

The background of the slide is a complex network of thin grey lines connecting various sized circles. Some circles are solid blue, some are solid dark blue, and some are white with a dark blue center. The overall effect is a sense of interconnectedness and digital technology.

VERERBUNG

Vincent Uhlmann
IT-Akademie Dr. Heuer GmbH

VERERBUNG

- Objekte sind in Familien gruppiert
- Beispielsweise gehören Autos, LKWs und Motorräder zur Familie der Fahrzeuge
- Spieler, Trainer und Torwarte sind Teil der Personenfamilie
- Mitarbeiter umfassen Angestellte, Auszubildende und Praktikanten

VERERBUNG

- Familien von Objekten zeigen gemeinsame Merkmale
- Gleichzeitig unterscheiden sie sich in spezifischen Details
- Vererbung ermöglicht es, Eigenschaften, Methoden etc. von einer Objektfamilie auf eine andere zu übertragen
- Dies reduziert redundante Implementierungen und mögliche Fehler

GRUNDLEGENDE BEGRIFFE

- Basisklasse / Superklasse
- Abgeleitete Klasse / Subklasse
- Beispiel: Die Klasse Fahrzeug als Basisklasse, die Klasse Auto als Subklasse

```
// Basisklasse  
public class Fahrzeug  
{  
}
```

```
// Abgeleitete Klasse  
public class Auto : Fahrzeug  
{  
}
```

ATTRIBUTE UND METHODEN

- Die Klasse Fahrzeug enthält Attribute wie Hersteller, Modell und Baujahr
- Ebenso definiert sie Methoden wie Fahren() und Ausgabe()
- Durch Vererbung erbt die Klasse Auto diese Attribute und Methoden
- Konstruktoren werden nicht vererbt
- Statische Konstruktoren werden nicht vererbt
- Finalizer werden nicht vererbt

SICHTBARKEIT BEI VERERBUNG

- Private Attribute und Methoden sind für erbende Klassen nicht zugänglich
- Für Vererbung wird die Sichtbarkeit "protected" verwendet
- Nur auf public oder protected Attribute oder Methoden kann zugegriffen werden
- Mit protected gekennzeichnete Attribute und Methoden sind innerhalb der Vererbungshierarchie sichtbar, aber außerhalb nicht

ATTRIBUTE UND METHODEN

```
// Basisklasse
public class Fahrzeug
{
    public string Modell;
    protected string Hersteller;
    private string _x;
}

// Abgeleitete Klasse
public class Auto : Fahrzeug
{
    // Modell kann verwendet werden
    // Hersteller kann verwendet werden
    // _x kann nicht verwendet werden
}
```

KONSTRUKTOREN UND VERERBUNG

- Eine Subklasse benötigt eine Instanz der Basisklasse
- Um dies zu erreichen, muss der Konstruktor der Basisklasse aufgerufen werden
- Hierzu wird der leere Konstruktor der Basisklasse aufgerufen
- Existiert dieser nicht, muss ein parametrisierter Konstruktor der Basisklasse aufgerufen werden

ZUGRIFF AUF BASISKLASSENMEMBER

- Über "base" kann auf Attribute, Methoden und Konstruktoren der Basisklasse zugegriffen werden
- Funktioniert ähnlich wie this
- Dies ist nur möglich, wenn die Sichtbarkeit in der Basisklasse dies zulässt
- Der Zugriff ist nur auf die direkte Basisklasse möglich
- Mehrfachverschachtelte Aufrufe wie "base.base.Methode()" sind nicht erlaubt

ZUGRIFF AUF BASISKLASSENMEMBER

```
// Basisklasse
public class Fahrzeug
{
    private string _name;

    public Fahrzeug(string name)
    {
        _name = name;
    }
}

// Abgeleitete Klasse
public class Auto : Fahrzeug
{
    public Auto(string name) : base(name)
    {
    }
}
```

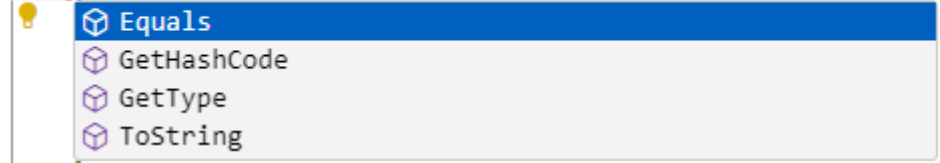
OBJECT

- In .NET erben alle Typen automatisch von Object
- Daher gibt es für jedes Objekt eine feste Standardauswahl an Methoden
- Mehr dazu bei dem Thema Polymorphie

```
public class Car  
{  
}
```

```
Car car = new Car();
```

```
car.
```



