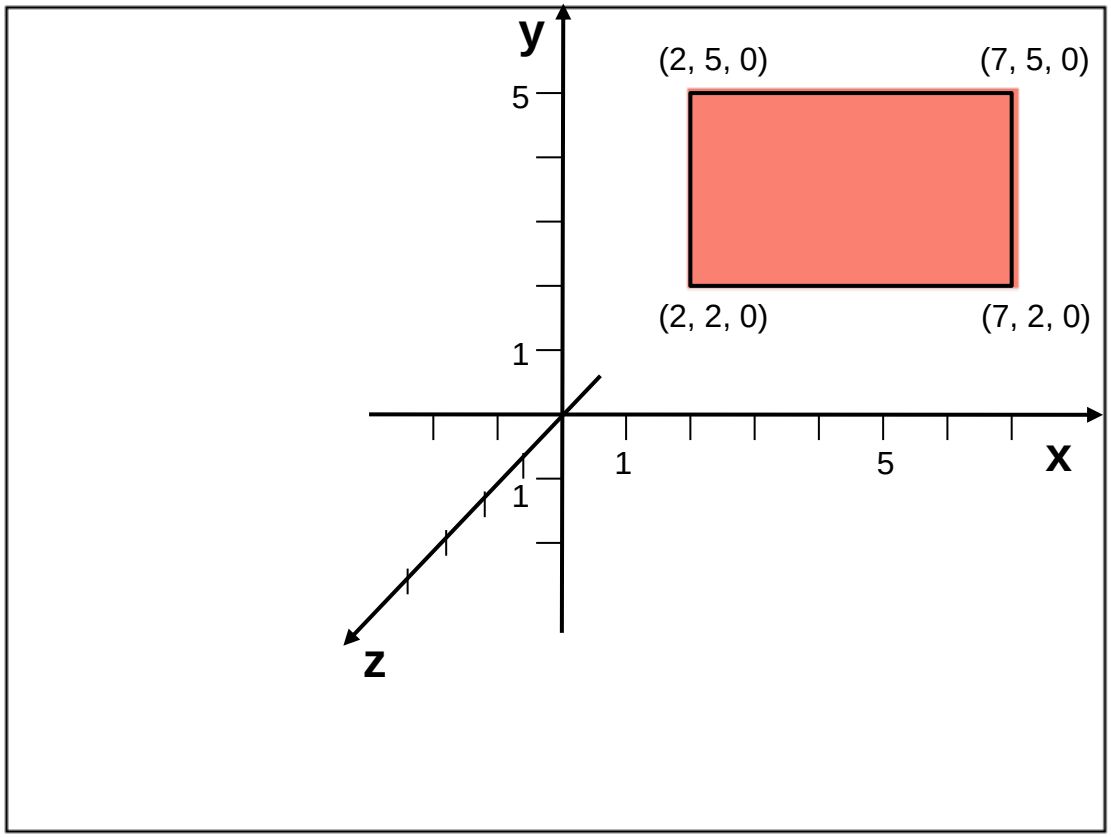
Fraunhofer IGD

Eigene Geometrien gestalten

Bsp.: ein einfaches Rechteck mit <indexedFaceSet>

```
<indexedFaceSet
  coordIndex="0 1 2 3 -1">
   <coordinate point=
  "2 2 0, 7 2 0, 7 5 0, 2 5 0">
  </coordinate>
</indexedFaceSet>
```

- Wichtigste Bausteine
  - Die Eckpunkte eines Polygons (hier “face”), geg. als <coordinate>
  - Der Index auf einen solchen Eckpunkt, geg. als Liste: “coordIndex”

