Tristan Izlar

COP2362

TUTORIAL 3-2: Chapter 12 - Working with Inheritance

I worked alone.

Text

Description automatically generated with low confidence

Program.cs

using System;

namespace Vehicles

{

class Program

{

static void doWork()

{

Console.WriteLine("Journey by airplane:");

Airplane myPlane = new Airplane();

myPlane.StartEngine("Contact");

myPlane.TakeOff();

myPlane.Drive();

myPlane.StopEngine("Whirr");

Console.WriteLine("\nJourney by car:");

Car myCar = new Car();

myCar.StartEngine("Brm brm");

myCar.Accelerate();

myCar.Drive();

myCar.Brake();

myCar.StopEngine("Phut phut");

Console.WriteLine("\nTesting polymorphism");

Vehicle v = myCar;

v.Drive();

v = myPlane;

v.Drive();

}

static void Main()

{

doWork();

}

}

}

Vehicle.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Vehicles

{

class Vehicle

{

public void StartEngine(string noiseToMakeWhenStarting)

{

Console.WriteLine($"Starting engine: {noiseToMakeWhenStarting}");

}

public void StopEngine(string noiseToMakeWhenStopping)

{

Console.WriteLine($"Stopping engine: {noiseToMakeWhenStopping}");

}

public virtual void Drive()

{

Console.WriteLine("Default implementation of the Drive method");

}

}

}

Airplane.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Vehicles

{

class Airplane : Vehicle

{

public void TakeOff()

{

Console.WriteLine("Taking off");

}

public void Land()

{

Console.WriteLine("Landing");

}

public override void Drive()

{

Console.WriteLine("Flying");

}

}

}

Car.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Vehicles

{

class Car:Vehicle

{

public void Accelerate()

{

Console.WriteLine("Accelerating");

}

public void Brake()

{

Console.WriteLine("Braking");

}

public override void Drive()

{

Console.WriteLine("Motoring");

}

}

}

A screenshot of a computer

Description automatically generated with medium confidence

Program.cs

using System;

using Extensions;

namespace ExtensionMethod

{

class Program

{

static void doWork()

{

int x = 591;

for(int i = 2; i <= 10; i++)

{

Console.WriteLine($"{x} in base {i} is {x.ConvertToBase(i)}");

}

}

static void Main()

{

doWork();

}

}

}

Util.cs

using System;

using Extensions;

namespace Extensions

{

static class Util

{

public static int ConvertToBase(this int i, int baseToConvertTo)

{

if (baseToConvertTo < 2 || baseToConvertTo > 10)

{

throw new ArgumentException("Value cannot be converted to base " + baseToConvertTo.ToString());

}

int result = 0;

int iterations = 0;

do

{

int nextDigit = i % baseToConvertTo;

i /= baseToConvertTo;

result += nextDigit \* (int)Math.Pow(10, iterations);

iterations++;

}

while (i != 0);

return result;

}

}

}