

# Namespace ProgrammierenLernen

## Classes

[Fahrzeug](#)

Parent Class to represent all vehicles.

[Flugzeug](#)

[Giraffe](#)

[Linie](#)

[Motorrad](#)

Class to represent motorcycles.

[PKW](#)

[Person](#)

[PolizeiRoboter](#)

[Punkt](#)

[RDreieck](#)

[Roboter](#)

[Roomba](#)

[Utils](#)

[Viereck](#)

[Wolf](#)

## Structs

[Point](#)

## Interfaces

[IFiguren](#)

# Class Fahrzeug

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

Parent Class to represent all vehicles.

```
public abstract class Fahrzeug
```








## Inheritance

[object](#)  ← Fahrzeug

## Derived

[Motorrad](#), [PKW](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#) 

# Constructors

## Fahrzeug(string, string, int, string)

Constructor for a generic vehicle

```
protected Fahrzeug(string marke, string klasse, int baujahr, string motor)
```

## Parameters

marke [string](#) 

klasse [string](#) 

baujahr [int](#) 

motor [string](#) 

# Fields

## aktuelleGeschwindigkeit

`protected int` aktuelleGeschwindigkeit

Field Value

[int](#)

## baujahr

`protected int` baujahr

Field Value

[int](#)

## klasse

`protected string` klasse

Field Value

[string](#)

## marke

`protected string` marke

Field Value

[string](#)

## maxGeschwindigkeit

```
protected int maxGeschwindigkeit
```

Field Value

[int](#)

## motor

```
protected string motor
```

Field Value

[string](#)

## motorLäuft

```
protected bool motorLäuft
```

Field Value

[bool](#)

## Properties

### Baujahr

```
public int Baujahr { get; }
```

Property Value

[int](#)

# Klasse

```
public string Klasse { get; }
```

Property Value

[string](#)

# Marke

```
public string Marke { get; set; }
```

Property Value

[string](#)

# Motor

```
public string Motor { get; set; }
```

Property Value

[string](#)

# Methods

## Bremsen(int)

Method to decrease the speed of the vehicle given a specific increment.

```
public void Bremsen(int inkrement)
```

Parameters

inkrement [int](#)

Dies ist der inkrement, der unsere geschwindigkeit erhoeht.

## Hupen()

Abstract method to implement the sound of a vehicles horn.

```
public abstract void Hupen()
```

## ManageMotor()

Change the current state of the motor.

```
public void ManageMotor()
```

## Parken()

Allows parking only if the motor is off and the speed is 0. Otherwise we turn the motor off or brake until we reach the speed 0.

```
public bool Parken()
```

## Returns

[bool](#)

Boolean: Ob wir geparkt sind.

## SchnellerFahren(int)

Increase the speed of our vehicle given a specific increment.

```
public void SchnellerFahren(int inkrement)
```

## Parameters



# Class Flugzeug

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Flugzeug
```

## Inheritance

[object](#)  ← Flugzeug

## Inherited Members


[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 


## Constructors

Flugzeug(string, string, int, double)

```
public Flugzeug(string hersteller, string modell, int sitzplaetze,  
double maxGeschwindigkeit)
```

## Parameters

hersteller [string](#) 

modell [string](#) 

sitzplaetze [int](#) 

maxGeschwindigkeit [double](#) 

## Properties

### Hersteller

```
public string Hersteller { get; set; }
```



Property Value

[string](#) 

# Class Giraffe

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Giraffe
```

## Inheritance

[object](#)  ← Giraffe

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

# Properties

## Name

```
public string Name { get; }
```

## Property Value

[string](#) 

# Methods

## Essen(string)

```
public void Essen(string futter)
```

## Parameters

futter [string](#) 

# Geraeusch()

```
public void Geraeusch()
```

# Interface IFiguren

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public interface IFiguren
```

## Properties

### Hoehe

```
double Hoehe { get; set; }
```

Property Value

[double](#)

### Laenge

```
double Laenge { get; set; }
```

Property Value

[double](#)

### X

```
int X { get; set; }
```

Property Value

[int](#)

Y

```
int Y { get; set; }
```

Property Value

[int](#)

## Methods

Oberflaeche()

```
double Oberflaeche()
```

Returns

[double](#)

Perimiter()