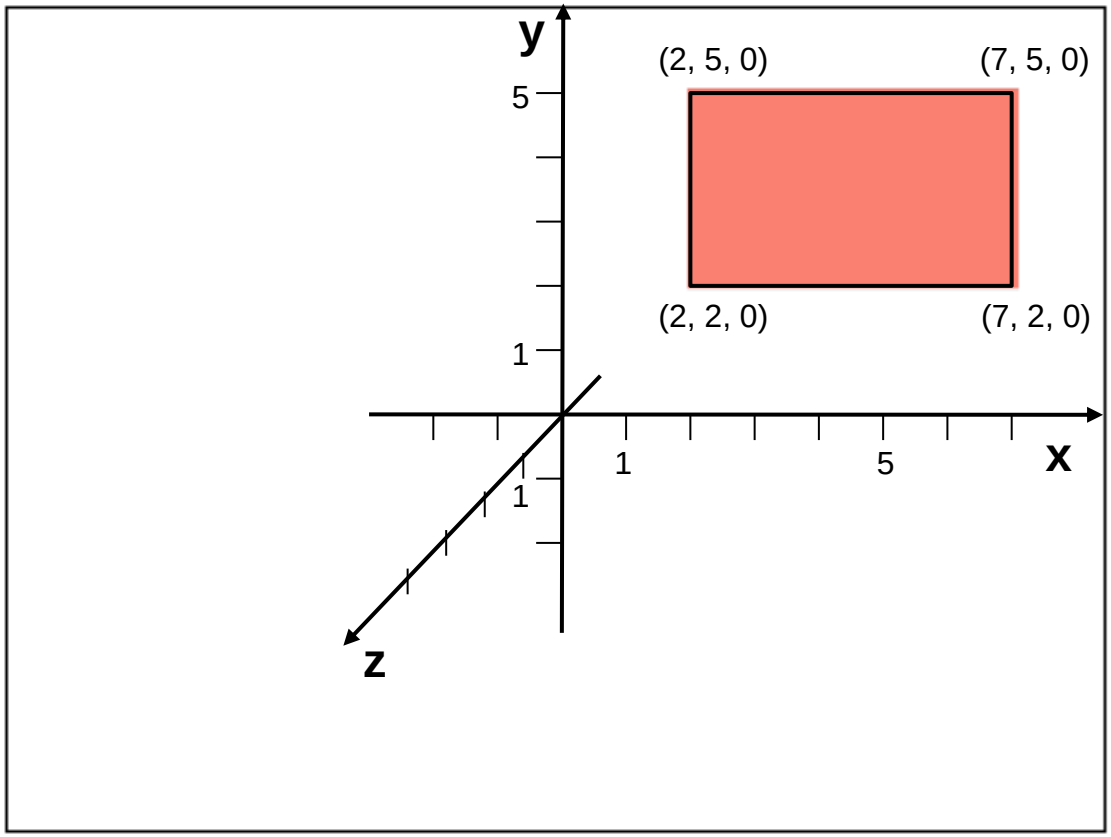


Eigene Geometrien gestalten

Bsp.: ein einfaches Rechteck mit <indexedFaceSet>

```
<indexedFaceSet
  coordIndex="0 1 2 3 -1">
   <coordinate point=
  "2 2 0, 7 2 0, 7 5 0, 2 5 0">
  </coordinate>
</indexedFaceSet>
```

- Das Ende eines Polygons und der Beginn eines neuen wird durch eine “-1” in der Indexliste markiert
- Auf diese Weise lassen sich beliebig komplexe 3D-Objekte erzeugen

