Вау! ИИ готовит к ЕГЭ по информатике **Попробовать**

Урок Спринт 1.9

Объектно-ориентированное программирование и Go

Сегодня мы расскажем вам, что же такое объектно-ориентированное программирование (ООП), чем оно полезно и при чём тут кошачье мяуканье.

Когда мы пользуемся методом ООП, мы создаём объекты, которые взаимодействуют друг с другом, чтобы выполнять поставленные задачи.

В реальной жизни мы тоже используем объекты — например, телефон, тостер или автомобиль. У каждого из них есть набор свойств (например, форма, размер, фирма-производительница) и методов (например, звонить, поджаривать хлеб, ехать). В ООП мы можем создать классы, которые описывают свойства и методы, и объекты на основе этих классов.

Например, мы укажем для класса «автомобиль» свойства: «марка», «модель», «цвет», — и методы: «ехать», «тормозить», «разворачиваться». Если мы создадим объект «автомобиль» на основе этого класса, мы выберем для него марку, модель и цвет, а затем используем его методы для езды, торможения и разворота.

В языке Go можно программировать в ООП стиле, но нет классов, поэтому на сегодняшнем уроке мы будем приводить примеры из разных языков программирования.

Начнём с определения слова «класс»:

Класс — это шаблон, у которого есть поля и методы. Мы можем создавать конкретные переменные-экземпляры (объекты) этого класса. В них будет содержаться информация их класса. Также они могут выполнять команды с помощью методов.

Хм-м-м... Ничего не напоминает?

Точно! Классы в ООП похожи на структуры в Go. Здесь мы можем описать её тип, добавить к ней методы и создать конкретную переменную с типом этой структуры. Тем не менее, структуры и классы всё же не одно и то же. Об их различиях мы поговорим позже.

Чтобы создать класс на языке C# (произносится как «Си Шарп», а не «Це Решётка»), нам нужно ключевое слово `class`:

```
// Создаём класс Cat
public class Cat
{
        // Поля (переменные) обозначается таким образом
        // Сначала идёт модификатор доступа (public), потом тип (string) и название (Name)
    public string Name

        // А вот эта штука называется «конструктор». Она нужна, чтобы давать значения полям, когда объявляется объект
    public Cat(string name)
    {
        Name = name;
    }

        // Метод Meow() — то есть «мяукать»
        // void — это тип возвращаемого значения
    public void Meow()
    {
                // А вот так в C# работает вывод в консоль
        Console.WriteLine($"{Name} мяукнул(а)!");
    }
}

// Класс Program — основной класс любой программы на языке C#
class Program
{
        // Как и у нас, в C# есть метод main
```

```
    static void Main()
    {
                // Создаём объект класса Cat с именем «Барсик»
                // Это эквивалент нашего var myCat Cat = Cat{name:"Барсик"}
        Cat myCat = new Cat("Барсик");
                // Вызываем метод мяуканья
        myCat.Meow();
    }
}
```

А теперь напишем то же самое на Go:

```
package main

import "fmt"

type Cat struct {
    Name string
}

// Конструктор
func NewCat(name string) Cat {
    return Cat{Name: name}
}

func (c Cat) Meow() {
    fmt.Printf("%s мяукнул(а)!\n", c.Name)
}

func main() {
    myCat := NewCat("Барсик")
    myCat.Meow()
}
```

Мы уверены, что вы хорошо знаете Go и вам не нужно объяснять, что тут происходит 🙂. А теперь вернёмся к страшным определениям и ужасным примерам из мира ООП.

Справка

Исключительное право на учебную программу и все сопутствующие ей учебные материалы, доступные в рамках сервиса, принадлежат АНО ДПО «Образовательные технологии Яндекса». Воспроизведение, копирование, распространение и иное использование программы и материалов допустимо только с предварительного письменного согласия АНО ДПО «Образовательные технологии Яндекса».
Пользовательское соглашение.

© 2018 – 2024 ООО «Яндекс»