```
double Perimiter()
```

Returns

[double](#)

# Class Linie

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public class Linie : IFiguren
```

## Inheritance

[object](#) ← Linie

## Implements

[IFiguren](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Properties

## Hoehe

```
public double Hoehe { get; set; }
```

Property Value

[double](#)

## Laenge

```
public double Laenge { get; set; }
```

Property Value

[double](#)

X

```
public int x { get; set; }
```

Property Value

[int](#)

Y

```
public int y { get; set; }
```

Property Value

[int](#)

## Methods

Oberflaeche()

```
public double Oberflaeche()
```

Returns

[double](#)

Perimiter()

```
public double Perimiter()
```

Returns

[double](#)

# Class Motorrad

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll








Class to represent motorcycles.

```
public class Motorrad : Fahrzeug
```

## Inheritance

[object](#)  ← [Fahrzeug](#) ← Motorrad

## Inherited Members

[Fahrzeug.marke](#) , [Fahrzeug.klasse](#) , [Fahrzeug.baujahr](#) , [Fahrzeug.motor](#) , [Fahrzeug.maxGeschwindigkeit](#) , [Fahrzeug.motorLäuft](#) , [Fahrzeug.aktuelleGeschwindigkeit](#) , [Fahrzeug.Motor](#) , [Fahrzeug.Baujahr](#) , [Fahrzeug.Klasse](#) , [Fahrzeug.Marke](#) , [Fahrzeug.Parken\(\)](#) , [Fahrzeug.Bremsen\(int\)](#) , [Fahrzeug.SchnellerFahren\(int\)](#) , [Fahrzeug.ManageMotor\(\)](#) , [object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Constructors

Motorrad(string, string, int, string)

```
public Motorrad(string marke, string klasse, int baujahr, string motor)
```

## Parameters

marke [string](#) 

klasse [string](#) 

baujahr [int](#) 

motor [string](#) 

## Methods



# Hupen()

Abstract method to implement the sound of a vehicles horn.

```
public override void Hupen()
```

# Class PKW

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class PKW : Fahrzeug
```

## Inheritance

[object](#)  ← [Fahrzeug](#) ← PKW

## Inherited Members

[Fahrzeug.marke](#) , [Fahrzeug.klasse](#) , [Fahrzeug.baujahr](#) , [Fahrzeug.motor](#) , [Fahrzeug.maxGeschwindigkeit](#) ,  
[Fahrzeug.motorLäuft](#) , [Fahrzeug.aktuelleGeschwindigkeit](#) , [Fahrzeug.Motor](#) , [Fahrzeug.Baujahr](#) ,  
[Fahrzeug.Klasse](#) , [Fahrzeug.Marke](#) , [Fahrzeug.Parken\(\)](#) , [Fahrzeug.Bremsen\(int\)](#) ,  
[Fahrzeug.SchnellerFahren\(int\)](#) , [Fahrzeug.ManageMotor\(\)](#) , [object.Equals\(object\)](#)  ,  
[object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Constructors

PKW(string, string, int, string)

```
public PKW(string marke, string klasse, int baujahr, string motor)
```

## Parameters

marke [string](#) 

klasse [string](#) 

baujahr [int](#) 

motor [string](#) 

## Methods

# Hupen()

Abstract method to implement the sound of a vehicles horn.

```
public override void Hupen()
```

# Class Person

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Person
```

## Inheritance

[object](#)  ← Person

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 


# Constructors

Person(string, string, bool)

```
public Person(string name, string adresse, bool krimineller)
```

## Parameters

name [string](#) 

adresse [string](#) 

krimineller [bool](#) 

# Fields

adresse

```
public string adresse
```

## Field Value

[string](#)

krimineller

```
public bool krimineller
```

Field Value

[bool](#)

name

```
public string name
```

Field Value

[string](#)

# Struct Point

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public struct Point
```

## Inherited Members

[ValueType.Equals\(object\)](#)<sup>↗</sup> , [ValueType.GetHashCode\(\)](#)<sup>↗</sup> , [ValueType.ToString\(\)](#)<sup>↗</sup> ,  
[object.Equals\(object, object\)](#)<sup>↗</sup> , [object.GetType\(\)](#)<sup>↗</sup> , [object.ReferenceEquals\(object, object\)](#)<sup>↗</sup>

## Constructors

### Point(int, int)

