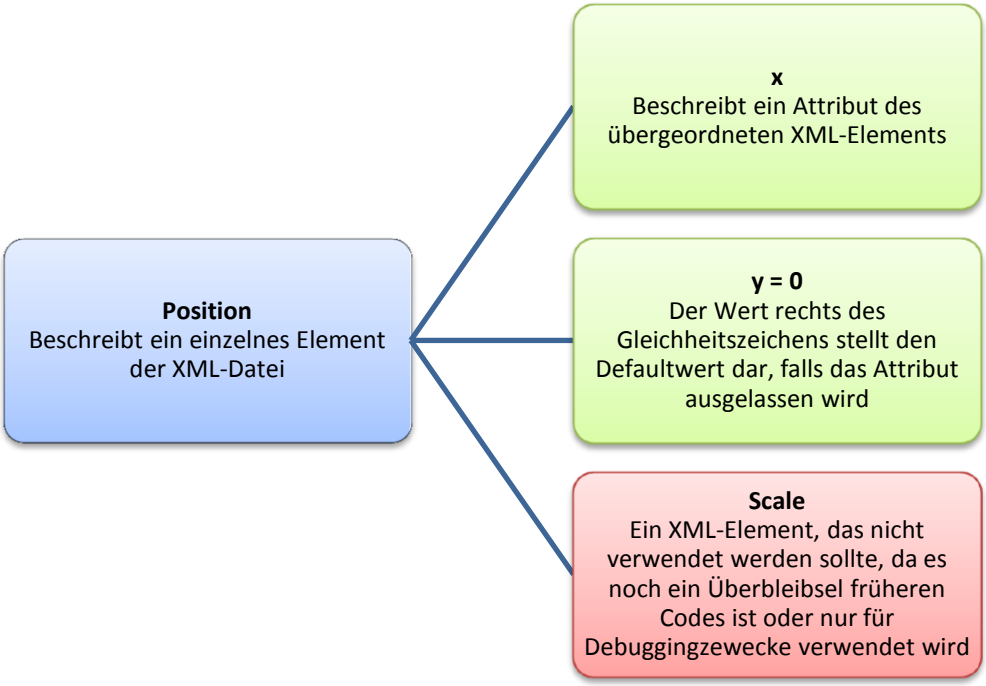
Aufbau interaktiver 3D-Engines

ActorComponents

Im Folgenden findet sich eine Übersicht über alle bereits implementierten Actorcomponents der Engine. Dabei werden XML-Elemente, die die Komponente beschreiben auf folgende Weise farblich dargestellt.



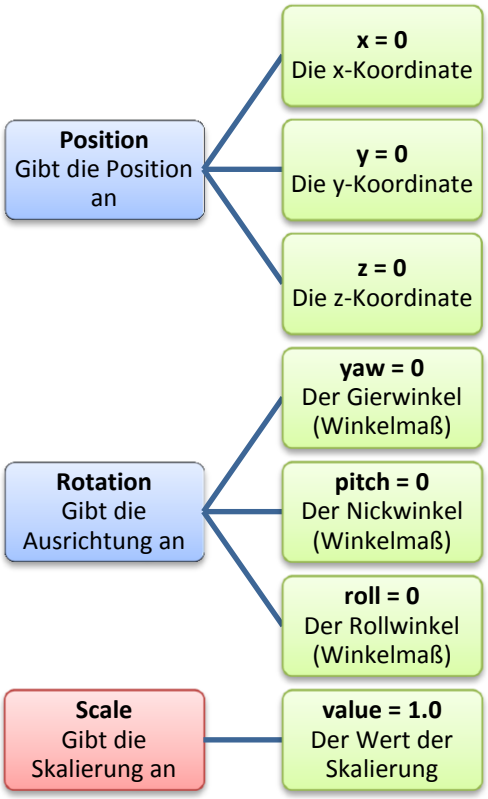
TransformComponent

Ein Actor mit einer TransformComponent besitzt eine eindeutige Position und Ausrichtung im Raum.

Wichtige Methoden

- `Pose getPose()`
  - Liefert ein Objekt, das die Position und Ausrichtung kapselt. Das Objekt kann direkt manipuliert werden, wodurch ActorMovedEvents erzeugt werden
  - Darf nur manipuliert werden, falls der Actor kein RigidBody durch eine PhysicsComponent darstellt

