Modul 24

Objektorientierte Programmierung

VITRUVIUS HOCHSCHULE

Oliver Ziegler

Überblick

- **Namespaces**
- **Unity-Documentation Research**

- **Funktionen**
  - **Parameter**
  - **Rückgabewerte**

Namespaces


```
Student myStudent;
```

Die Klasse Student ist nicht zu finden, weil sie sich im NS Vitruvius befindet

```
Vitruvius.Student myStudent;
```

Explizite Defintion zur Nutzung der Bibiliothek

Using vereint den Namespace mit der aktuellen Ebene
 Ohne Namespace-Angabe nutzbar

```
using Vitruvius;

Student myStudent;
```

Namespaces

```
class Student
public int semester;
```

```
Student myStudent;
```

Bezieht sich auf die selbst definierte Klasse

```
Vitruvius.Student myStudent2;
```

Explizite Defintion zur Nutzung der Bibiliothek

```
Namespace Vitruvius
class Student
public int semester;
```

```
using Vitruvius;

Student myStudent2;
```

Führt zu einer Unklarheit, welche Klasse gemeint ist

Verschachtelte Namespaces


```
Vitruvius.GameDesign.Unternehmenspr  uProjekt;
```

Explizite Variablen - Definition

```
using Vitruvius.GameDesign;
using Vitruvius.ModeDesign;

Unternehmenspr uProjekt;
Kleidung   jacke;
```

Using vereint den Namespace mit der aktuellen Ebene

 Ohne Namespace-Angabe nutzbar

Namespaces

Namespace Vitruvius

class Student

```
public int semester;
```

Namespace ExterneBib

class Student

```
public int klassenStufe;
```

```
Student myStudent;
```

Die Klasse Student ist nicht zu finden

```
Vitruvius.Student myStudent;
ExterneBib.Student myStudent2;
```

Explizite Definition

```
using Vitruvius;

Student myStudent;
ExterneBib.Student myStudent2;
```

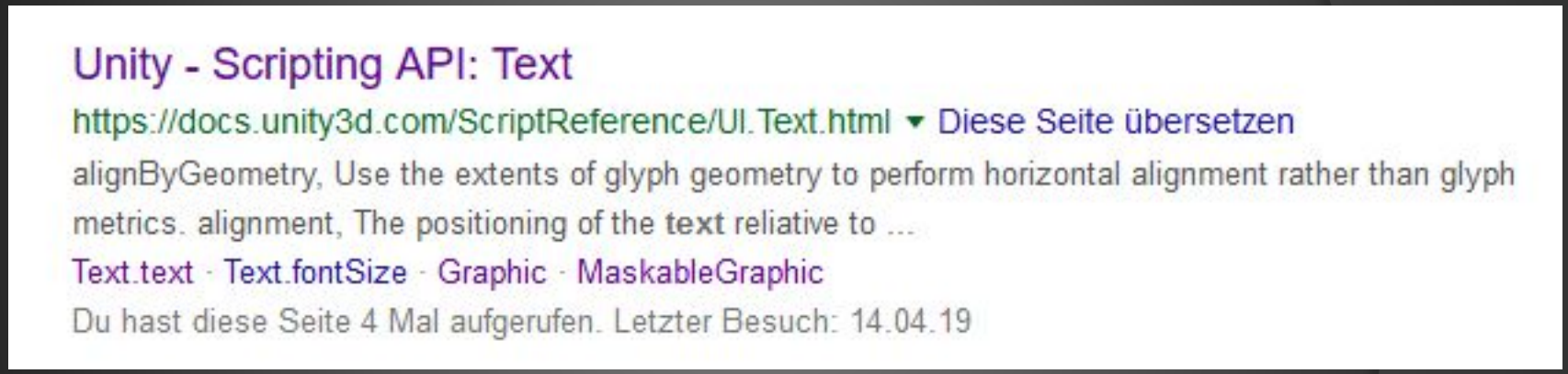
„using“ des häufig benutzten Namespaces
Explizit für selten benutzt

Documentation Research

- **Suchen nach Funktion oder Variable einer Klasse**
  - **Unity-Documentation**
  - **Internet-Research, Foren**

- **Beispiel: UnityEngine.UI.Text.text**

Google: „Unity Text“



Google: „Unity Text“

unity | DOCUMENTATION

Version: **2018.3** (switch to 2019.1b or 2017.4)

- Navigation
- Outline
- PositionAsUV1
- RawImage
- RectMask2D
- Scrollbar
- ScrollRect
- Selectable
- Shadow
- Slider
- SpriteState
- Text
- Toggle
- ToggleGroup
- VertexHelper
- VerticalLayoutGroup
- Interfaces
- Enumerations
- UnityEngine.Video
- UnityEngine.Windows
- UnityEngine.WSA
- UnityEngine.XR
- Classes
- Interfaces

Text

class in UnityEngine.UI / Inherits from:UI.MaskableGraphic
Implements interfaces:ILayoutElement

Class in **UnityEngine.UI**

Descri

The default Graphic to draw font data to screen.

Properties

alignByGeometry	Use the extents of glyph geometry to perform horizontal alignment r
alignment	The positioning of the text reliative to its RectTransform.
cachedTextGenerator	The cached TextGenerator used when generating visible Text.
cachedTextGeneratorForLayout	The cached TextGenerator used when determine Layout.
flexibleHeight	Called by the layout system.
flexibleWidth	Called by the layout system.
font	The Font used by the text.
fontSize	The size that the Font should render at.
fontStyle	FontStyle used by the text.
horizontalOverflow	Horizontal overflow mode.

