

An abstract graphic featuring three blue circles of varying sizes (large, medium, and small) arranged vertically. Two thin, light blue diagonal lines intersect the circles, creating a sense of depth and perspective. The circles have a layered, 3D effect with lighter blue outlines.

Solarsystem

Dolacek | Özer

29.03.2015

Inhalt

Aufgabenstellung	3
Arbeitsschritte	4
Anforderungsanalyse	5
Designüberlegung	5
Sonnensystem(Relationen)	5
SW Design	7
Libaries.....	8
GUI Design	9
Entwicklung.....	10
Planeten	10
Texturen einbinden.....	10
Textur ein/aus	11
Licht ein/ aus.....	11
Kamera.....	11
Kameraposition	11
Geschwindigkeit schneller/ langsamer/ stoppen	12
Splashscreen	13
Quellen.....	14

Aufgabenstellung

Erstellen Sie eine einfache Animation unseres Sonnensystems:

In einem Team (2) sind folgende Anforderungen zu erfüllen.

- Ein zentraler Stern
- Zumindest 2 Planeten, die sich um die eigene Achse und in elliptischen Bahnen um den Zentralstern drehen
- Ein Planet hat zumindest einen Mond, der sich zusätzlich um seinen Planeten bewegt
- Kreativität ist gefragt: Weitere Planeten, Asteroiden, Galaxien,...
- Zumindest ein Planet wird mit einer Textur belegt (Erde, Mars,... sind im Netz verfügbar)

Events:

- Mittels Maus kann die Kameraposition angepasst werden: Zumindest eine Überkopf-Sicht und parallel der Planetenbahnen
- Da es sich um eine Animation handelt, kann diese auch gestoppt werden. Mittels Tasten kann die Geschwindigkeit gedrosselt und beschleunigt werden.
- Mittels Mausklick kann eine Punktlichtquelle und die Texturierung ein- und ausgeschaltet werden.
- Schatten: Auch Monde und Planeten werfen Schatten.

Hinweise:

- Ein Objekt kann einfach mittels `glutSolidSphere()` erstellt werden.
- Die Planeten werden mittels Modelkommandos bewegt: `glRotate()`, `glTranslate()`
- Die Kameraposition wird mittels `gluLookAt()` gesetzt
- Bedenken Sie bei der Perspektive, dass entfernte Objekte kleiner - nahe entsprechende größer darzustellen sind. Wichtig ist dabei auch eine möglichst glaubhafte Darstellung. `gluPerspective()`, `glFrustum()`
- Für das Einbetten einer Textur wird die Library Pillow benötigt! Die Community unterstützt Sie bei der Verwendung.

Arbeitsschritte

1. Gruppenbildung
2. Anforderungsanalyse (Funktionelle und nicht funktionelle Arbeitsanforderungen)
 - a. Tabellarische Auflistung
 - b. Rücksprache mit Auftraggeber
3. Designüberlegung
 - a. Prototyp- Evaluation
 - b. SW-Design
 - c. GUI-Design
 - d. Rücksprache mit Auftraggeber
 - e. UAT- Überlegung
4. Implementierung & Tests
5. Abgabemodalitäten klären & Abgabe

Anforderungsanalyse

Anforderung	Priorität	Verantwortlicher	Zeit in min		Status				
			SOLL	IST	Design	Implentiert	Test	Dokumentiert	Fertig
Zentraler Stern	High	Dolacek	60	180	X	X	X	X	X
Planet 1	High	Dolacek	70	120	X	X	X	X	X
Planet 2	High	Dolacek	70	120	X	X	X	X	X
Drehung P1	High	Dolacek	120	240	X	X	X	X	X
Drehung P2	High	Dolacek	120	240	X	X	X	X	X
Mond	High	Dolacek	60	120	X	X	X	X	X
Drehung Mond	High	Dolacek	100	120	X	X	X	X	X
Textur	High	Dolacek	80	210	X	X	X	X	X
Licht&Schatten	High	Dolacek	120	120	X	X	X	X	X
Kameraposition	Middle	Dolacek	120	180	X	X		X	
Geschwindigkeit der Animation	Middle	Dolacek	110	90	X	X	X	X	X
Testen	High	Özer, Dolacek	180	220	X	~		X	~
Kreativität(Hintergründe, usw)	Low	Özer, Dolacek	120	70	X	X	NICHT TESTBAR	X	X
Performance	Middle	Özer	90	30	X	X	X	X	X
Splashscreen	Middle	Özer	80	90	X	X	X	X	X
PyGame lernen	High	Özer	120	100	X	X	NICHT TESTBAR	X	X
Libaries finden	High	Özer	60	90	X	X	NICHT TESTBAR	X	X
Gesamt			28	39					
			Stunden	Stunden					

