



## Einführung „VPython auf Deutsch“

"VPython auf Deutsch" ist eine Erweiterung zu VPython (siehe <http://vpython.org>).

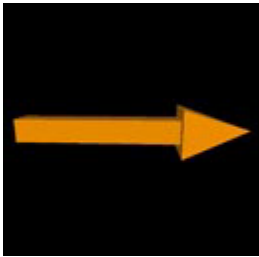
Es wurden hierfür die Versionen VPython 6.11 python 2.7.9 verwendet. Sämtliche Möglichkeiten, die VPython bzw. Python bieten, sind auch in VPython auf Deutsch enthalten. Mit Hilfe des neuen Moduls "DePy" geben sich folgende weitere Varianten:

- Programmieren mit deutschen Bezeichnungen
- Erlernen von objektorientierten Konzepten mit Hilfe einer passenden didaktischen Oberfläche.
- Werden nur die deutschen Bezeichnungen, ohne die Oberfläche gewollt, so wird nur das Modul "DePyTranslation" benötigt.

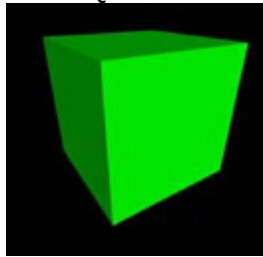
### Deutsche VPython-Klassen und Bezeichnungen

Folgende Klassen können mit ihren deutschen Begriffen verwendet werden:

