

Arbeitsblatt: DNET1

Name:

Kurznamen:

Verloren im Weltall

Die Bewegungen der Planeten werden durch die sogenannten Keplerschen Gesetze bestimmt. Diese gehen aber von einem Zentralgestirn aus, in dem der Grossteil der Masse konzentriert ist, was für unser Sonnensystem ja auch der Fall ist. Gäbe es dieses Zentralgestirn nicht, kämen kompliziertere Formeln zur Anwendung. Ab drei Massen existiert keine geschlossene Formel mehr, sondern die Bewegungen liessen sich nur noch numerisch bestimmen. Es sollen hier die Planetenbahnen ohne die Sonne berechnet werden. Wir ignorieren den Umstand, dass ohne diese es vermutlich niemanden geben würde, den diese Berechnung interessiert.




Aufgabe 1 und 2

Leiten Sie von diesem Himmelskörper *Orb*, die Klasse *Planet* und *Spaceship* ab (unterschiedliche Klassen, weil wir Spaceship später noch erweitern bzw. anpassen werden). In der Klasse *Orb* wird die Bewegung der Himmelskörper implementiert, während die Darstellung in *Planet* gemacht wird. Die Bewegung wird in einer späteren Aufgabe implementiert. Erstellen Sie ein Desktop Forms Projekt.

Create a new project

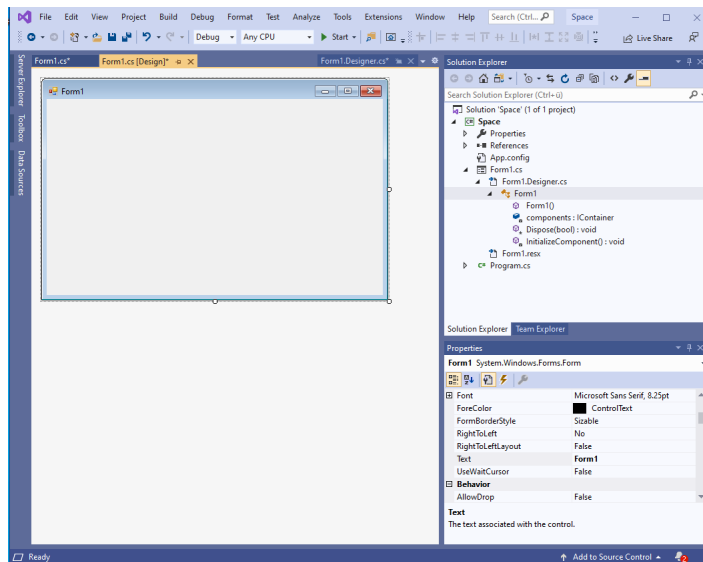
Desktop Forms

All languages All platforms All project types

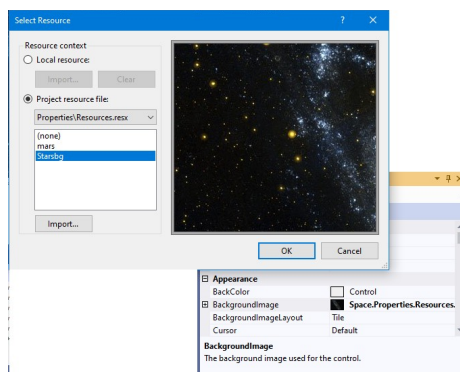
 Windows Forms App (.NET Framework)
A project for creating an application with a Windows Forms (WinForms) user interface

C# Windows Desktop

Und fügen Sie die Klassen *Orb*, *Planet* und *Spaceship* dem Projekt hinzu.



Setzen Sie zuerst den Sternenhintergrund über die Properties des Forms (als Resource) und fügen Sie bei dieser Gelegenheit gleich noch die andern Bilder als Ressourcen hinzu (über den gleichen Dialog)



Sie können dann nach dem entsprechenden Import (Projektname ist der Name den Sie für Ihr Projekt gewählt haben):

```
using <ProjectName>.Properties;
```

via die Resources Properties auf die Bilder zugreifen. Erstellen Sie für Testzwecke einen Knopf in Ihrem Form der Mars an der Stelle 100 100 in halber Grösse zeichnet. Einfach drag-and-drop aus der Toolbox und Doppelklick um eine Event Verarbeitungsmethode zu generieren. Dort kann dann das Bild gezeichnet werden:

```
Image image = Resources.mars;
```

oder via Namen zugegriffen werden

```
Image image = (Bitmap)Resources.ResourceManager.GetObject("mars");
```

