



Национална програма
"Обучение за ИТ умения и кариера"
<https://it-kariera.mon.bg>

Министерството на
образованието и науката
<https://www.mon.bg>



Интерфейси

Interfaces



Интерфейсы

Интерфейс

Компилятора вътрешно добавя

Ключова дума

Модификатор
за достъп

```
public interface IPrintable {  
    void Print();  
}
```

Име

компилятор

```
public interface IPrintable {  
    public abstract void Print();  
}
```

Добавя public за
всички членове

Пример за интерфейс

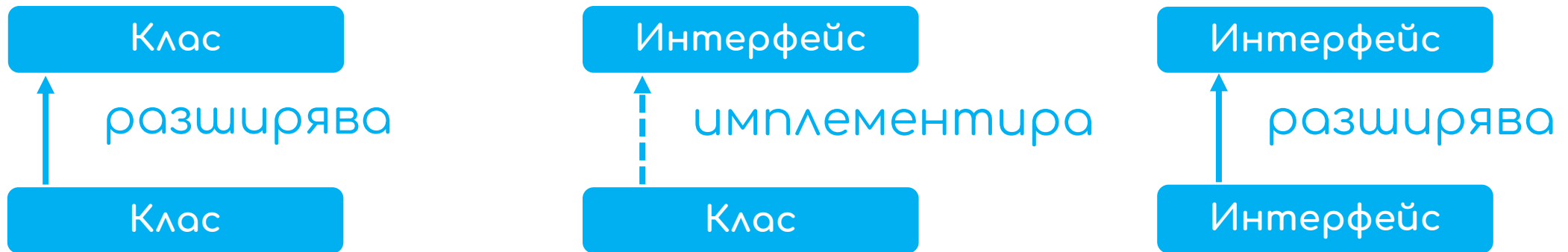
Имплементацията на Print() се задава в класа Document

```
public interface IPrintable
{
    void Print();
}
```

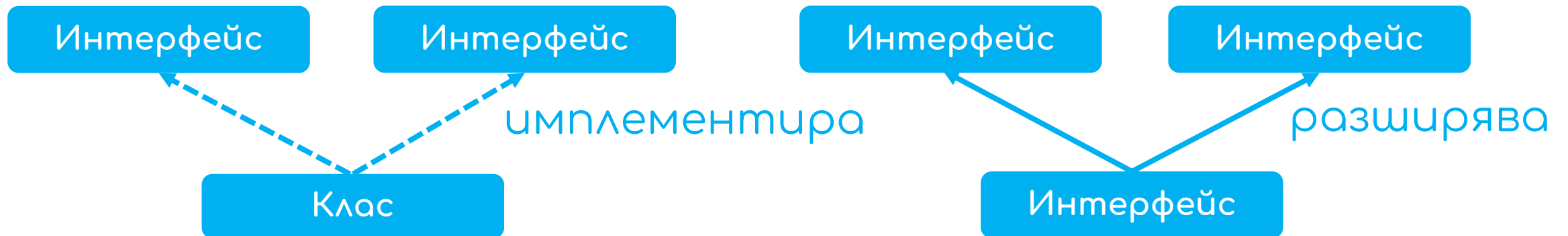
```
class Document : IPrintable
{
    public void Print()
    { Console.ReadLine("Hello"); }
}
```

Множествено наследяване

- Връзка между класове и интерфейси

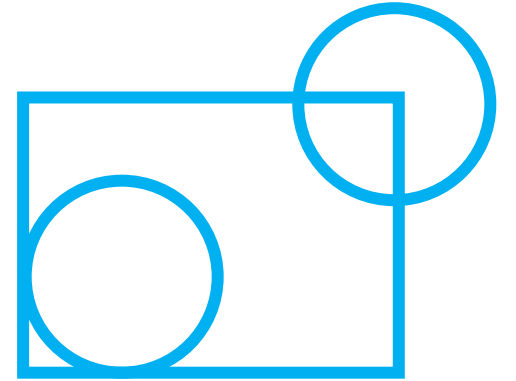


- Множествено наследяване



Задача: Фигури

- Създайте проект, който съдържа интерфейс за рисуваеми обекти
- **Имплементирайте** два типа фигури: **кръг** и **правоъгълник**
- И двата класа трябва да отпечатват на конзолата фигурата си със "*".



```
<<interface>>  
Drawable
```

```
+Draw()
```

```
<<Drawable>>  
Circle
```

```
+Radius: int
```

```
<<Drawable>>  
Rectangle
```

```
-Width: int  
-Height: int
```