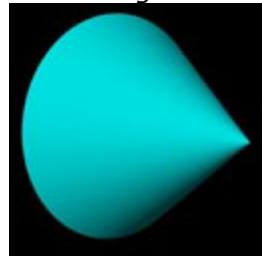
*Pfeil*



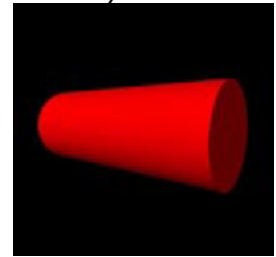
*Quader*



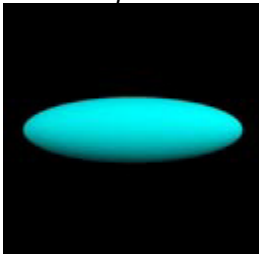
*Kegel*



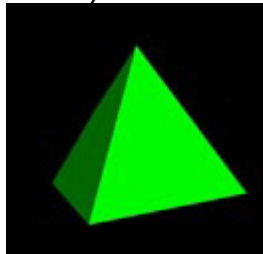
*Zylinder*



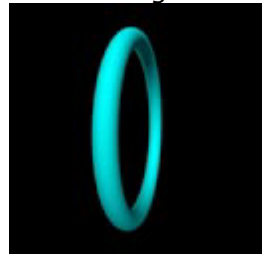
*Ellipsoid*



*Pyramide*



*Ring*



*Kugel*



### deutsche Bezeichnungen der Materialien:

Materialien.Ziegel

Materialien.Holz

Materialien.Erde

Materialien.Wolkenerde

Materialien.Marmor

Materialien.Glanz

Materialien.Metallic

Materialien.Granit

Materialien.Chrom

Materialien.Leuchtend

Materialien.Plastik

Materialien.Glitzer

Materialien.Schattenlos

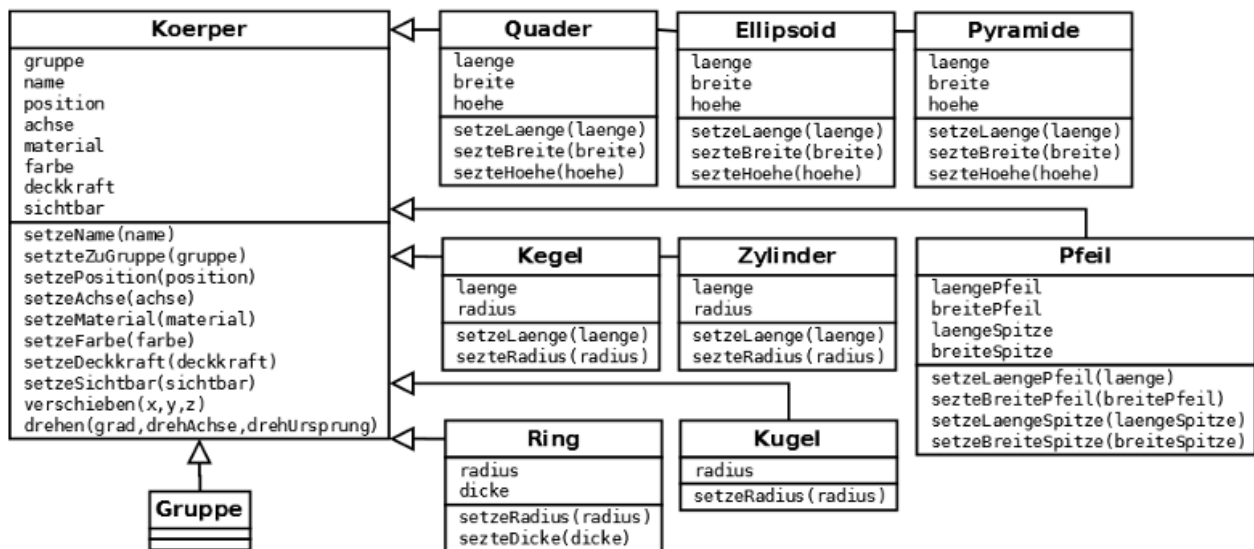




## deutsche Bezeichnungen der Farben:

Farben.rot	Farben.gelb	Farben.schwarz	Farben.gruen
Farben.orange	Farben.blau	Farben.cyan	Farben.magenta

## Klassendiagramm der deutschen VPython-Klassen (ohne Fenster):



## deutsche Bezeichnungen der Fensterattribute:

Fenster
zentrum
autoZoom
benutzerZoom
vordergrundfarbe
hintergrundfarbe
dreiDBrille
autoZentrum
kameraRichtung
kameraWinkel
region
zoom
auf
titel
breite
hoehe
vollbild
setzeZentrum(zentrum)
setze...(...)

Um die Bezeichnungen der Fensterattribute kennenzulernen wird im Nachfolgenden eine Tabelle angezeigt, die sowohl den Attributnamen, als auch seine originale (englische) Bezeichnung enthält. Die genauere Bedeutung des Attributs kann dann in der Dokumentation zu VPython ([vpython.org](http://vpython.org)) nachgelesen werden.

Weiter wird unterschieden, ob die Attribute auch in der GUI Sinn machen oder nur für das deutsche VPython gedacht sind (also nur Einbindung des Moduls `DePyTranslation`). Dies ist z. B. beim Attribut `titel` der Fall, innerhalb der GUI kann kein Titel des 3D-Grafik-Fensters gesetzt werden.

Alle Attribute besitzen zudem die Methode `setzeATTRIBUTNAME`, z. B. für das Attribut `zentrum` ist dies `setzeZentrum(zentrum)`.

Attribute, die auch innerhalb der GUI Sinn machen:

Attributname	urprünglicher (englischer Name)
zentrum	center
autoZoom	autoscale
benutzerZoom	userzoom





benutzerRotation	userspin
vordergrundfarbe	foreground
hintergrundfarbe	background
dreidBrille	stereo
autoZentrum	autocenter
kameraRichtung	forward
kameraWinkel	fov
region	range
zoom	scale
auf	up

Attribute, die mit der GUI **keinen** Sinn machen:

Attributname	urprünglicher (englischer Name)
titel	title
breite	width
hoehe	height
vollbild	fullscreen

Sollen die deutschen (Fenster-)bezeichnungen, ohne die didaktische Oberfläche verwendet werden, so muss dem Programm einmal mitgeteilt werden, dass das Fenster-Objekt, das die programmierten Objekte enthält nun "fenster" heißt und von der (deutschen) Klasse Fenster ist. Dazu wird folgender Aufbau benötigt (siehe DePyDeutsch.py)

```
# Umlaute und Programm in deutscher Sprache laden:
# -*- coding: utf-8 -*-
from DePyTranslation import *
```

# Belasse folgenden beide Zeilen immer am Anfang deines Programms, um auch auf  
# deutsche Fensterattribute zugreifen zu können

```
fenster = Fenster()
scene = fenster
```

# Hier beginnt nun dein eigentliches Programm

*An dieser Stelle steht der vom Schüler implementierte Code.*

Ein komplettes Beispiel beinhaltet die Datei DePyDeutschBeispiel.py.