Google: „Unity Text“

unity | DOCUMENTATION

Version: **2018.3** (switch to 2019.1b or 2017.4)

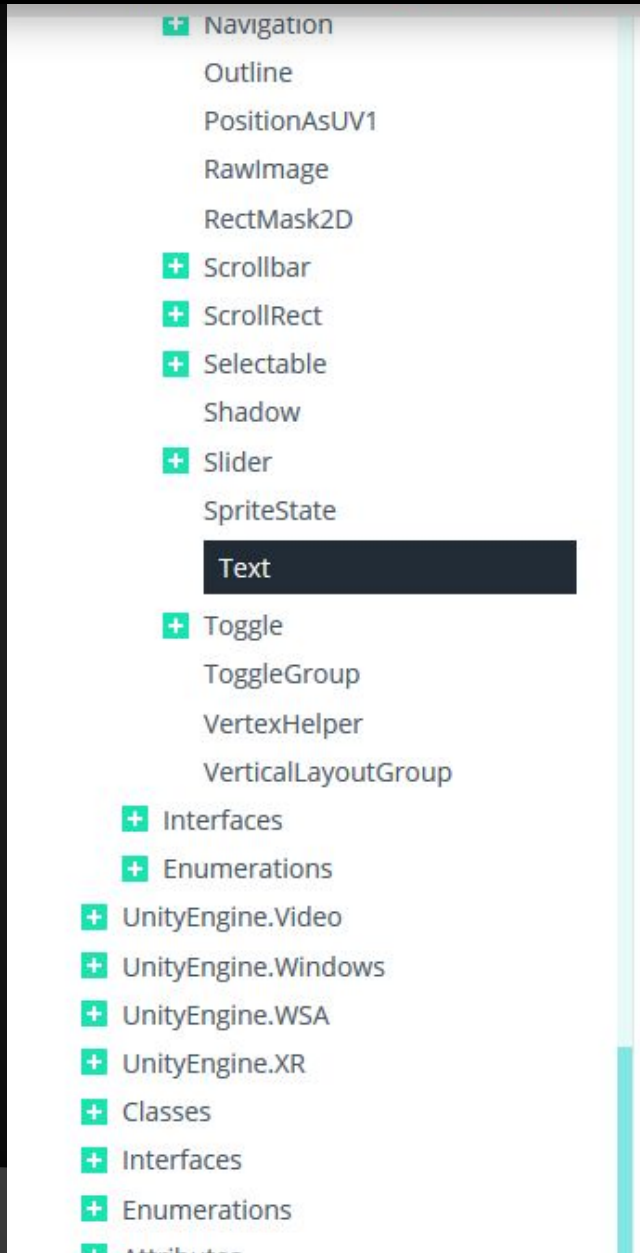
- Navigation
- Outline
- PositionAsUV1
- RawImage
- RectMask2D
- Scrollbar
- ScrollRect
- Selectable
- Shadow
- Slider
- SpriteState
- Text
- Toggle
- ToggleGroup
- VertexHelper
- VerticalLayoutGroup
- Interfaces
- Enumerations
- UnityEngine.Video
- UnityEngine.Windows
- UnityEngine.WSA

horizontalOverflow	Horizontal overflow mode.
layoutPriority	Called by the layout system.
lineSpacing	Line spacing, specified as a factor of font line height.
mainTexture	The Texture that comes from the Font.
minHeight	Called by the layout system.
minWidth	Called by the layout system.
pixelsPerUnit	(Read Only) Provides information about how fonts ar
preferredHeight	
preferredWidth	
resizeTextForBestFit	Should the text be allowed to auto resized.
resizeTextMaxSize	The maximum size the text is allowed to be. 1 = infin
resizeTextMinSize	The minimum size the text is allowed to be.
supportRichText	Whether this Text will support rich text.
text	The string value this Text displays.
ve	Vertical overflow mode.

string value the Text displays

Click

Google: „Unity Text“



Text.text

public string **text**;

Description

The `string` value this Text displays.

Use this to access or edit the message displayed in Text. Use other Text properties such as size, f
properties modify the text in this example:

```
using UnityEngine;
using UnityEngine.UI;

//
//
// A Spac                                    n.
// Two me
//
// Inside Awake a Canvas and Text are created.

public class Example : MonoBehaviour
{
    private enum UpDown { Down = -1, Start = 0, Up = 1 };
    private Text text;
    private UpDown textChanged = UpDown.Start;
```

using UnityEngine.UI

private UnityEngine.UI.Text text

private Text text

Google: „Unity Text“

unity | DOCUMENTATION

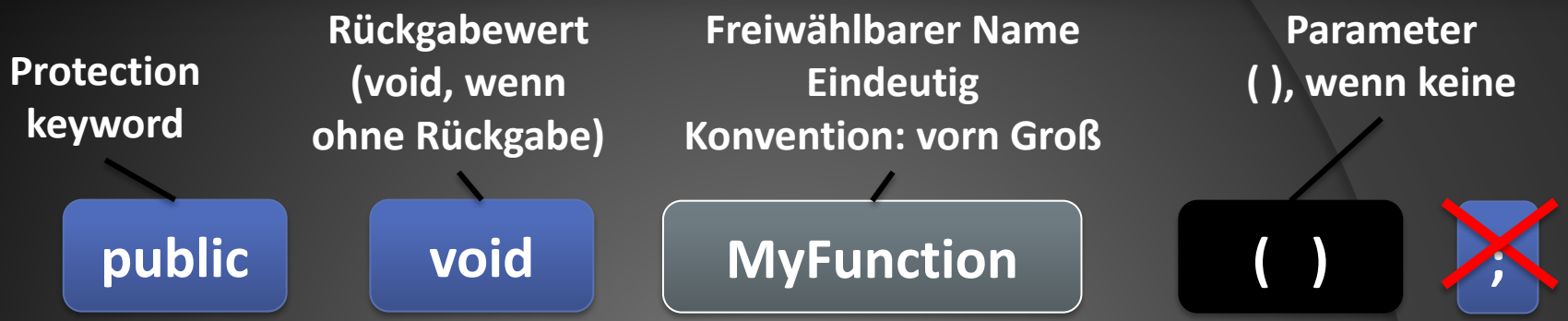
Version: **2018.3** (switch to 2019.1b or 2017.4)