XML-Unterelemente



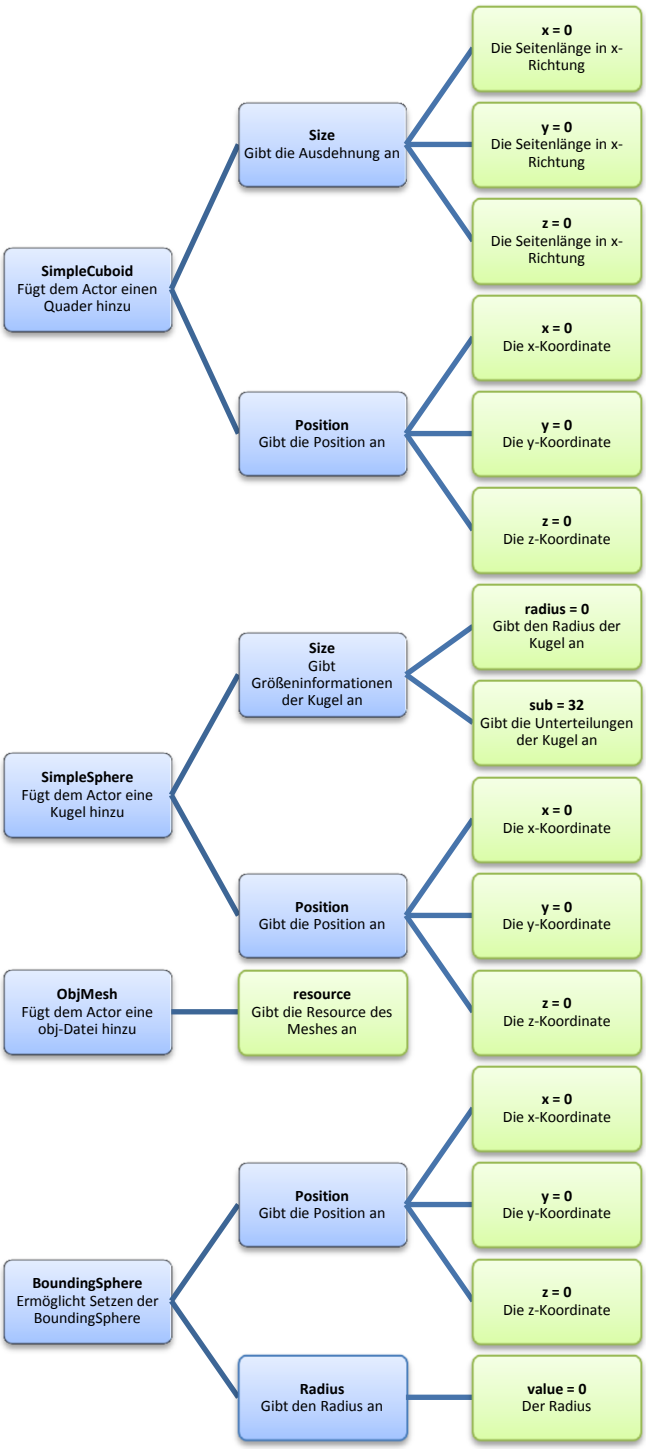
RenderComponent

Ein Actor mit einer RenderComponent kann mittels mehrerer Meshes graphisch dargestellt werden.

Wichtige Methoden

- `void drawGeometries()`
  - Bindet einen evtl. vorhandenen eigenen Effekt und zeichnet alle Meshes der Komponente
- `BoundingSphere getBoundingSphere()`
  - Liefert eine Kugel, die alle Meshes enthält