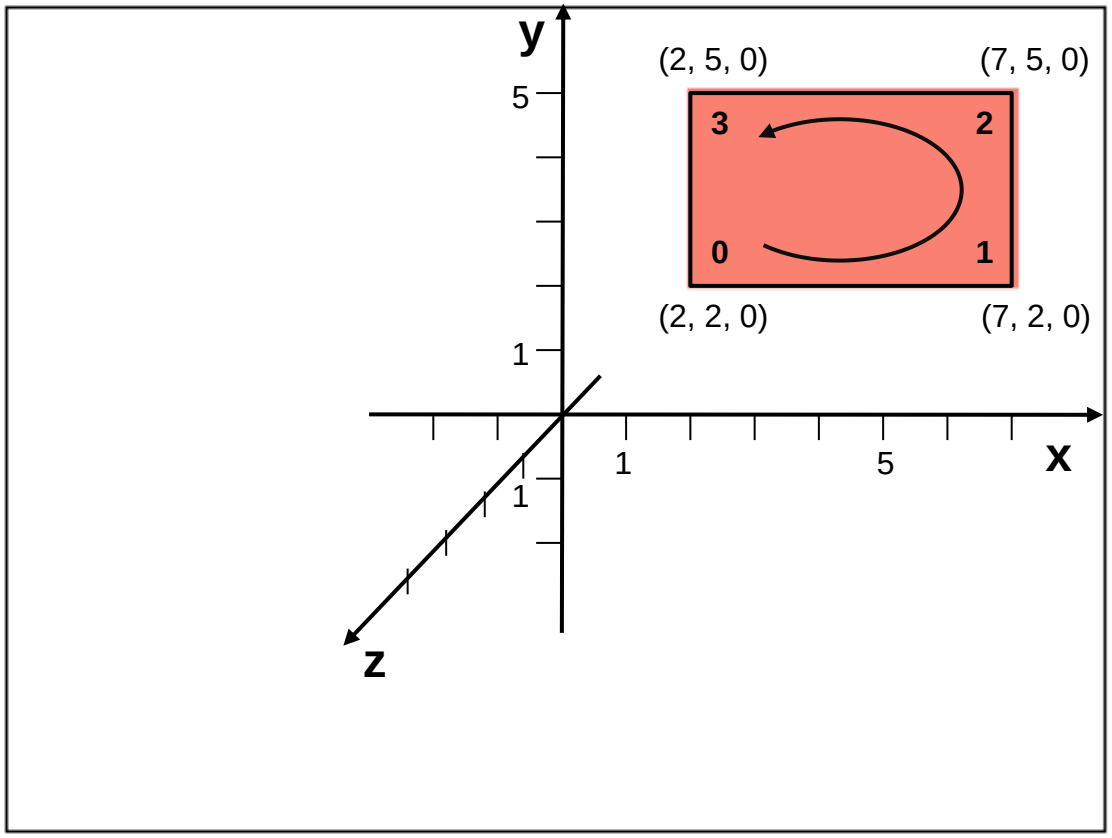


Eigene Geometrien gestalten

Bsp.: ein einfaches Rechteck mit <indexedFaceSet>

```
<indexedFaceSet
  coordIndex="0 1 2 3 -1">
   <coordinate point=
  "2 2 0, 7 2 0, 7 5 0, 2 5 0">
  </coordinate>
</indexedFaceSet>
```

- Die Indizes (außer “-1”) zeigen dabei je auf die Listenposition einer 3D-Koordinate in <coordinate>
- Die Koordinaten eines Polygons werden dabei je im **Gegenuhrzeigersinn** angegeben



Für Fortgeschrittene: Animationen
TimeSensor und X3D-Interpolator-Knoten

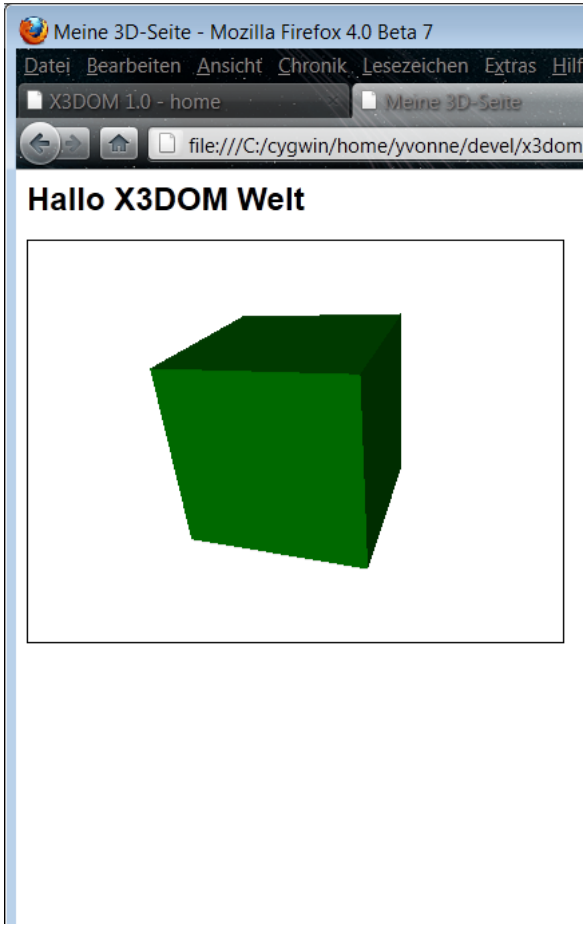
```
<scene>
 <transform id="trafo" rotation="0 1 0 0">
  <shape>
   <appearance>
    <material diffuseColor="red">
    </material>
   </appearance>
   <box></box>
  </shape>
 </transform>
 <timeSensor id="ts" loop="true" cycleInterval="2">
 </timeSensor>
 <orientationInterpolator id="oi" key="0.0 0.5 1.0"
   keyValue="0 1 0 0, 0 1 0 3.14, 0 1 0 6.28">
 </orientationInterpolator>
 <ROUTE fromNode='ts' fromField='fraction_changed'
   toNode='oi' toField='set_fraction'></ROUTE>
 <ROUTE fromNode='oi' fromField='value_changed'
   toNode='trafo' toField='set_rotation'></ROUTE>
</scene>
```