- Navigation
- Outline
- PositionAsUV1
- RawImage
- RectMask2D
- Scrollbar
- ScrollRect
- Selectable
- Shadow
- Slider
- SpriteState
- Text
- Toggle
- ToggleGroup
- VertexHelper
- VerticalLayoutGroup
- Interfaces
- Enumerations
- UnityEngine.Video
- UnityEngine.Windows

text.text = „Press space key“;
Text.fontSize = 48;
Text.alignment = TextAnchor.MiddleCenter;

```
canvas.renderMode = RenderMode.ScreenSpaceOverlay;

// Create the Text GameObject.
GameObject textGO = new GameObject();
textGO.transform.parent = canvasGO.transform;
textGO.AddComponent<Text>();

// Set Text component properties.
text = textGO.GetComponent<Text>();
text.font = arial;
text.text = "Press space key";
text.fontSize = 48;
text.alignment = TextAnchor.MiddleCenter;

// Provide Text position and size using RectTransform.
RectTransform rectTransform;
rectTransform = text.GetComponent<RectTransform>();
```

Funktionsdefinition




```
{

Inhalt/ Befehle, jeweils mit Semikolon

}
```

KEIN SEMIKOLON!

Die Funktion wird beschrieben, aber nicht ausgeführt, kein Speicher vorbereitet Kein Befehl

Return Funktion

void MyFunction( )

```
{
DoStuff();
DoSomeMoreStuff();

return;

ThisStuffWillNeverBeDone();
}
```

- **Jede Funktion kann durch einen simplen (leeren) return-Befehl beendet werden.**

Rückgabewert

- Ein errechneter oder ausgelesener Wert wird als Ergebnis, wie eine Variable zurückgegeben
- Komplexe Funktionen, die ein Ergebnis bereitstellen, einmal definieren und oft verwenden
- Meist mit Parameter

void MyFunction( )

```
{
string aktuelleZeit  = AktuelleZeitAlsText();

DateTime geburtsdatum;
int alter = ErrechneAlter( geburtsdatum );
}
```

string AktuelleZeitAlsText()

int ErrechneAlter
(DateTime gebDatum)

Funktion mit Rückgabewert

- Definieren dieser Funktionsart
- Datentyp des Rückgabewerts definieren

```
void MyFunction()
{
int eineZahl = 5;GibFuenf( );
Debug.Log ( eineZahl ) ;
}
```

Funktion wird zur Laufzeit durch Rückgabewert ersetzt.

```
int GibFuenf( )
{
return 5;
}
```

Return-Befehl muss vorhanden sein

Funktion mit Rückgabewert

- **Definieren dieser Funktionsart**

void = leer
(Keine Rückgabe)

```
void GreifeGegnerAn()
{
schadenGesamt = grundwert + Waffenschaden();

schadenGesamt = schadenGesamt * SchadenMultiplikatorDurchBuff();

Gegner.ReceiveDamage( schadenGesamt );
}
```

Datentyp des Rückgabe-Werts

Rückgabe-Befehl

```
int Waffenschaden( )
{
int schaden = waffe.schaden + waffe.verzauberung + staerke;
return schaden;
}
```

Funktionsvariablen

class MyComponent

```
public int myGlobalNumber;
```

void MyFunction()

```
public int myNumber;

myGlobalNumber = 10;
myNumber = 55;
```

void MyFunction2()

```
public int myNumber;

myGlobalNumber = 1000;
myNumber = myNumber + 30;
```

Klassenvariablen

- Werden auf derselben Ebene wie Funktionen definiert
- Erscheinen im Inspector
- Werden bis zum Szenenwechsel gespeichert

Funktionsvariablen

- Werden innerhalb von Funktionen definiert
- Erscheinen NICHT im Inspector
- Werden nach der Funktion gelöscht

