Software Development

Spezielle Klassen

Software-Lösungen. Einfach. Clever.

EGELI informatik

Inhalt

- Abstrakte Klassen und Methoden
- Versiegelte Klassen
- Partielle Klassen
- Statische Klassen

Abstrakte Klasse

- Klassen, die lediglich ihr "Erbmaterial" an andere Klassen weitergeben und von denen selbst keine Instanzen gebildet werden, bezeichnet man als abstrakt
- Um zu verhindern, dass von abstrakten Klassen Instanzen gebildet werden, müssen diese mit dem Schlüsselwort **`abstract`** gekennzeichnet werden

C #

```
public abstract class BasisKlasse
{…}

// Deklaration ist nach wie vor möglich:
BasisKlasse basisKlasse;

// Instanzierung schlägt aber fehl:
basisklasse = new BasisKlasse(); // → Fehler
```

Abstrakte Klassen und Methoden

- Eine Basisklasse kann die Implementation einer Methode für alle abgeleiteten Klassen erzwingen, ohne eine Basis-Methode zur Verfügung zu stellen. Dies geschieht durch das Schlüsselwort **`abstract`**
- Die Basisklasse selber ist dann ebenfalls abstrakt, da nicht alle Methoden implementiert sind.

C #

```
public abstract class BasisKlasse
{
    public abstract void Info(string s); // Muss in allen abgeleiteten Klassen überschrieben werden
    public virtual void Ausgabe(string s)// Kann in den abgeleiteten Klassen überschrieben werden
    { Console.WriteLine(s);}
}
```

Abstrakte Klassen und Methoden

Die abgeleitenen Klassen sind gezwungen, die Info-Methode zu überschreiben.
Andernfalls sind sie ebenfalls abstract

C #

```
public class Klasse1 : BasisKlasse
{
    DateTime date;
    public override void Info(string s)       // muss implementiert werden
    { Console.WriteLine(s + date.Year); }

    public override void Ausgabe(string s)
    { Console.WriteLine($"Ausgabe: {s}"); }   // kann implementiert werden
}
```

Versiegelte Klassen

- Wenn man unbedingt verhindern möchte, dass andere Programmierer von einer entwickelten Komponente weitere Subklassen ableiten, müssen diese Klassen mithilfe des Schlüsselwort **`sealed`** geschützt werden

C #

```
public sealed class GeschuetzteKlasse: BasisKlasse
{…}

Beim Versuch, von einer Subklasse zu erben, schlägt Ihnen der Compiler erbarmungslos
auf die Pfoten

public class Subklasse: GeschuetzteKlasse   // → Fehler
{…}
```

Partielle Klassen

- Das Konzept partieller Klassen ermöglicht es, den Quellcode einer Klasse auf mehrere einzelne Dateien aufzusplitten.
- Alle Teile der Klasse sind mit dem Schlüsselwort **`partial`** zu kennzeichnen.

C #

```
public class Kunde
{
    public Kunde(string name)
    {
        Name = name;
    }
    public string Name { get; protected set; }
    public double Guthaben { get; protected set; }

    public void AddGuthaben(double betrag)
    {
        Guthaben += betrag;
    }
}
```

Partielle Klassen

C #

```
public partial class Kunde
{
    public Kunde(string name)
    {
        Name = name;
    }
}

public partial class Kunde
{
    public string Name { get; protected set; }
    public double Guthaben { get; protected set; }
}

public partial class Kunde
{
    public void AddGuthaben(double betrag)
    {
        Guthaben += betrag;
    }
}
```

Kann wie folgt aufgeteilt werden

Statische Klassen

- Mit dem Schlüsselwort static kann man nicht nur statische Klassenmitglieder (Eingenschaften, Methoden), sondern auch komplette statische Klassen deklarieren
- z.B. für Formelsammlungen

C #

```
public static class Kugel
{
    private static double Pi = Math.PI;

    public static double Durchmesser_Volumen(double durchmesser)
    {
        double vol = Math.Pow(durchmesser,3) * Pi/6.0;
        return vol;
    }
}

Aufruf:
double v = 1.0;
double d = Kugel.Durchmesser_Volumen(v);
```