XML-Unterelemente



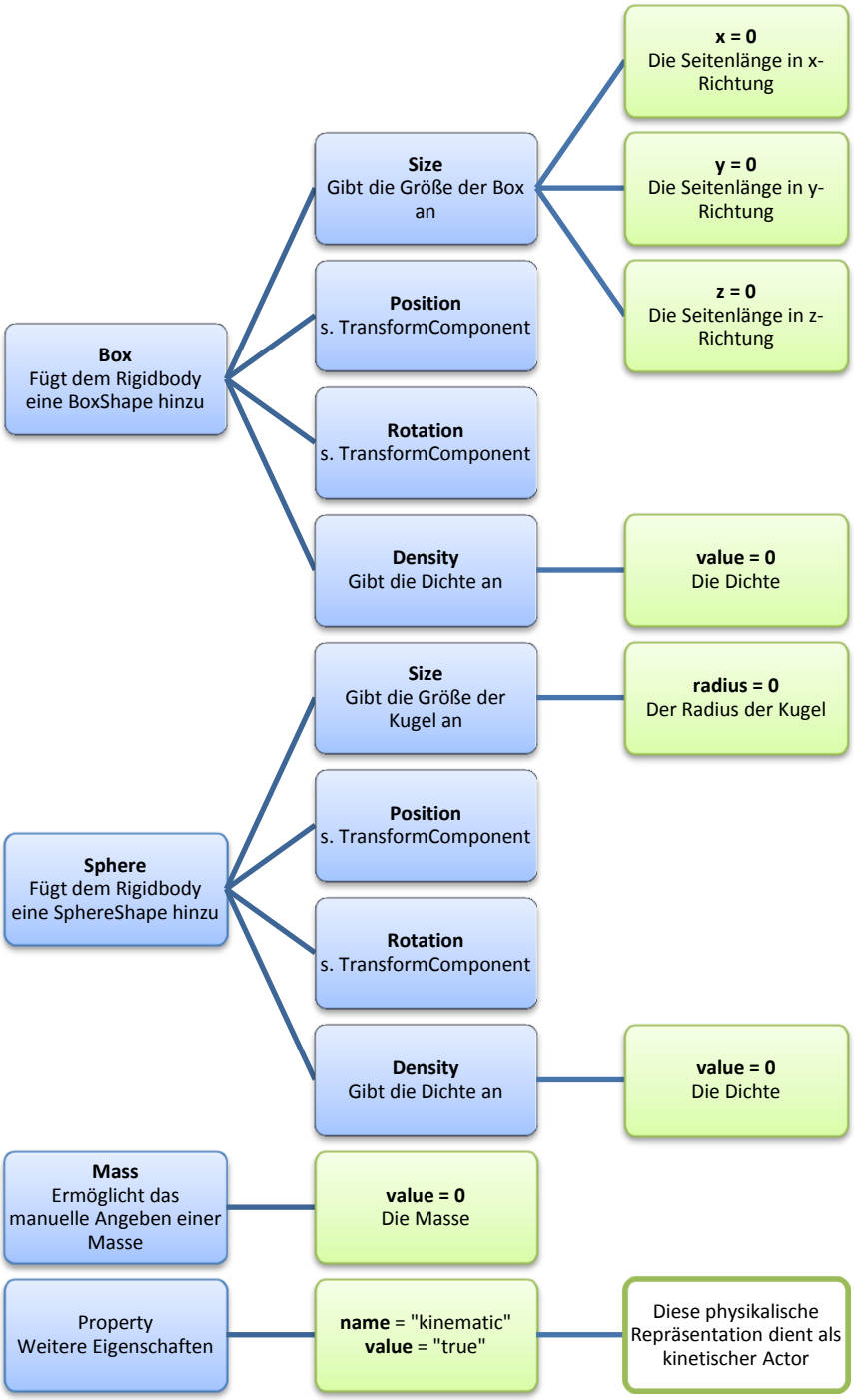
PhysicsComponent

Verbindet einen Rigidbody mit dem Actor. In der Komponente können eine Reihe von Shapes angegeben werden, die die physikalische Repräsentation angeben. Jedem Shape kann eine Dichte zugewiesen werden. Haben alle Shapes die Dichte 0, so ist der Rigidbody statisch und wird dementsprechend von Kräften und Kollisionen nicht beeinflusst. Er selbst beeinflusst andere dynamische Rigidbodies jedoch weiterhin. Ein Actor mit dieser Komponente benötigt zwingend eine TransformComponent.