Klassen/Funktionsvariablen

KlassenVariablen

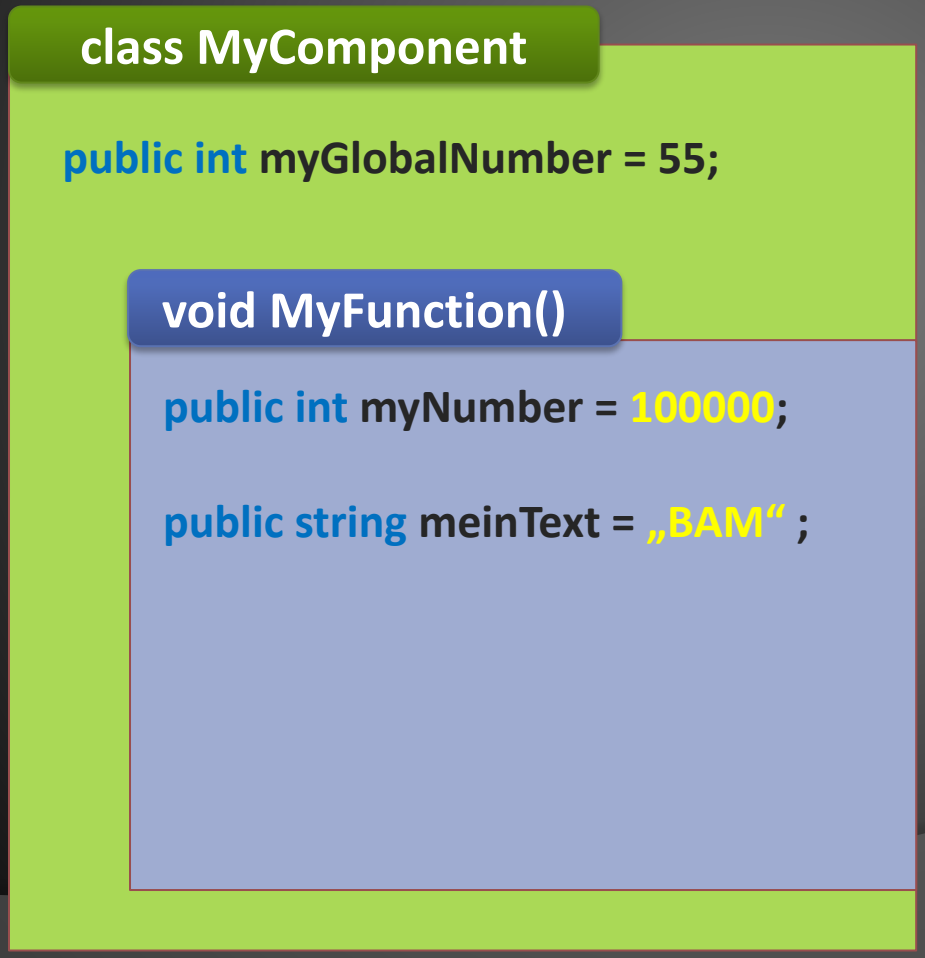
- werden außerhalb von Funktionen definiert
- Sie sind von allen Funktionen ansprechbar (intern und extern)
- Sie werden mit der Instanz gespeichert

FunktionsVariablen

- werden innerhalb einer Funktion definiert
- Sie existieren nur innerhalb dieser Funktion
- Inhalt/Speicher wird nach dem letzten Befehl wieder gelöscht

Funktionsvariablen

- **Variablen können übrigens schon bei der Definition mit Daten gefüllt werden**



Funktionsvariablen

class MyComponent

```
public int myGlobalNumber;
```

void SuperFunction()

```
public int myNumber = 5;
myNumber = 1000;
myGlobalNumber = myGlobalNumber +
        myNumber;
```

void AwesomeFunction()

```
public int myNumber;

myGlobalNumber = myGlobalNumber +
            myNumber;
```

Wie groß ist „myGlobalNumber“ am Ende?

```
myGlobalNumber = 20;

SuperFunction();

AwesomeFunction();

SuperFunction();
```

Funktion: Parameter

- Ermöglichen Funktionen ohne globale Variablen situationsspezifische Aufgaben zu erfüllen

Funktion 1

```
int x = Stuff();
MachEtwas();
EngineGo();
Hardware = x;
```

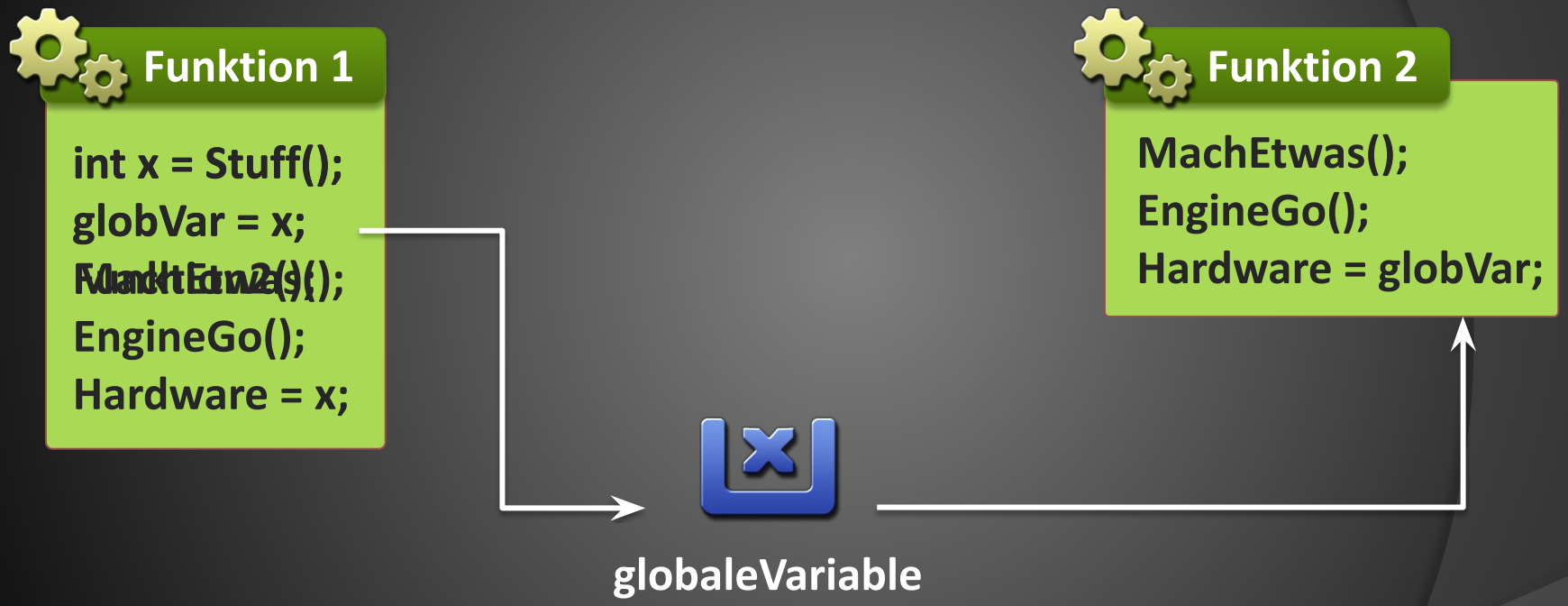
Funktion 2

```
MachEtwas();
EngineGo();
```