Designüberlegung

Sonnensystem(Relationen)



Größenangaben (Durchmesser - Ø) der einzelnen Planeten und der Sonne

<u>Sonne</u>	ca. 1.390.000 Kilometer	<u>Jupiter</u>	ca. 143.000 Kilometer
<u>Merkur</u>	ca. 4.900 Kilometer	<u>Saturn</u>	ca. 120.500 Kilometer
<u>Venus</u>	ca. 12.100 Kilometer	<u>Uranus</u>	ca. 51.100 Kilometer
<u>Erde</u>	ca. 12.800 Kilometer	<u>Neptun</u>	ca. 49.500 Kilometer
<u>Mars</u>	ca. 6.800 Kilometer		

Größenvergleich der Planeten mit Hilfe von „greifbaren“ Dingen

<u>Sonne</u>	ein großer Gymnastikball (Ø 1 Meter)	<u>Jupiter</u>	größerer Apfel (Ø 14 cm)
<u>Merkur</u>	kleine Erbse	<u>Saturn</u>	mittlerer Apfel (Ø 12 cm)
<u>Venus</u>	Kirsche	<u>Uranus</u>	Mandarine (Ø 5 cm)
<u>Erde</u>	Kirsche	<u>Neptun</u>	Mandarine (Ø 5 cm)
<u>Mars</u>	große Erbse		

Wenn man jetzt noch die Entfernungen untereinander berücksichtigen will, müsste z. B. der Jupiter in einer Entfernung von ca. 700 Metern zum Medizinball (Sonne) hingelegt werden. Im Weltall entspricht das einer Entfernung von 778.360.000 Kilometern (778,36 Mio. Kilometer). Unvorstellbar weit weg.

Die Relationen des Sonnensystems, siehe Link [\[5\]](#)

Es wurden die Planeten Erde und Venus gewählt.

Planet	Durchmesser	Umlaufzeit	Python Durchmesser	Python Umlaufzeit
Erde	12 800 km	365 Tage	0.1	0.03
Venus	12 100 km	224 Tage	0.1	0.02