```
public Point(int x, int y)
```

## Parameters

x [int](#)<sup>↗</sup>

y [int](#)<sup>↗</sup>

## Fields

X

```
public int x
```

## Field Value

[int](#)<sup>↗</sup>

y

```
public int y
```

Field Value

[int](#)

## Methods

Display()

```
public void Display()
```

# Class PolizeiRoboter

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class PolizeiRoboter : Roboter
```

## Inheritance

[object](#)  ← [Roboter](#) ← PolizeiRoboter

## Inherited Members

[Roboter.hersteller](#) , [Roboter.modell](#) , [Roboter.id](#) , [Roboter.Start\(\)](#) , [Roboter.Stop\(\)](#) ,  
[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Constructors

### PolizeiRoboter(string)

```
public PolizeiRoboter(string einsatzAdresse)
```

## Parameters

einsatzAdresse [string](#) 

## Methods

### GerauescheMachen()

```
public override void GerauescheMachen()
```

### KriminellenEntfernen(Person)



```
public void KriminellenEntfernen(Person person)
```

Parameters

person [Person](#)

## KriminellenHinzufuegen(Person)

```
public void KriminellenHinzufuegen(Person person)
```

Parameters

person [Person](#)

## PersonVergleichen(Person)

```
public void PersonVergleichen(Person person)
```

Parameters

person [Person](#)

# Class Punkt

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public class Punkt : IFiguren
```








## Inheritance

[object](#)  ← Punkt

## Implements

[IFiguren](#)

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

# Properties

## Hoehe

```
public double Hoehe { get; set; }
```

## Property Value

[double](#) 

## Laenge

```
public double Laenge { get; set; }
```

## Property Value

[double](#) 

# X

```
public int x { get; set; }
```

Property Value

[int](#)

# Y

```
public int y { get; set; }
```

Property Value

[int](#)

## Methods

### Oberflaeche()

```
public double Oberflaeche()
```

Returns

[double](#)

### Perimiter()

```
public double Perimiter()
```

Returns

[double](#)

# Class RDreieck

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public class RDreieck : IFiguren
```

## Inheritance

[object](#) ← RDreieck

## Implements

[IFiguren](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Properties

## Hoehe

```
public double Hoehe { get; set; }
```

Property Value

[double](#)

## Laenge

```
public double Laenge { get; set; }
```

Property Value

[double](#)

# X

```
public int x { get; set; }
```

Property Value

[int](#)

# Y

```
public int y { get; set; }
```

Property Value

[int](#)

## Methods

### Oberflaeche()

```
public double Oberflaeche()
```

Returns

[double](#)

### Perimiter()

```
public double Perimiter()
```

Returns

[double](#)

# Class Roboter

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public abstract class Roboter
```








## Inheritance

[object](#)  ← Roboter

## Derived

[PolizeiRoboter](#), [Roomba](#)

## Inherited Members


[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 


# Constructors

Roboter(string, string)

```
public Roboter(string hersteller, string modell)
```

## Parameters

hersteller [string](#) 

modell [string](#) 

# Fields

hersteller

```
protected string hersteller
```

## Field Value

[string](#)

id

```
protected string id
```

Field Value

[string](#)

modell

```
protected string modell
```

Field Value

[string](#)

## Methods

GerauescheMachen()

```
public abstract void GerauescheMachen()
```

Start()

```
public void Start()
```

Stop()

```
public void Stop()
```





# Class Roomba

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Roomba : Roboter
```

## Inheritance

[object](#)  ← [Roboter](#) ← Roomba

## Inherited Members


[Roboter.hersteller](#) , [Roboter.modell](#) , [Roboter.id](#) , [Roboter.Start\(\)](#) , [Roboter.Stop\(\)](#) ,  
[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

## Constructors

### Roomba(string)

```
public Roomba(string modell)
```

## Parameters

**modell** [string](#) 

## Methods

### GerauescheMachen()

```
public override void GerauescheMachen()
```

## Exceptions

[IOException](#) 

Ein Fehler ist aufgetreten beim auslesen.