- Ohne Variablenbenutzung, kann die Funktion 2 nur immer gleiche Befehle ausführen

Funktion: Parameter

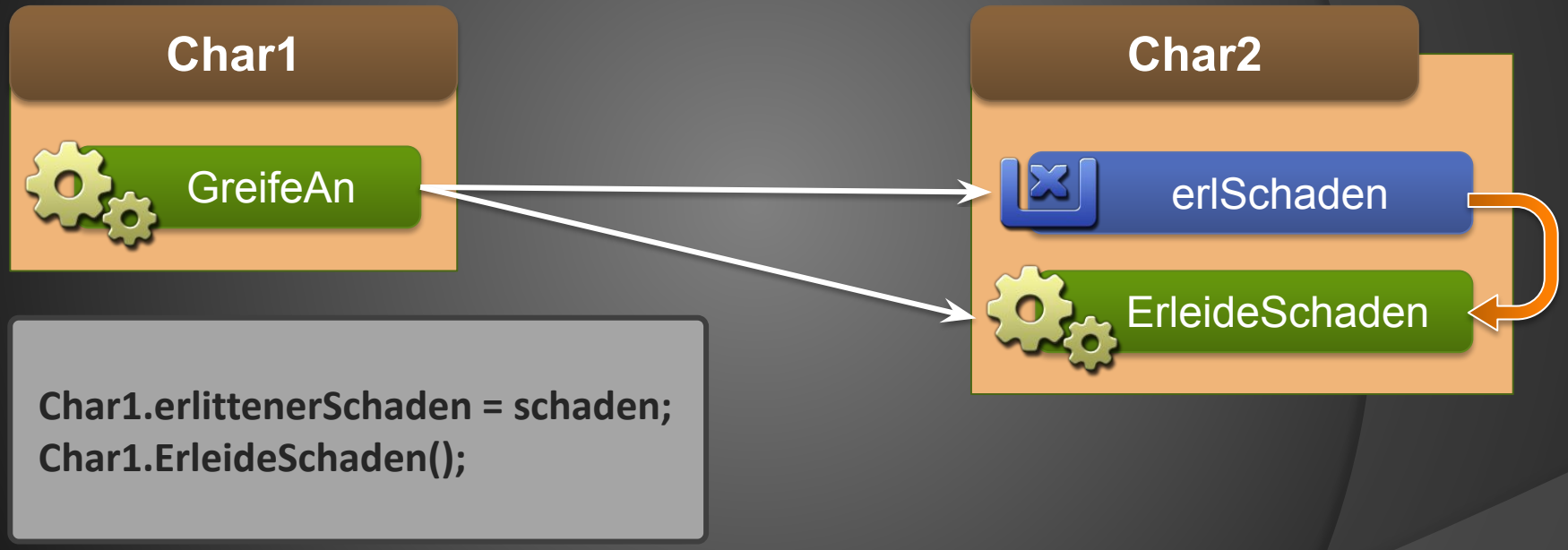
- Ermöglichen Funktionen ohne globale Variablen situationsspezifische Aufgaben zu erfüllen



- Für die Variablennutzung müssten Variablen angelegt werden, die für jeden erreichbar sind

Funktion: Parameter

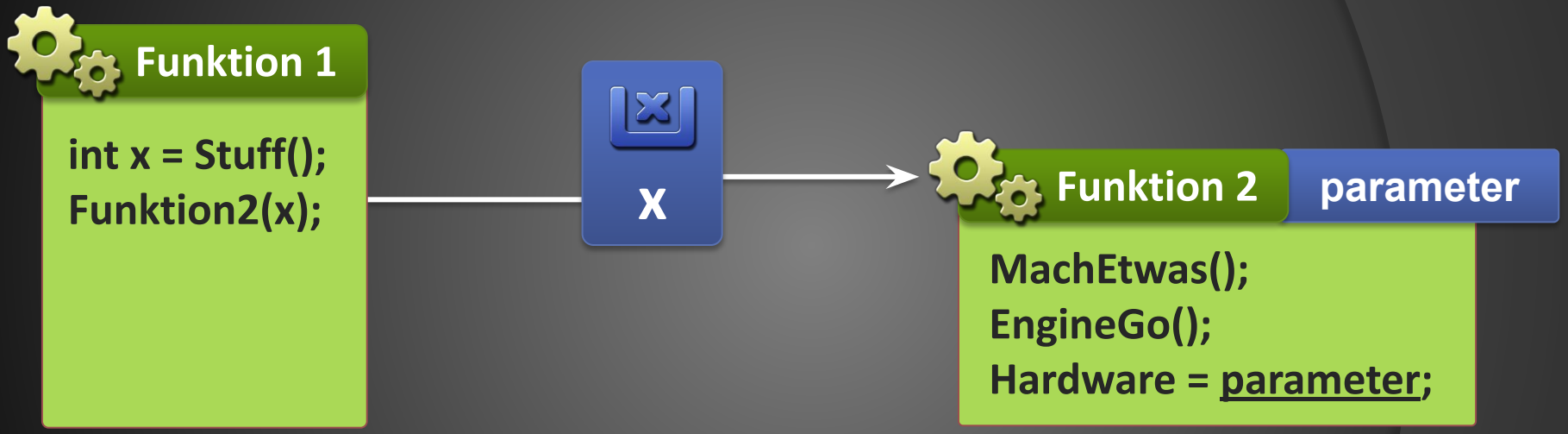
- **Ermöglichen Funktionen ohne globale Variablen situationsspezifische Aufgaben zu erfüllen**