#Erde

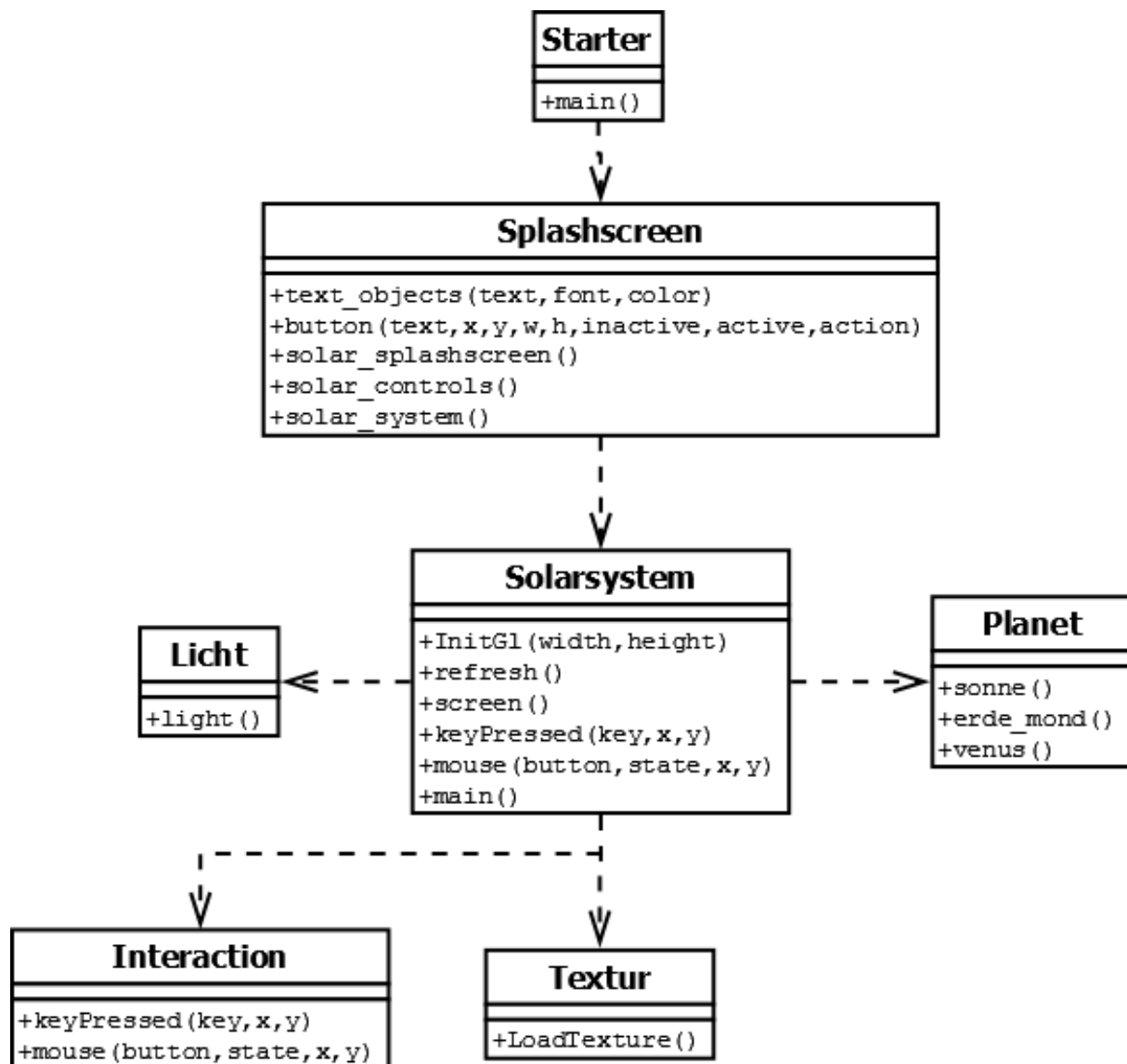
```
planet.DrawGLScene_erde(self, 0.1, 0.8, 0.0, 0.0, 0.03, self.erde)
```

```
planet.DrawGLScene_erde(self, 0.03, 1, 0.0, 0.0, 0, self.mond)
```

#Venus

```
Planet.DrawGLScene_venus(self, 0.1, 1.5, -0.3, 0.5, 0.02, self.rosa)
```

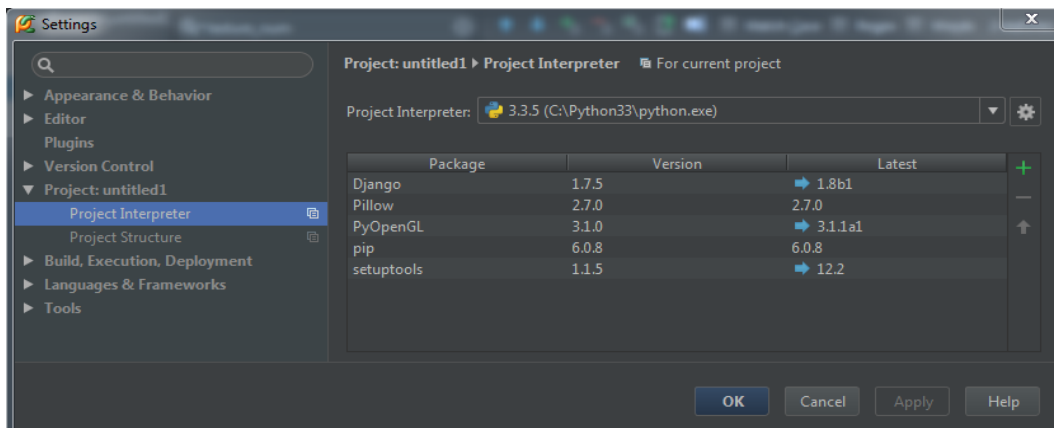
SW Design



Libaries

- Pygame **[3]**
- PyOpenGL
- Numpy
- Pillow

PyCharm -> File -> Settings -> Project Interpreter



Mögliche Fehler:

1. Für Glut wird eine zusätzliche .dll Datei aus dem Internet benötigt. Diese muss dann in dem Package eingebunden werden. Es gibt eine 32 und 64 Bit Version! Wenn Python 64bit installiert ist, muss die 64bit Version heruntergeladen werden.