## MappingErstellen(string)

```
public void MappingErstellen(string zimmer)
```

Parameters

zimmer [string](#)

## MappingsAusgeben()

```
public void MappingsAusgeben()
```

## MuellEntsorgen()

```
public bool MuellEntsorgen()
```

Returns

[bool](#)

## Putzen(int)

```
public bool Putzen(int indexVonZimmer)
```

Parameters

indexVonZimmer [int](#)

Returns

[bool](#)

# Class Utils

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Utils
```

## Inheritance

[object](#)  ← Utils

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

# Methods

## BerechneSchaltjahr(int)

```
public bool BerechneSchaltjahr(int jahr)
```

## Parameters

**jahr** [int](#) 

## Returns

[bool](#) 

## CalculateSumme(int)

```
public static void CalculateSumme(int target)
```

## Parameters

**target** [int](#) 

## Collatz(int, int)

```
public static (int, int) Collatz(int n, int schritte = 0)
```

### Parameters

n [int](#)

schritte [int](#)

### Returns

([int](#), [int](#))

## DateiName(string)

```
public static string DateiName(string pfad)
```

### Parameters

pfad [string](#)

### Returns

[string](#)

## DateiTyp(string)

```
public static string DateiTyp(string pfad)
```

### Parameters

pfad [string](#)

### Returns

[string](#)

## EingabePruefen()

```
public static void EingabePruefen()
```

## EinkaufsListeGenerator()

```
public static List<string> EinkaufsListeGenerator()
```

Returns

[List](#) <[string](#)>

## EinsDichMich()

```
public static void EinsDichMich()
```

## Faktoriell(BigInteger)

```
public static BigInteger Faktoriell(BigInteger zahl)
```

Parameters

zahl [BigInteger](#)

Returns

[BigInteger](#)

## FindIndex(double[], double)

```
public static int FindIndex(double[] zahlen, double gesuchteZahl)
```

## Parameters

zahlen [double](#)[]

gesuchteZahl [double](#)

## Returns

[int](#)

## GenerateRandomWord(int)

```
public string GenerateRandomWord(int length)
```

## Parameters

length [int](#)

## Returns

[string](#)

## JoinArray(int[], string)

```
public static string JoinArray(int[] zeichenFolgen, string seperator = " ")
```

## Parameters

zeichenFolgen [int](#)[]

seperator [string](#)

## Returns

[string](#)

## JoinArray(string[], string)

```
public static string JoinArray(string[] zeichenFolgen, string seperator = " ")
```

### Parameters

zeichenFolgen [string](#)[]

seperator [string](#)

### Returns

[string](#)

## ListStatistics(List<double>)

```
public static (double Durchschnitt, double Summe, double Min, double Max, int Count)  
ListStatistics(List<double> zahlen)
```

### Parameters

zahlen [List](#)<[double](#)>

### Returns

([double](#) Durchschnitt, [double](#) Summe, [double](#) Min, [double](#) Max, [int](#) Count)

## MinDictionary(Dictionary<string, decimal>)

```
public static decimal MinDictionary(Dictionary<string, decimal> produkte)
```

### Parameters

produkte [Dictionary](#)<[string](#), [decimal](#)>

Returns

[decimal](#) 

## NANP(string)

```
public static void NANP(string telephoneNumber)
```

Parameters

telephoneNumber [string](#) 

## NcR(BigInteger, BigInteger)

Calculates n choose r.

```
public static BigInteger NcR(BigInteger n, BigInteger r)
```

Parameters

n [BigInteger](#) 

r [BigInteger](#) 

Returns

[BigInteger](#) 

## OpenWith(string)

```
public static void OpenWith(string pfaad)
```

Parameters

pfaad [string](#) 



## Pangram()

```
public static Tuple<List<string>, List<string>> Pangram()
```

Returns

[Tuple](#) <[List](#) <[string](#)>, [List](#) <[string](#)>>

## Pangram(string)

Ein gegebener Satz wird aufgeteilt in einzelne Woerter. Danach werden diese Woerter umgedreht mit unsere Reverse() methode. Dann geben wir wieder alle Woerter in einen String zurueck ohne die Reihenfolge zu veraendern.