Решение: Фигури

```
public interface IDrawable {  
    void Draw();  
}
```

```
public class Rectangle : IDrawable {  
    // TODO: добавете полета и конструктор  
    public void Draw() { ...
```

```
public class Circle : IDrawable {  
    // TODO: добавете полета и конструктор  
    public void Draw() { ...
```

Решение: Фигури – Чертане на правоъгълник

```
public void Draw()
{
    DrawLine(this.Width, '*', '*');
    for (int i = 1; i < this.Height - 1; ++i)
        DrawLine(this.Width, '*', ' ');
    DrawLine(this.Width, '*', '*');
}
private void DrawLine(int width, char end, char mid)
{
    Console.Write(end);
    for (int i = 1; i < width - 1; ++i)
        Console.Write(mid);
    Console.WriteLine(end);
}
```


Решение: Фигури – чертане на кръг

```
double r_in = this.Radius - 0.4;
double r_out = this.Radius + 0.4;
for (double y = this.Radius; y >= -this.Radius; --y)
{
    for (double x = -this.Radius; x < r_out; x += 0.5)
    {
        double value = x * x + y * y;
        if (value >= r_in * r_in && value <= r_out * r_out)
            Console.Write("*");
        else
            Console.Write(" ");
        Console.WriteLine();
    }
}
```

Интерфейс с/у абстрактен клас [1/3]

Абстрактен клас

- Класът може да наследява **само един** абстрактен клас.
- Абстрактните класове могат да **предоставят целия код** и/или само детайлите, които трябва да се презапишат.

Интерфейс

- Класът може да **имплементира няколко интерфейса**.
- Интерфейсът **не може да предоставя никакъв код**, предоставя само описание.

Интерфейс с/у абстрактен клас [2/3]

Абстрактен клас

- Абстрактния клас **може да съдържа модификатори за достъп**
- Ако множество имплементации са от сходен вид и **имат общо поведение или статут**, то абстрактния клас е по-добър избор.

Интерфейс

- Интерфейсите нямат **модификатори за достъп**. Всичко е публично по подразбиране.
- Ако множество имплементации споделят само **сигнатурата на методите и нищо друго**, то тогава интерфейсът е по-добър избор.

Интерфейс с/у абстрактен клас [3/3]