## Attributwertzuweisungen:

Attributwerte können mit Hilfe der Punktnotation entweder direkt oder mit Hilfe von Methoden zugewiesen werden.

### Beispiel:

Ein Objekt namens `Planet` wird erstellt, die Attribute `name`, `position` und `material` sollen Werte erhalten.

#### Möglichkeit 1:

```
planet = Kugel()
planet.name = "Erde"
planet.position = (2,1,5)
planet.material = Materialien.Erde
```

#### Möglichkeit 2:

```
planet = Kugel()
planet.setzeName("Erde")
planet.setzePosition(2,1,5)
planet.setzeMaterial(Materialen.Erde)
```

## Gruppierung von Objekten:

Um eine Aggregation zu erstellen (mehrere Objekte zu gruppieren) gibt es entweder die Möglichkeit, die Objekte beim Erstellen der Gruppe dieser zuzuweisen, oder aber auch durch Attributwertzuweisung bzw. durch Methodenaufruf.

### Beispiel:

Eine neue Gruppe namens `Rakete` soll erstellt werden. Diese Gruppe soll die (schon vorher erstellten) Objekte `rumpf` und `spitze` enthalten (der erste Parameter ist jeweils der Wert des Gruppenattributs `name`, in diesem Beispiel wäre der `name` der Gruppe also `'Rakete'`).

#### Möglichkeit 1:

```
rakete = Gruppe('Rakete', rumpf, spitze)
```

#### Möglichkeit 2:

```
rakete = Gruppe('Rakete')
rumpf.gruppe = rakete
spitze.gruppe = rakete
```

#### Möglichkeit 3:

```
rakete = Gruppe('Rakete')
rumpf.setzezuGruppe(rakete)
spitze.setzezuGruppe(rakete)
```





## didaktische Oberfläche

Um die didaktische Oberfläche aufzurufen, wird folgender Aufbau benötigt (siehe Datei *DePyAufbau.py*):

```
# Umlaute und Programm in deutscher Sprache laden:
# -*- coding: utf-8 -*-
from DePy import *

# Ab hier kannst Du Objekte erstellen und deren Attribute
# verändern, sowie Objekte zu Gruppen zusammenfügen und diese
# animieren:

In diesem Bereich steht der eigene Code

# Belasse folgende Zeile immer am Ende deines Programms:
objekteAnzeigen()
```

### Beispiel:

Als Einführungsbeispiel dient die Datei *DePyBeispielRakete.py*. Eine Rakete besteht aus einem Zylinder und einem Kegel. Die Erde ist eine Kugel.

```
# Umlaute und Programm in deutscher Sprache laden:
# -*- coding: utf-8 -*-
from DePy import *

# Ab hier kannst Du Objekte erstellen und deren Attribute verändern,
# sowie Objekte zu Gruppen zusammenfügen und diese animieren:
planet = Kugel()
planet.name = 'Erde'
planet.radius = 10
planet.material = Materialien.Wolkenerde

rumpf = Zylinder()
rumpf.name = 'Rumpf'
rumpf.laenge = 5
rumpf.farbe = Farben.weiss
rumpf.material = Materialien.Granit

spitze = Kegel()
spitze.name = 'Spitze'
spitze.position = (5,0,0)
spitze.achse = (3,0,0)
spitze.farbe = (1,0.75,0)
spitze.material = Materialien.Plastik

rakete = Gruppe('Rakete', rumpf, spitze)
rakete.position = (10,0,0)

fenster.zentrum = (10,0,0)

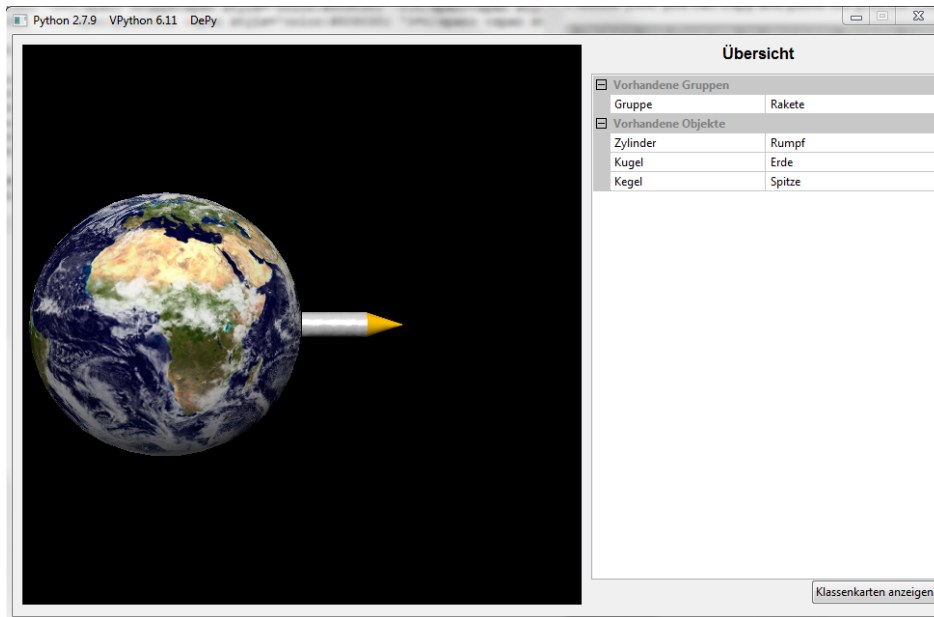
# Belasse folgende Zeile immer am Ende deines Programms:
objekteAnzeigen()
```

