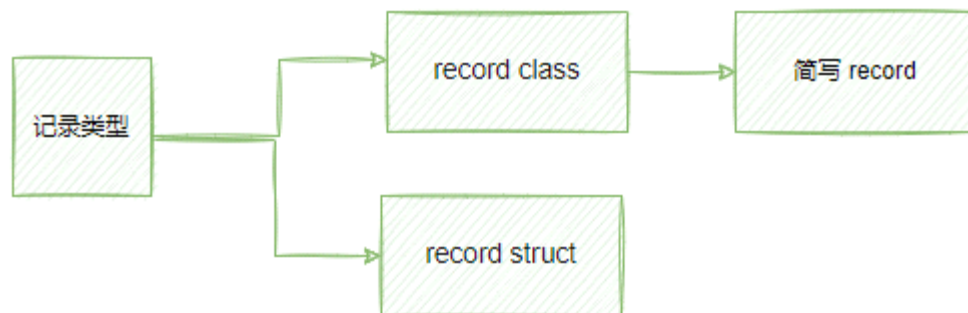


# 第09期-细说记录类型的用法

2022年3月23日 9:56

## 类型说明

记录类型是 C# 9.0 开始引入的新类型，使用 `record` 关键字定义，默认定义的记录类型是引用类型，在 C# 10.0 中，引入的值类型的记录类型使用 `record struct` 定义。



## 定义方式

### 引用类型的记录

```
public record Person
{
    public string FirstName { get; init; } = default!;
    public string LastName { get; init; } = default!;
};

public record Person(string FirstName, string LastName);
```

### 值类型的记录

```
public record struct Point
{
    public double X { get; init; }
    public double Y { get; init; }
    public double Z { get; init; }
}

public readonly record struct Point(double X, double Y, double Z);
```

## 记录类型的序列化

```
public record Person([property: JsonPropertyName("firstName")]string
```

```
FirstName, [property: JsonPropertyName("LastName")]string LastName);
```

## 不可变性

init-only 属性无论是通过位置参数（record class 和 readonly record struct）创建的，还是通过指定 init 访问器创建的，都具有浅的不可变性。

## 值相等性

- 对于 class 类型，如果两个对象引用内存中的同一对象，则这两个对象相等。
- 对于 struct 类型，如果两个对象是相同的类型并且存储相同的值，则这两个对象相等。
- 对于 record 类型（包括 record struct 和 readonly record struct），如果两个对象是相同的类型并且存储相同的值，则这两个对象相等。

重写 Object.Equals(Object) 和重载 operator ==

```
public record Person(string FirstName, string LastName, string[]
PhoneNumbers);

public static void Main()
{
    var phoneNumbers = new string[2];
    Person person1 = new("Nancy", "Davolio", phoneNumbers);
    Person person2 = new("Nancy", "Davolio", phoneNumbers);
    Console.WriteLine(person1 == person2); // output: True

    person1.PhoneNumbers[0] = "555-1234";
    Console.WriteLine(person1 == person2); // output: True

    Console.WriteLine(ReferenceEquals(person1, person2)); // output: False
}
```

## 非破坏性拷贝

```
Person person2 = person1 with { FirstName = "John" };
```

## 内置显示格式设置

*<record type name> { <property name> = <value>, <property name> = <value>, ... }*

内部重写 ToString 方法，ToString 方法内部又调用 PrintMembers 方法打印成员。

```
protected virtual bool PrintMembers(StringBuilder stringBuilder)
{
```

```
    return true;
}
```

## 记录类型继承

记录不能从类继承，类也不能从记录继承。

```
public abstract record Person(string FirstName, string LastName);
public record Teacher(string FirstName, string LastName, int Grade)
    : Person(FirstName, LastName);
```

## 元组析构器

```
Person teacher = new Teacher("Nancy", "Davolio", 3);
var(firstName, lastName) = teacher; // Doesn't deconstruct Grade
```