- Der <timeSensor> namens „ts“ befeuert mit Hilfe der ersten ROUTE den <orientationInterpolator> „oi“, der für die Drehung um die y-Achse (0,1,0) sorgt
- Das Ergebnis (value) wird über die nächste ROUTE an das Feld ‘rotation’ des <transform>-Knotens “trafo” weitergereicht, was die Animation bewirkt

Für Fortgeschrittene: Attribute lassen sich über DOM-Events(*) interaktiv verändern

```
<shape>
   <appearance>
      <material id="mat" diffuseColor="red">
      </material>
   </appearance>
   <box onclick="
         document.getElementById('mat').
         setAttribute('diffuseColor', 'green');" >
   </box>
</shape>
...
```

(*) Gute Tutorials und Beispiele zu HTML, CSS, JavaScript und DOM-Scripting gibt es auf SELFHTML: http://de.selfhtml.org/

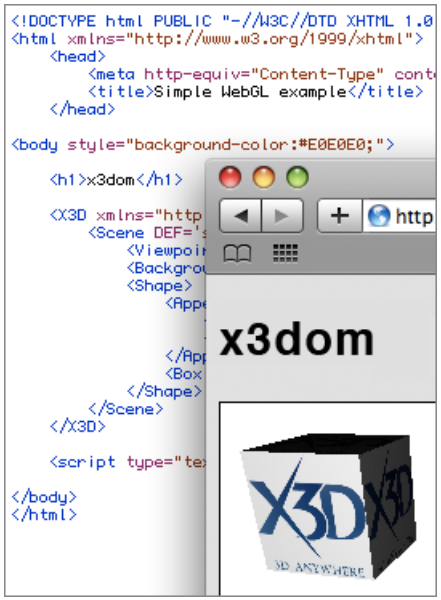


Exkurs: Zum Weiterlesen…

- Mehr Informationen und Beispiele zu X3DOM gibt es direkt auf www.x3dom.org
  - Beispiele (auf Englisch): http://www.x3dom.org/?page_id=5
  - Noch mehr Beispiele/Tests: http://www.x3dom.org/x3dom/src/
  - Tutorials (auf Englisch): http://www.x3dom.org/?page_id=482
  - Beispiele (auf Deutsch): http://www.x3dom.org/iX/
- Veröffentlichungen zu X3DOM
  - Behr, Johannes; Eschler, Peter; Jung, Yvonne; Zöllner, Michael: X3DOM – A DOM-based HTML5 / X3D Integration Model. In: Proceedings Web3D 2009, S. 127-135
  - Behr, Johannes; Jung, Yvonne; Keil, Jens; Drevensek, Timm; Zöllner, Michael; Eschler, Peter; Fellner, Dieter: A Scalable Architecture for the HTML5 / X3D Integration Model X3DOM. In: Proceedings Web3D 2010, S. 185-193
  - Slusallek, Philipp; Behr, Johannes; Jung, Yvonne; Sons, Kristian: Erdverbunden – X3DOM & XML3D: Transformationen und Interaktion. In: iX 12 (2010), S. 116-121
  - Slusallek, Philipp; Behr, Johannes; Jung, Yvonne; Sons, Kristian: Browser(t)räume – X3DOM & XML3D: Deklaratives 3D in HTML5. In: iX 11 (2010), S. 54-63

Deklarative 3D-Grafik im Webbrowser

http://www.x3dom.org/



Noch Fragen?