<http://sourceforge.net/projects/freeglut/>

2. Bei dem Import muss man darauf achten, dass man die unterliegenden Packages richtig anspricht. D.h. Groß- und Kleinschreibung beachten.

GUI Design

Splashscreen:



Der Menüpunkt "About Us" wurde durch Controls ersetzt. Dem User wird beim Anklicken auf Controls eine Übersicht der Steuerung des Programms angezeigt.

Entwicklung

Planeten

```
gluSphere(quadratic, radius, 32, 32)
```



Texturen einbinden

für beliebige Objekte:

```
glBindTexture(GL_TEXTURE_2D, txt)  
quadratic = gluNewQuadric()  
gluQuadricNormals(quadratic, GLU_SMOOTH)  
gluQuadricTexture(quadratic, GL_TRUE)
```

Textur ein/aus

Die Textur wird mit der rechten Maustaste ein und ausgeschalten.

```
if (button==GLUT_RIGHT_BUTTON) :  
    if state == GLUT_DOWN:
```

Die Textur wird dann mit glEnable(GL_TEXTURE_2D) eingeschalten und mit glDisable(GL_TEXTURE_2D) ausgeschalten.

Licht ein/ aus

Die Licht wird mit der linken Maustaste ein und ausgeschalten.
Die Taste wird wie folg angesprochen:

```
if (button==GLUT_LEFT_BUTTON) :  
    if state == GLUT_DOWN:
```

[7]

Das Licht wird dann mit glEnable(GL_LIGHTING) eingeschalten und mit glDisable(GL_LIGHTING) ausgeschalten.

Kamera

Die Kamera wird mit gluLookAt definiert.
Die Kamera hat in diesem Programm eine Entfernung von 4 und der Vektor vom Objekt zum Aug ist 1/1/1.

```
gluLookAt(0, 0, 4, 0, 0, 0, 1, 1, 1)
```

Kameraposition

Die Kameraposition wird mit der Taste C geändert.
Es gibt nur eine Überkopf-Ansicht.

```
if key == b'c':
```

[8]

Die neue Kameraposition hat folgende Werte.

```
gluLookAt(0, 4, 4, 0, 0, 0, 1, 1, 1)
```