- **In anderen Objekten müssten zunächst Variablen gesetzt und dann die entsprechende Funktion ausgeführt werden**
- **□ Diese Variable müsste für jeden zugänglich sein**

Funktion: Parameter

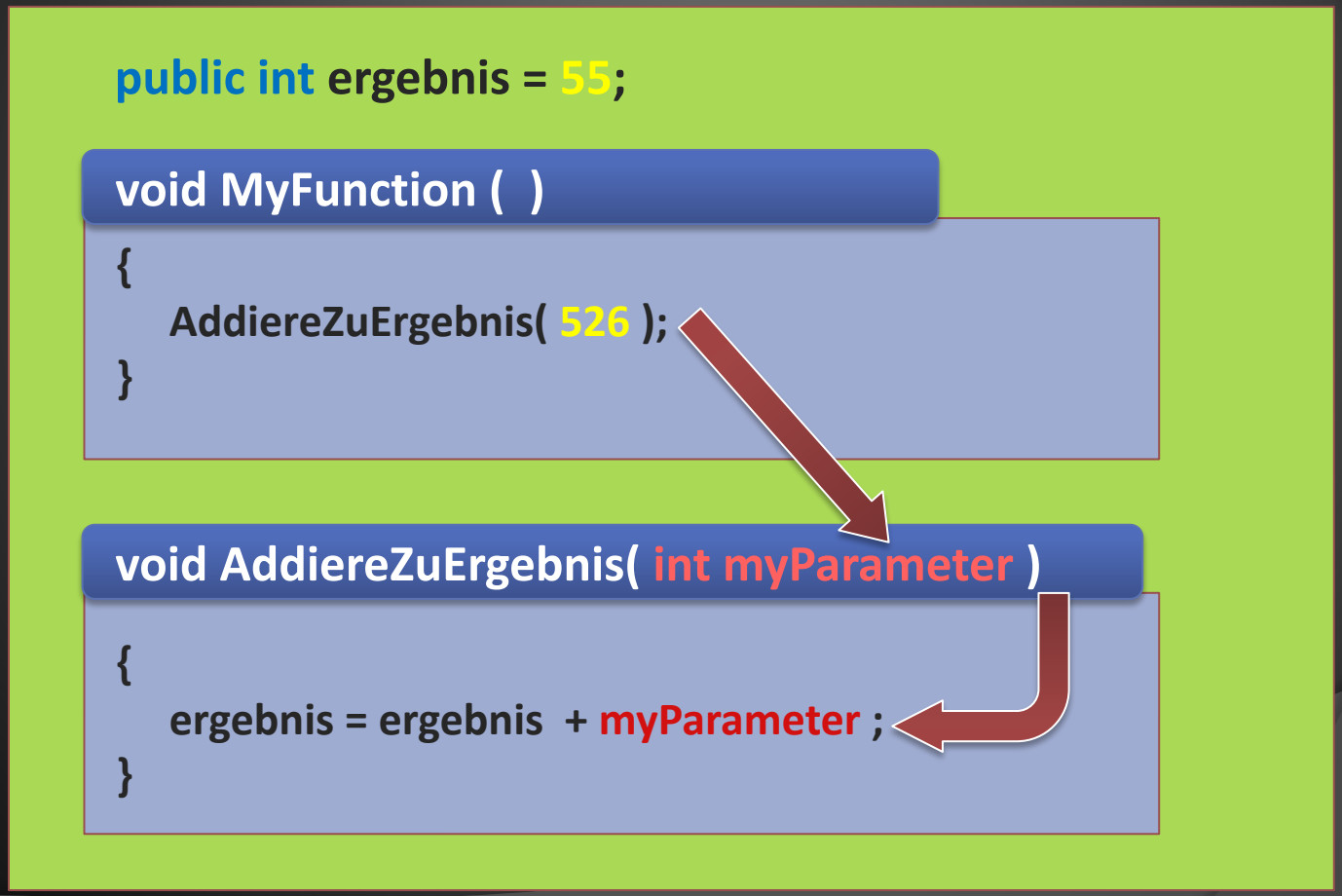
- Ermöglichen Funktionen ohne globale Variablen situationsspezifische Aufgaben zu erfüllen



- Der Wert der Variable wird übergeben, ohne zusätzlichen Befehl
- Wie bei Funktionsvariablen, wird der Parameter nach der Funktion gelöscht.

Funktion: Parameter

- **Erlauben es, Daten als Funktionsvariablen in die Funktion mitzugeben**



Parameter

- **Funktionsvariablen, die von außen gesetzt werden**

```
Debug.Log( „Das ist ein Test“ );
```

```
public class Debug
{

    public void Log( string text )
    {
        DisplayInConsole(  text  ) ;
    }

}
```

Parameter

- Ermöglichen das Auslagern eines Vorgangs in eine andere Funktion
- Erhöhen so Lesbarkeit und senken die Veränderungszeit

```
public void ReceiveDamage( int  damage )
{
        text = damage;
        text.animationStyle = 1;   // normaler Schaden
        text.StarteAnimation();
        leben = leben – damage;
       if( leben <= 0 )
       {
             leben = 0;
             StarteTodesAnimation();
             ZeigeGameOverScreen();
             LassGegnerLachen();
       }
}
```

Parameter

- Ermöglichen das Auslagern eines Vorgangs in eine andere Funktion
- Erhöhen so Lesbarkeit und senken die Veränderungszeit

```
public void ReceiveDamage( int  damage )
{
        DisplayDamage( damage );
        text = damage;
        text.animationStyle = 1;  // normaler Schaden
        text.StarteAnimation();
        leben = leben – damage;
}
public void DisplayDamage( int damage)
{
        text = damage;
        text.animationStyle = 1;   // normaler Schaden
        text.StarteAnimation();
}
```