*dies ist der vom Schüler selbst programmierte Code*





Nach Start von obigem Programms erscheint folgende Oberfläche:



In der Übersicht werden alle vom Benutzer erzeugten Objekte und Gruppen angezeigt (hier Objekte Rumpf, Erde und Spitze und die Gruppe Rakete).

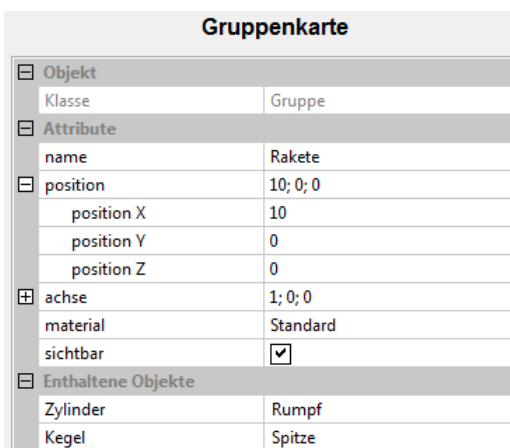
### Gruppen:

Um sich die Objektkarten der Gruppe(n) anzeigen zu lassen, klickt man auf den Text (hier: Gruppe bzw. Rakete).

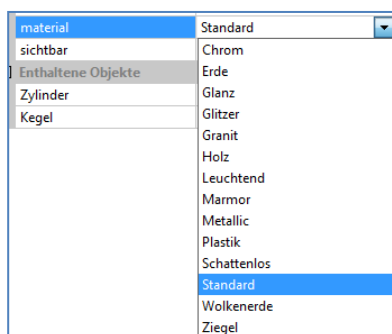
Es erscheint die Möglichkeit, das Objekt unsichtbar zu stellen (Auge) bzw. die Objektkarte darzustellen (Doppelpfeil).



### Objektkarte der Gruppe:



Sämtliche Attributwerte lassen sich durch Überschreiben ändern. Bei Klick auf `material` erscheint eine Liste aller möglichen Materialien.



Um wieder zur Übersicht zu gelangen, wird der Button "zurück zur Übersicht" geklickt.



## Allgemeine Objekte:

Um sich die zugehörige Objektkarte anzeigen zu lassen, gibt es zwei Varianten:

### *Variante 1:*

Klick auf den Text in der Übersicht (also z. B. auf Rumpf)

### *Variante 2:*

Klick auf das Objekt im Grafikfenster.

Analog der Gruppen-Objektkarte lassen sich auch hier die Attributwerte ändern.

## Klassenkarten:

Mit Hilfe des Buttons "Klassenkarten anzeigen" lässt sich eine Übersicht aller möglichen Klassen inkl. derer Attribute und Methoden darstellen.