```
public static string Pangram(string eingabe)
```

Parameters

eingabe [string](#)

Unser Eingabe Satz

Returns

[string](#)

Eingabe Soll umgedreht werden.

## Pangram(string, string)

```
public static string Pangram(string input, string sep = " ")
```

Parameters

input [string](#)

sep [string](#)

Returns

[string](#)

## PascalTriangle(int)

Creates a pyramid representing the values in Pascals Triangle.

```
public static void PascalTriangle(int hoehe)
```

Parameters

hoehe [int](#)

## PinUeberpruefung(string)

```
public static bool PinUeberpruefung(string echtenPin)
```

Parameters

echtenPin [string](#)

Returns

[bool](#)

## PlanetenAlter()

```
public static void PlanetenAlter()
```

## PrintArray(int[])

```
public static void PrintArray(int[] array)
```

Parameters

array [int](#)[]

## PrintArray(string[])

```
public static void PrintArray(string[] array)
```

Parameters

array [string](#)[]

## PyramideErstellen(int)

```
public static void PyramideErstellen(int n)
```

Parameters

n [int](#)

## RandomChars(int, bool)

```
public static string RandomChars(int length, bool includeUpperCase = false)
```

Parameters

length [int](#)

includeUpperCase [bool](#)

Returns

[string](#)

## RechnungErstellenOhneGesamtBetrag(Dictionary<string, decimal>, Dictionary<string, int>)

```
public static string RechnungErstellenOhneGesamtBetrag(Dictionary<string, decimal> produkte,  
Dictionary<string, int> warenkorb)
```

### Parameters

produkte [Dictionary](#) <[string](#), [decimal](#)>

warenkorb [Dictionary](#) <[string](#), [int](#)>

### Returns

[string](#)

## RegentropfenFaktoren(int)

```
public static void RegentropfenFaktoren(int obereGrenze)
```

### Parameters

obereGrenze [int](#)

## RepeatInput()

```
public static void RepeatInput()
```

## ReverseString(string)

```
public static string ReverseString(string wort)
```

### Parameters

wort [string](#)

Returns

[string](#)

## RnaUebersetzung(string)

```
public static List<string> RnaUebersetzung(string rnaSequenz)
```

Parameters

*rnaSequenz* [string](#)

Returns

[List](#) <[string](#)>

## StudentAusgabe()

Erzeugt eine schoene ausgabe anhand von die StudentenInfos Tupeln.

```
public static void StudentAusgabe()
```

## Summe(decimal[])

```
public static decimal Summe(decimal[] zahlen)
```

Parameters

*zahlen* [decimal](#)[]

Returns

[decimal](#)

## Summe(int, int)

```
public static int Summe(int a, int b)
```

### Parameters

**a** [int](#)

**b** [int](#)

### Returns

[int](#)

## Summe(int[])

```
public static int Summe(int[] zahlen)
```

### Parameters

**zahlen** [int](#)[]

### Returns

[int](#)

## TechEinkaufen(decimal)

```
public static void TechEinkaufen(decimal budget)
```

### Parameters

**budget** [decimal](#)

# Class Viereck

Namespace: [ProgrammierenLernen](#)

Assembly: ProgrammierenLernen.dll

```
public class Viereck : IFiguren
```

## Inheritance

[object](#) ← Viereck

## Implements

[IFiguren](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

# Properties

## Hoehe

```
public double Hoehe { get; set; }
```

Property Value

[double](#)

## Laenge

```
public double Laenge { get; set; }
```

Property Value

[double](#)

# X

```
public int x { get; set; }
```

Property Value

[int](#)

# Y

```
public int y { get; set; }
```

Property Value

[int](#)

## Methods

### Oberflaeche()

```
public double Oberflaeche()
```

Returns

[double](#)

### Perimiter()

```
public double Perimiter()
```

Returns

[double](#)



# Class Wolf

Namespace: [ProgrammierenLernen](#)








Assembly: ProgrammierenLernen.dll

```
public class Wolf
```

## Inheritance

[object](#)  ← Wolf

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  , [object.GetType\(\)](#)  ,  
[object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  , [object.ToString\(\)](#) 

# Properties

## Name

```
public string Name { get; }
```

## Property Value

[string](#) 

# Methods

## Essen(string)

```
public void Essen(string futter)
```

## Parameters

futter [string](#) 

# Geraeusch()

```
public void Geraeusch()
```