Parameter

- Ermöglichen das Auslagern eines Vorgangs in eine andere Funktion

```
public void ReceiveDamage( int  damage )
{
        DisplayDamage( damage );
        leben = leben – damage;
       if( leben <= 0 )
       {
              leben = 0;
              StarteTodesAnimation();
              ZeigeGameOverScreen();
              LassGegnerLachen();
       }
}
```

Parameter

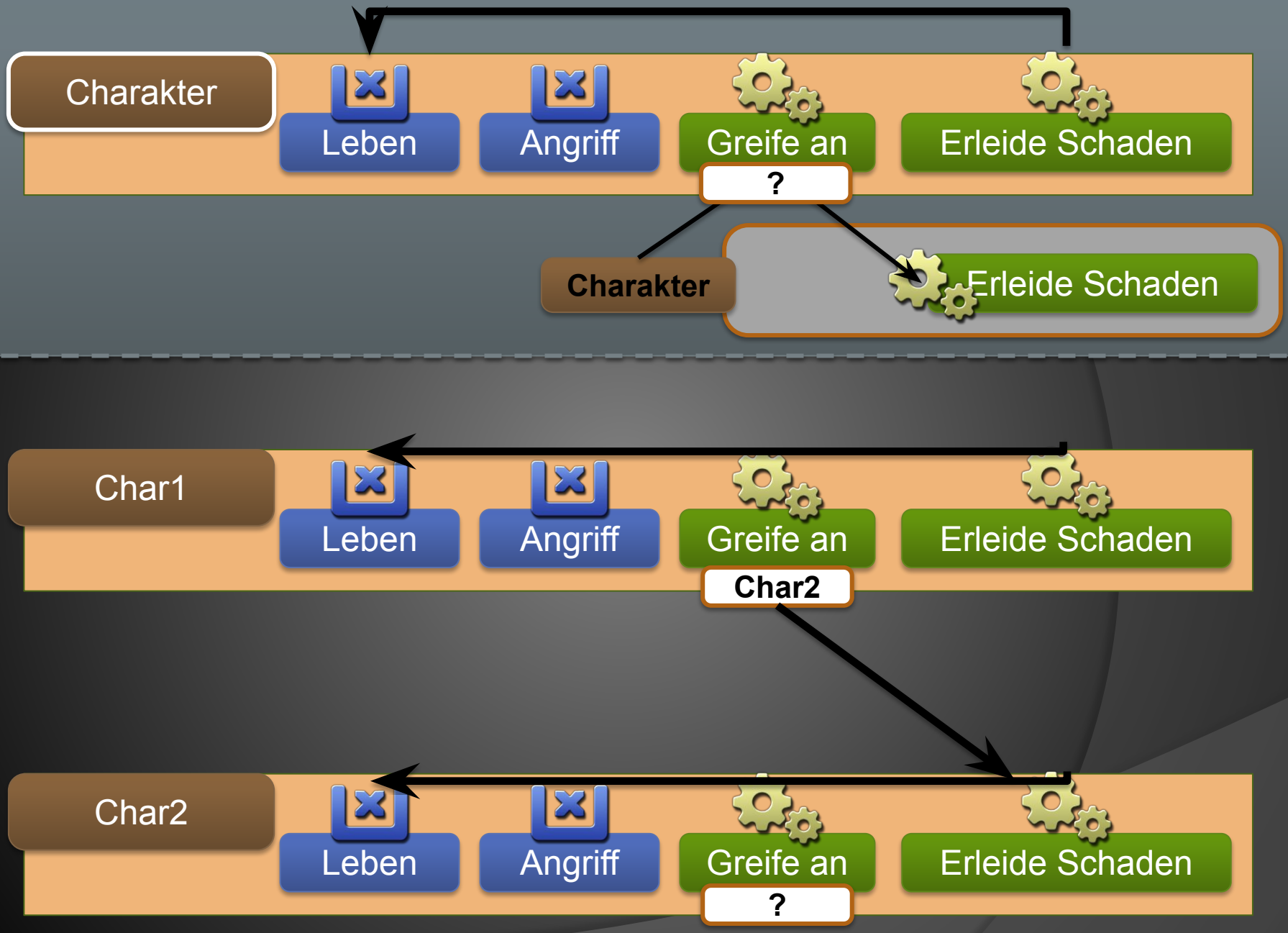
- Ermöglichen das Auslagern eines Vorgangs in eine andere Funktion

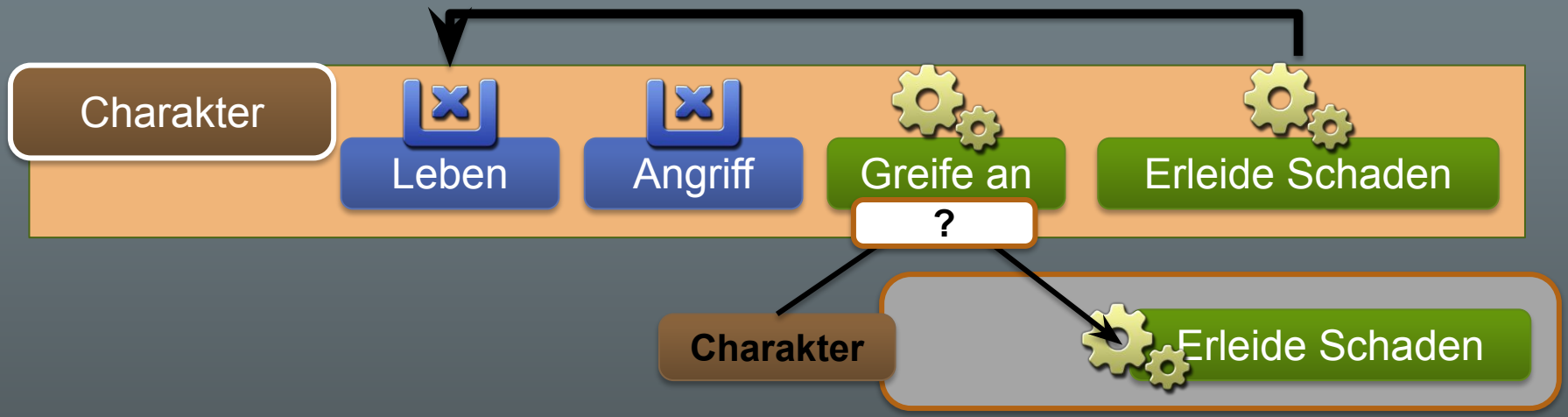
```
public void ReceiveDamage( int  damage )
{
        DisplayDamage( damage );
        leben = leben – damage;
}

public void AffectDamageOnLive( int damage )
{
      leben = leben – damage;
      if( leben <= 0 )
      {
            leben = 0;
            StarteTodesAnimation();
            ZeigeGameOverScreen();
            LassGegnerLachen();
      }
}
```