Geschwindigkeit schneller/ langsamer/ stoppen

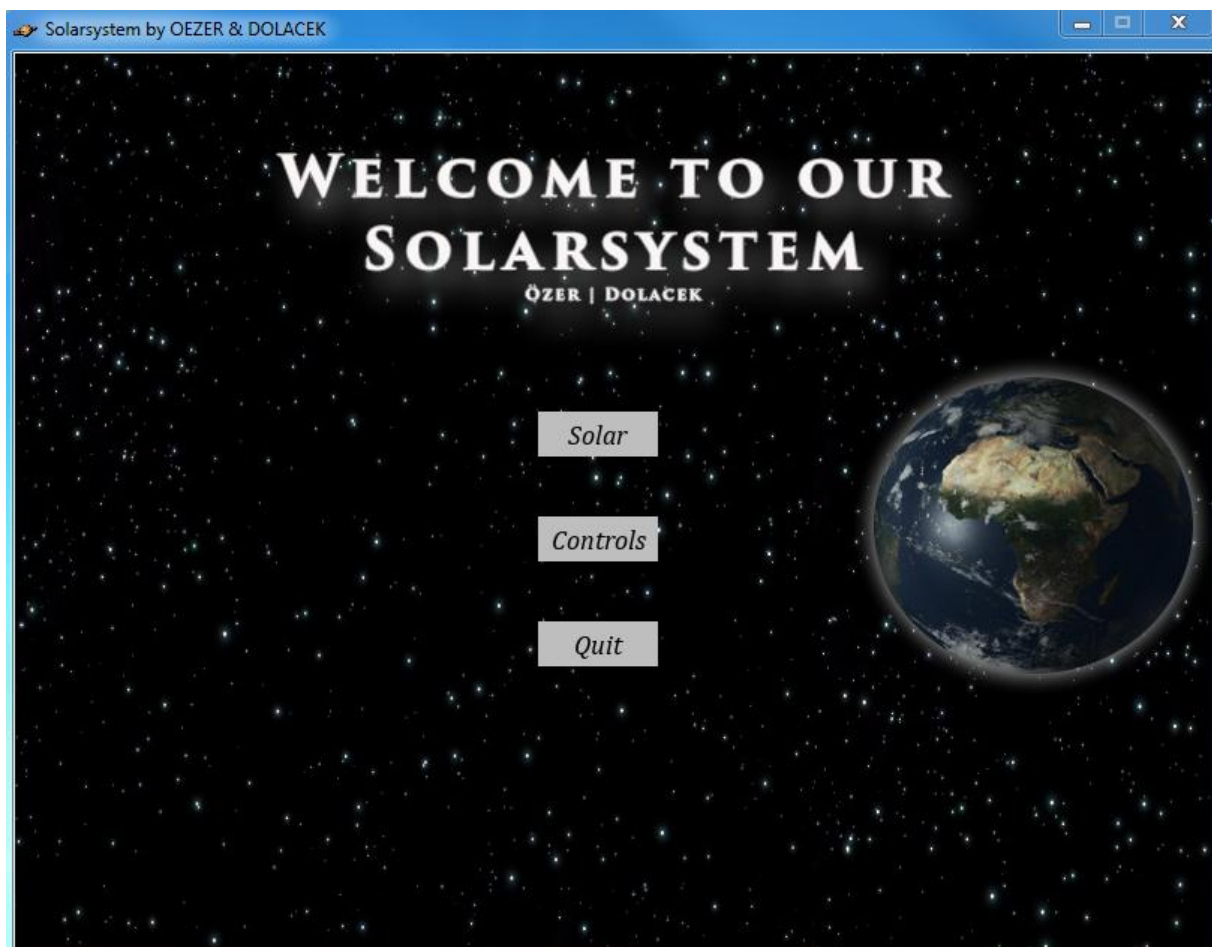
Die Animation wird mit d schneller gemacht, mit a langsamer und mit s gestoppt.

Je nachdem welche Taste gedrückt wurde, wird ein Wert dazu gezählt oder abgezogen.

Bei der gestoppten Animation ist alles auf 0.

Splashscreen

```
def solar_intro():  
    """  
    Solarsystem Splashscreen  
    :return:  
    """  
    intro = True  
  
    while intro:  
        for event in pygame.event.get():  
            if event.type == pygame.QUIT:  
                pygame.quit()  
                quit()  
            if event.type == pygame.KEYDOWN:  
                if event.key == pygame.K_c:  
                    intro = False  
                if event.key == pygame.K_q:
```



Quellen

[1] pydoc.net,
<http://pydoc.net/Python/PyOpenGL-Demo/3.0.1b1/PyOpenGL-Demo.NeHe.lesson18/>,
04.03.2015

[2] StackOverflow-User: eryksun,
<http://stackoverflow.com/questions/6756820/python-pil-image-tostring>,
08.03.2015

[3] Chistoph Gohlke,
<http://www.lfd.uci.edu/~gohlke/pythonlibs/#pygame>,
06.03.2015

[4] *"John F. Fay, John Tsiombikas, and Diederick C. Niehorster are the current maintainers of the FreeGLUT project".*,
<http://freeglut.sourceforge.net>,
10.03.2015

[5] Astronomie.de,
<http://www.astronomie.de/astronomie-fuer-kinder/interessantes-fuer-lehrer-eltern/in-der-schule/groessenvergleich-der-planeten/>,
16.03.2015

[6] deepsky.at,
<http://www.deepsky.at/tabellen/planeten.shtml>,
15.03.2015

[7] opengl.org,
<https://www.opengl.org/documentation/specs/glut/spec3/node50.html>,
18.03.2015

[8] swiftless.com,
<http://www.swiftless.com/tutorials/opengl/keyboard.html>,
18.03.2015