Абстрактен клас

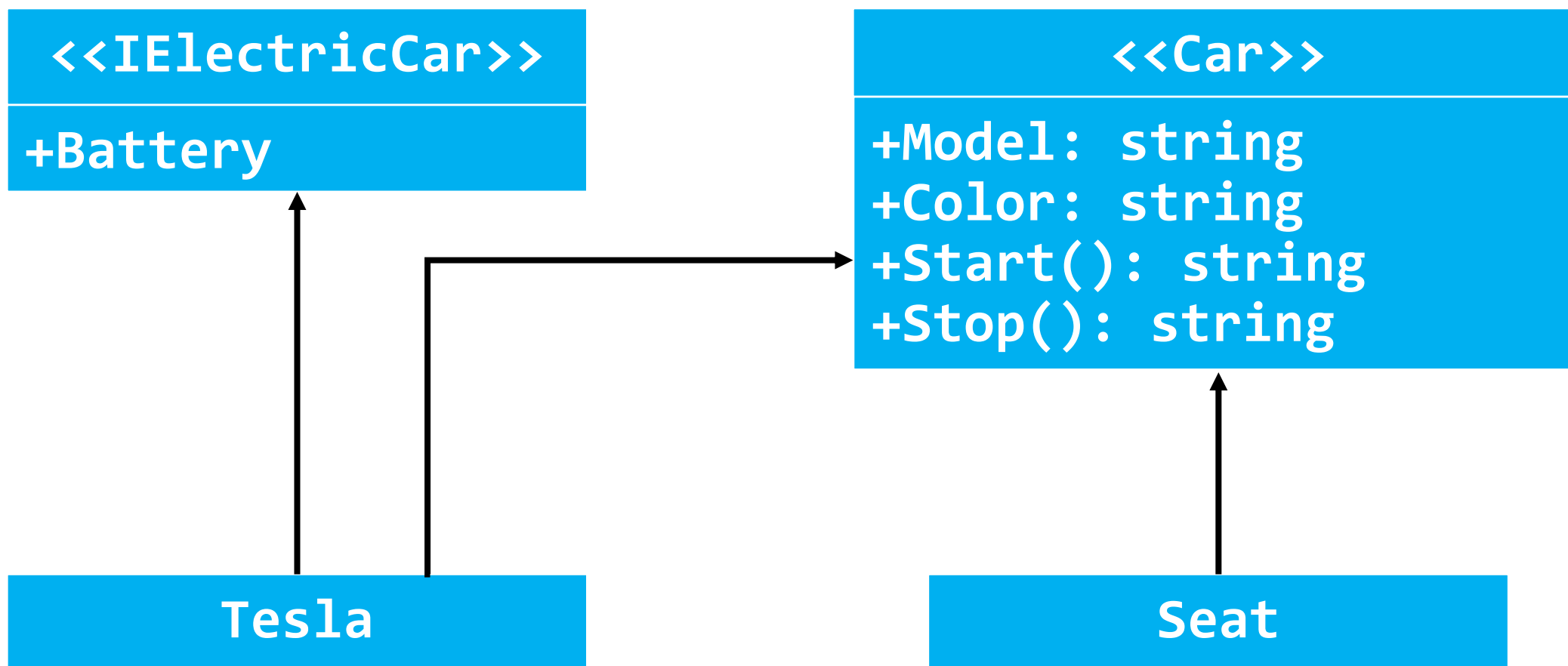
- Абстрактният клас **може да притежава полета и константи**
- Ако добавим **нов метод** към абстрактен клас, то имаме опцията да създадем **имплементация по подразбиране** и така съществуващият код ще може да работи коректно.

Интерфейс

- Не **поддържа полета**
- Ако добавим **нов метод** към интерфейс, то трябва да **проследим всичките** му имплементации и да **дефинираме имплементация** за новия метод.

Задача: Коли

Постройте йерархия от интерфейси и класове



Решение: Колу [1/2]

```
public interface ICar
{
    string Model { get; }
    string Color { get; }
    string Start();
    string Stop();
}
```

```
public interface IElectricCar
{
    int Batteries { get; }
}
```

Решение: Колу [2/2]

```
public class Tesla : ICar, IElectricCar
{
    public string Model { get; private set; }
    public string Color { get; private set; }
    public int Batteries { get; private set; }
    public Tesla(string model, string color, int batteries)
    { // TODO: Add Logic here }
    public string Start()
    { // TODO: Add Logic here }
    public string Stop()
    { // TODO: Add Logic here }
}
```

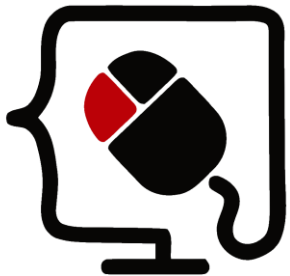
Решение: Колу [3/2]

```
public class Seat : ICar
{
    public string Model { get; private set; }
    public string Color { get; private set; }
    public Tesla(string model, string color)
    { //TODO: Add Logic here }
    public string Start()
    { //TODO: Add Logic here }
    public string Stop()
    { //TODO: Add Logic here }
}
```




Интерфейси

Лаб



Национална програма
"Обучение за ИТ умения и кариера"
<https://it-kariera.mon.bg>

Министерството на
образованието и науката
<https://www.mon.bg>



Документът е разработен за нуждите на Национална програма "Обучение за ИТ умения и кариера" на Министерството на образованието и науката (МОН) и се разпространява под свободен лиценз CC-BY-NC-SA (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).