Parameter

- Ermöglichen das Auslagern eines Vorgangs in eine andere Funktion

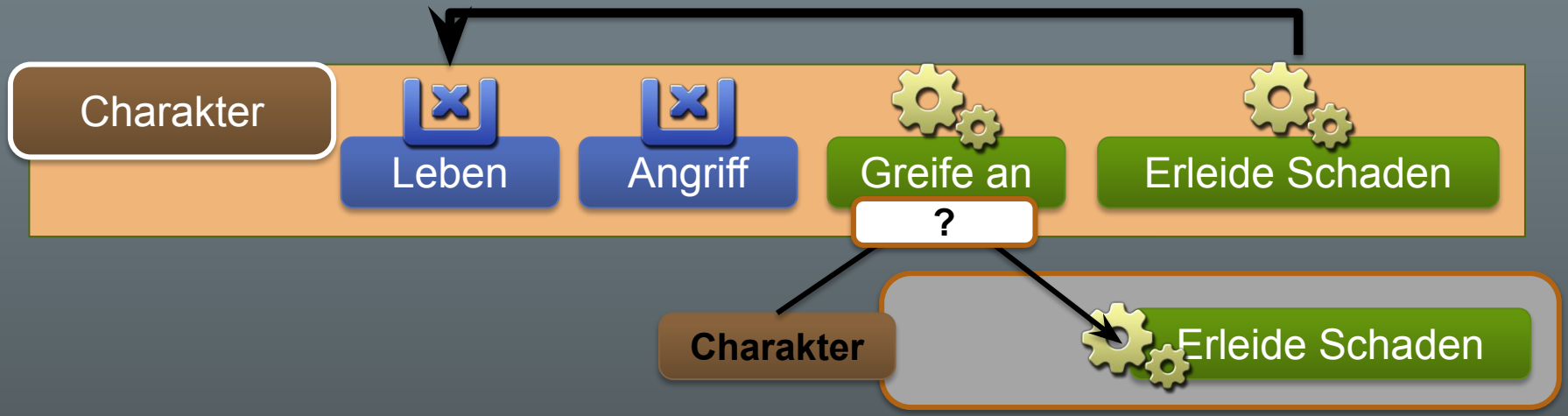
```
public void ReceiveDamage( int  damage )
{
        DisplayDamage( damage );
        AffectDamageOnLive( damage );
}

public void AffectDamageOnLive( int damage )
{
      leben = leben – damage;
      if( leben <= 0 )
      {
            leben = 0;
            StarteTodesAnimation();
            ZeigeGameOverScreen();
            LassGegnerLachen();
      }
}
```


Charakter
Leben
Angriff
Greife an
Erleide Schaden
?
Charakter
Erleide Schaden
Char1
Leben
Angriff
Greife an
Erleide Schaden
Char2
Char2
Leben
Angriff
Greife an
Erleide Schaden
?




```
public int leben;
public int angriff;

public void GreifeAn( Character targetChar)
{
    targetChar.ErleideSchaden( angriff );
}

public void ErleideSchaden( int schaden )
{
    leben = leben – schaden;
}
```

Objekt-Wechsel




```
public int leben;
public int angriff;

public int level;

public void GreifeAn( Character targetChar )
{
    int schaden = (angriff * 4) * (level / 2);
    targetChar.ErleideSchaden( schaden );

}
```

Parameter + Rückgabewert

Parameter + Rückgabewert

```
public void GreifeAn( Charakter gegner )
{
   int schaden = waffe.staerke + waffe.verzauberung + staerke;
   schaden = schaden * level;

   int verteidigung = gegner.vert + gegner.schild.vert;
   verteidigung = verteidigung * gegner.level;

   schaden = schaden – verteidigung;
   gegner.ReceiveDamage( schaden );
}
```

Parameter + Rückgabewert

```
public void GreifeAn( Charakter gegner )
{
   int schaden = CalculatedDamage( );

   int verteidigung = CalculatedDefense( gegner );

   schaden = schaden – verteidigung;
   gegner.ReceiveDamage( schaden );
}
```

Parameter + Rückgabewert

```
public void GreifeAn( Charakter gegner )
{
   int schaden = CalculatedMyDamage( );
   schaden = CalculatedDamageAfterDefense( gegner, schaden );
   gegner.ReceiveDamage( schaden );
}
```

```
public int CalculateDamageAfterDefense( Charakter gegner, int schaden)
{
     int verteidigung = gegner.verteidigung + gegner.schild.verteidigung;
     verteidigung = verteidigung * gegner.level;
     schaden = schaden – verteidigung;
     return schaden;
}
```