Danach kann über das Graphics Object gezeichnet werden.

```
Graphics g = this.CreateGraphics();
```

```
g.DrawImage(image, 100, 100, image.Width/2, image.Height / 2);
```

Statt über einen Knopf sollen beim *Neuzeichnen des Forms* alle Himmelskörper über ihre draw Methoden gezeichnet werden. Zeichnen Sie die 3 Planeten und das Raumschiff zunächst an vordefinierten Positionen. Überschreiben Sie dafür die OnPaint Methode im Form und führen Sie eine Liste der Himmelskörper ein (im Form). Diese Liste wird bei der Erstellung des Form durch die 4 Himmelskörper

initialisiert. Bemerkung: Dass dadurch Darstellung und Logik nicht sauber getrennt wird, nehmen wird in Kauf.

Hinweise:

- Ähnlich wie bei Java existiert eine Methode OnPaint, die überschrieben werden kann:

```
protected override void OnPaint(PaintEventArgs e)
{
    Graphics g = e.Graphics;
    ...
}
```

- Graphics stellt einen Satz von Methoden zur Verfügung, mittels denen gezeichnet werden kann. Diese wird beim Aufruf der OnPaint Methode als Feld des Ereignisses mitgegeben. (e.Graphics)

Abgabe:

Praktikum: DT 4.1

Datei: Planet.cs

Abgabe:

Praktikum: DT4.2

Datei: Form1.cs

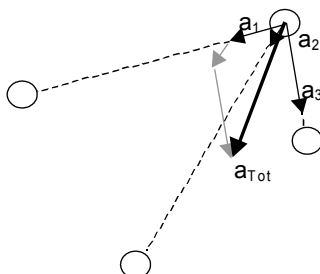
Aufgabe 3

→ noch ausprobieren und hochladen

Die Bewegungen sollen nun berechnet und angezeigt werden. Die Anziehungskraft zwischen zwei Massen berechnet sich aus (wie Ihnen sicherlich aus der Physik noch geläufig ist) $F = G * (M_1 * M_2) / r^2$. Die für eine Beschleunigung einer Masse notwendige Kraft berechnet sich aus $F = a * M$. Daraus lassen sich die resultierenden Beschleunigungen der einzelnen Massen in unserem System berechnen. Die resultierende Beschleunigung lässt sich einfach durch vektorielle Addition der einzelnen Beschleunigungskomponenten berechnen. Für die Berechnung der Beschleunigungskomponente hat sich folgende Formel bewährt:

$$\vec{a}_1 = |a_1| * \frac{\vec{r}_1}{|\vec{r}_1|}$$

Einheitsvektor in Richtung r



Gehen Sie im Beispiel von einem relativen Massenverhältnis der Planeten von 100 (Jupiter) zu 5 (Mars) zu 4 (Merkur) aus. Die Enterprise habe (wegen des Warp Antriebes und der geladenen Antimaterie) eine für ein Raumschiff enorme relative Masse von 1.

Hinweise:

- Die Gesamtbeschleunigung einer Masse setzt sich aus den Einzelbeschleunigungen zusammen. Achten Sie aber dabei, dass sie zuerst **sämtliche** Beschleunigungen und neuen Geschwindigkeiten berechnen und anschliessend die Massenpunkte verschieben (zwei Durchläufe).
- Für die Animation kann die Timer Klasse verwendet werden, das einfach mittels drag-and-drop aus der Toolbox in das Form gezogen werden kann.
- Setzen Sie im Konstruktor der Form Klasse die entsprechende Timer Properties

```
this.timer1.Enabled = true;  
this.timer1.Interval = 50;  
this.timer1.Tick += new System.EventHandler(this.timer1_Tick);
```
- Fügen Sie eine private void timer1_Tick(object sender, EventArgs e) Methode hinzu, die einen Refresh Aufruf des Neuzeichnen auslöst und damit einen nächsten Schritt der Planetensimulation.
- Folgende Anfangswerte sollten den Mars um den Jupiter kreisen lassen.

```
space.Add(new Planet("jupiter", 600, 400, -0.038, 0, 100));  
space.Add(new Planet("mars", 600, 600, 3.8, 0, 1));
```
- Folgende Anfangswerte sollten ein chaotisches Verhalten zeigen

```
spaceship = new Spaceship(600, 230, 3, 0, 1);  
space.Add(spaceship);  
space.Add(new Planet("jupiter", 600, 400, -0.099, 0, 100));  
space.Add(new Planet("mars", 300, 400, 0, 1.5, 4));  
space.Add(new Planet("merkur", 200, 200, 0, -1.5, 4));
```

Abgabe:

Praktikum: DT4.3

Datei: Orb.cs