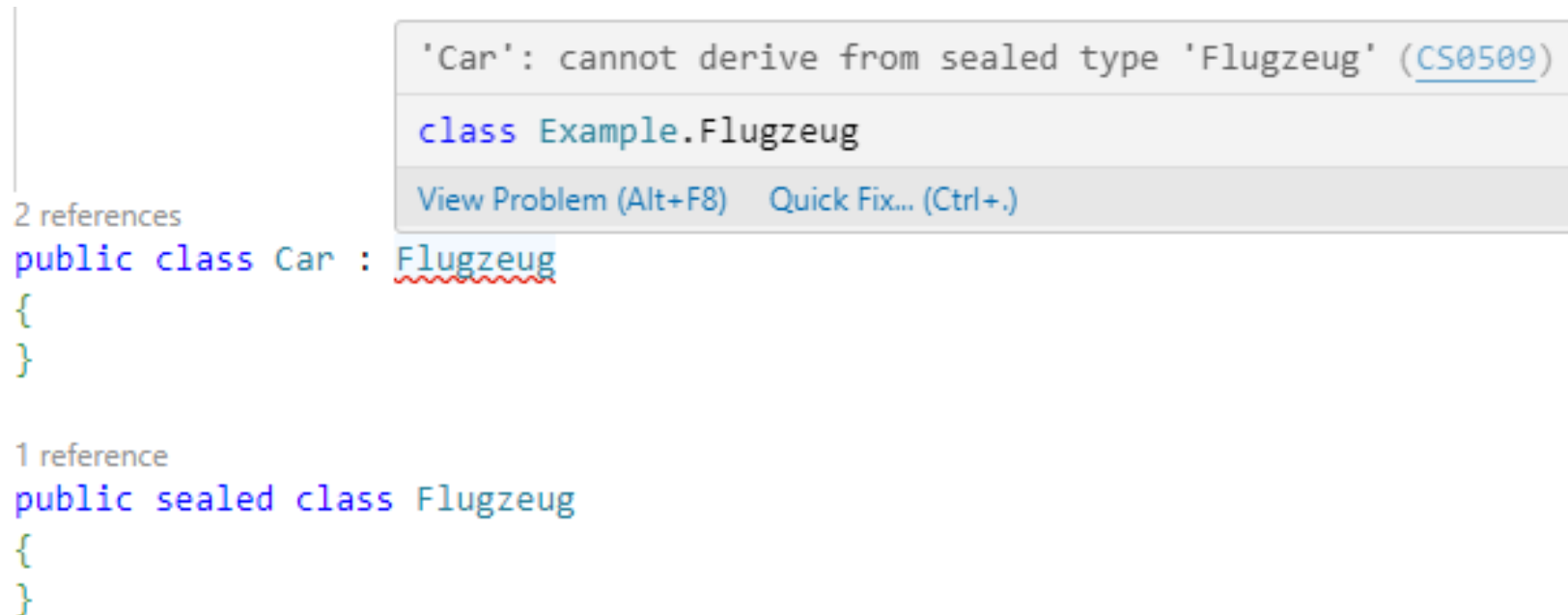
- Equals
- GetHashCode
- GetType
- ToString

Object

Equals(Object)
Equals(Object, Object)
Finalize()
GetHashCode()
GetType()
MemberwiseClone()
ReferenceEquals()
ToString()
#ctor()

SEALED

- Der Modifizierer sealed verhindert, dass andere Klassen von einer Klasse erben können



The screenshot shows an IDE with a compilation error and two code snippets. The error message, displayed in a light gray box, states: "'Car': cannot derive from sealed type 'Flugzeug' (CS0509)". Below the error message, there are two buttons: "View Problem (Alt+F8)" and "Quick Fix... (Ctrl+.)". To the left of the error box, the text "2 references" is visible. Below this, the following code is shown:

```
public class Car : Flugzeug
{
}
```

Below this code, the text "1 reference" is visible. Below this, the following code is shown:

```
public sealed class Flugzeug
{
}
```