Wichtige Methoden

Keine

XML-Unterelemente



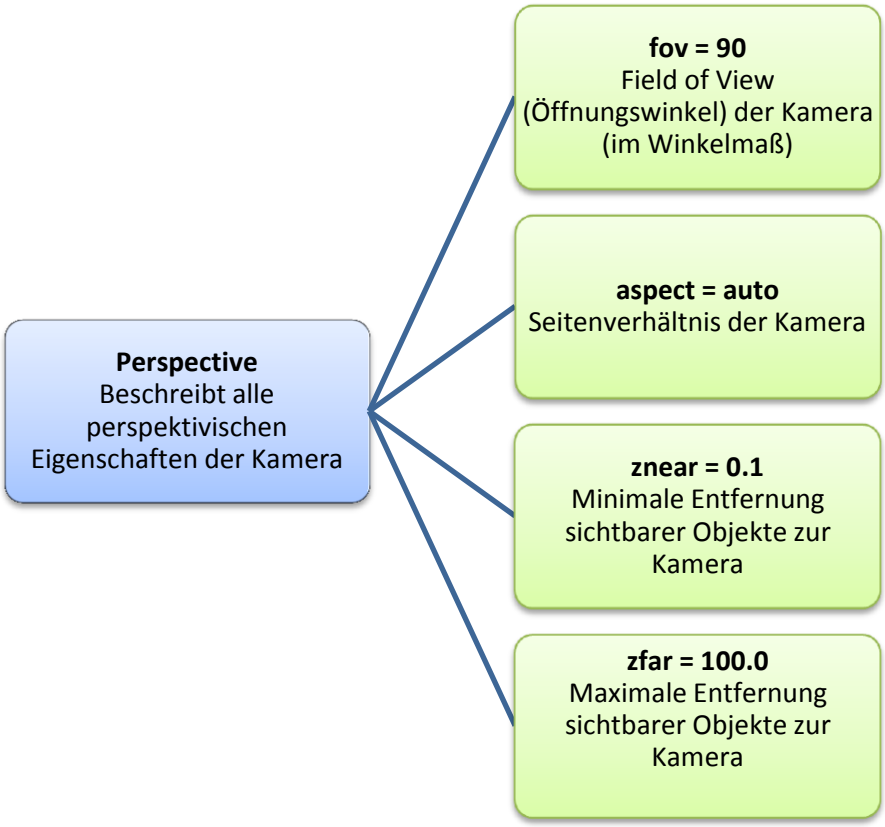
CameraComponent

Ein Actor mit dieser Komponente bietet die Möglichkeit, die Szene aus seiner Sicht zu rendern. Ein Actor mit dieser Komponente benötigt zwingend eine TransformComponent. Die Pose der TransformComponent wird bei einer Kamera immer global interpretiert. Daher sollte dieser Actor keinen Vorfahren mit TransformComponent besitzen.

Wichtige Methoden

- `PerspectiveCamera getPerspectiveCamera()`
    - Liefert das Kameraobjekt

XML-Unterelemente



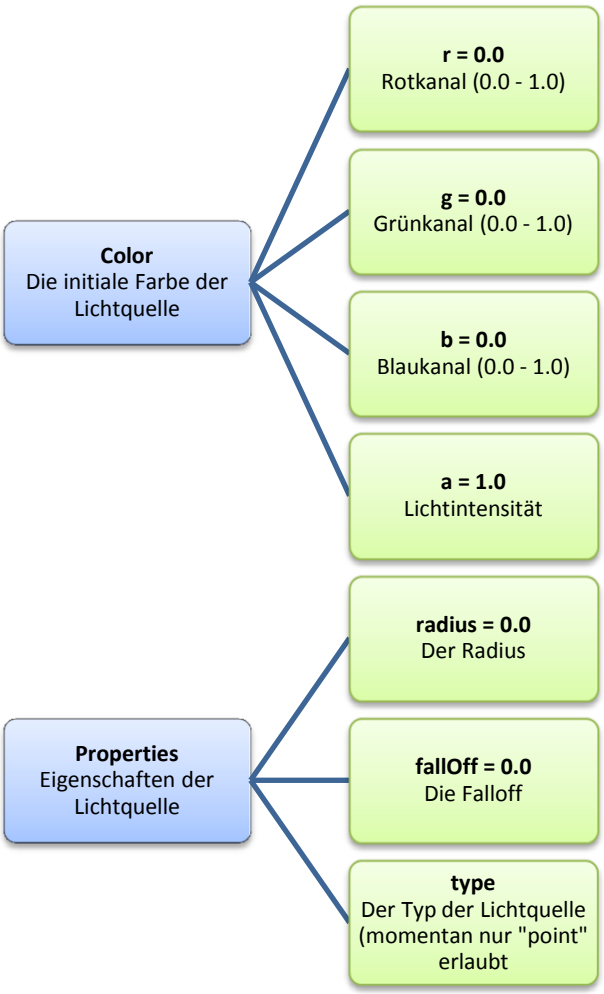
LightComponent

Lichter können mithilfe dieser Komponente platziert werden. In der Engine werden die acht Lichtquellen, die der aktiven Kamera am nächsten sind, angezeigt. Um diesen Wert zu erhöhen, muss der `#define MAX_LIGHTS` des Shaders `LightingFS.glsl` angepasst werden, sowie die statische Konstante *`MAX_LIGHTS`* der Klasse `LightAdmin`.

Wichtige Methoden

- `LightType getType()`
  - Liefert den Typ der Lichtquelle
- `Vector4f getColor()`
  - Liefert die Farbe der Lichtquelle
- `float getRadius()`
  - Liefert den Radius der Lichtquelle
- `void setRadius(float radius)`
  - Setzt einen neuen Radius der Lichtquelle (experimental)
- `float getFallOff()`
  - Liefert den Falloff der Lichtquelle. Dieser Wert beeinflusst, wie stark die Lichtintensität mit zunehmender Entfernung abnimmt. Je höher desto stärker.
- `void setFallOff(float fallOff)`
  - Setzt einen neuen Falloff der Lichtquelle

XML-Unterelemente



AudioComponent

Ermöglicht es, Sounds von der Position des Actors aus abzuspielen. Die Sounddateien müssen dabei im wav-Format vorliegen.

Wichtige Methoden

- `AudioBuffer getSample(String name)`
    - Liefert ein Soundobjekt, das abgespielt oder gestoppt werden kann

XML-Unterelemente