Bài 6.7: Tìm hiểu kiểu record

- ✓ Khái niệm và đặc điểm
- ✓ Cú pháp khai báo record
- ✓ Khi nào sử dụng struct, class, record?
- ✓ Sử dụng keyword with
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Khái niệm và đặc điểm

- ✓Record là một kiểu tham chiếu hướng dữ liệu. So với class-hướng hành động.
- ✓Được hỗ trợ từ phiên bản C# 9.
- ✓C# 10 bổ sung sự kết hợp của record struct và record class(tùy chọn) để làm rõ kiểu đang chỉ định là kiểu tham chiếu(mặc định) hay kiểu giá trị.
- ✓Record phân biệt với class ở chỗ, record sử dụng giá trị của thuộc tính làm nền tảng so sánh sự tương đương của hai đối tượng.
- ✓Hai biến record tương đương khi nó cùng kiểu và giá trị trong các trường, thuộc tính tương ứng của chúng là giống hệt nhau.
- ✓Trong khi đó class sử dụng tham chiếu đến địa chỉ bộ nhớ heap trong RAM để so sánh sự tương đương của 2 đối tượng.
- ✓Hai biến của class gọi là tương đương nếu nó cùng kiểu và cùng tham chiếu tới 1 đối tượng.

Cú pháp khai báo record

- ✓Ta có 2 cách để tạo một record: sử dụng các tham số theo thứ tự hoặc cú pháp khai báo thuộc tính.
- ✓Ví dụ sau sử dụng 2 cách trên để tạo record:

```
// định nghĩa record bởi tham số có thứ tự
2 references
public record Person(string FirstName, string LastName);

// định nghĩa record bởi khai báo thuộc tính chỉ khởi tạo
2 references
public record Student
{
    2 references
    public string Id { get; init; } = default!;
    1 reference
    public string FullName { get; init; } = default!;
    1 reference
    public int Age { get; init; } = default!;
}
```

Kết quả ví dụ

```
0 references
static void Main()
{
    // tạo đối tượng của record Person
    Person person = new("Huong", "Nguyen");
    Console.WriteLine(person);
    // kết quả: Person { FirstName = Huong, LastName = Nguyen }

    // tạo đối tượng của record Student
    Student student = new() {
        Id = "ST1001",
        FullName = "Le Hong Phong",
        Age = 20
    };
    Console.WriteLine(student);
    // kết quả: Student { Id = ST1001, FullName = Le Hong Phong, Age = 20 }
    // thử thay đổi giá trị của FirstName
    person.FirstName = "Hoang Trong Tuan"; // error!
    // thử thay đổi giá trị của mã sinh viên
    student.Id = "ST1005"; // same error
}
```

Đặc điểm của record

- ✓Để tạo record theo kiểu tham chiếu ta sử dụng record class hay viết gọn là record.
- ✓Để tạo record theo kiểu giá trị(value type), ta sử dụng record struct.
- ✓Khi tạo record, các record sẽ được so sánh tương đương dựa trên cơ sở giá trị các thuộc tính của nó.
- ✓Để làm được điều đó, trình biên dịch sẽ sinh tự động các phương thức sau cho kiểu record(cả record struct và record class):
  - ✓ Một phương thức ghi đè của Object.Equals(Object).
  - ✓ Một phương thức virtual Equals trong đó tham số là kiểu của record.
  - ✓ Một phương thức ghi đè của Object.GetHashCode().
  - ✓ Phương thức nạp chồng toán tử == và !=.
  - ✓ Triển khai interface System.IEquatable<T>.
- ✓Record cũng cung cấp một phương thức ghi đè Object.ToString() để trả về xâu kí tự mô tả thông tin dữ liệu của lớp.

Đặc điểm của positional records

- ✓Khi ta khai báo record tham số theo thứ tự, trình biên dịch sẽ tự tạo các phương thức sau:
  - ✓ Một phương thức khởi tạo đầy đủ tham số theo đúng thứ tự tham số được cung cấp trong khai báo record.
  - ✓ Các public properties cho các tham số trong constructor. Tất cả chúng đều là init-only(với record class) và là readonly record struct(nếu là record struct).
  - ✓ Một phương thức để trích xuất các thuộc tính từ record.
- ✓Ngoài ra ta có thể tự định nghĩa thêm các thuộc tính hoặc phương thức vào record.
- ✓Init-only là các thuộc tính chỉ đọc, được khởi tạo 1 lần khi tạo đối tượng sau đó không thể thay đổi.

```
// record readonly struct mô tả thông tin về nhiệt độ hàng ngày
8 references
public readonly record struct DailyTemperature(double HighTemp, double LowTemp)
{
    0 references
    public double Mean => (HighTemp - LowTemp) / 2; // nhiệt độ trung bình
}
```

Ví dụ minh họa

```
0 references
static void Main()
{
    DailyTemperature[] data = new DailyTemperature[]
    {
        new DailyTemperature(30, 11),
        new DailyTemperature(32, 12),
        new DailyTemperature(35, 15),
        new DailyTemperature(37, 20),
        new DailyTemperature(31, 11)
    };
    ShowTempData(data);
}

1 reference
private static void ShowTempData(DailyTemperature[] data)
{
    foreach (var item in data)
    {
        Console.WriteLine(item);
    }
}
```

```
Microsoft Visual Studio Debug Console
DailyTemperature { HighTemp = 30, LowTemp = 11, Mean = 20.5 }
DailyTemperature { HighTemp = 32, LowTemp = 12, Mean = 22 }
DailyTemperature { HighTemp = 35, LowTemp = 15, Mean = 25 }
DailyTemperature { HighTemp = 37, LowTemp = 20, Mean = 28.5 }
DailyTemperature { HighTemp = 31, LowTemp = 11, Mean = 21 }
```

Khi nào sử dụng cái nào?

- ✓Sử dụng kiểu class để tạo các kiểu hướng đối tượng tập trung chủ yếu vào trách nhiệm và hành vi của các đối tượng.
- ✓Sử dụng kiểu struct để tạo các cấu trúc dữ liệu nhỏ lưu trữ dữ liệu và các dữ liệu đó đủ nhỏ để việc sao chép hiệu quả.
- ✓Sử dụng kiểu record khi muốn tạo một kiểu có thể so sánh dựa trên giá trị thuộc tính, không muốn sao chép giá trị của đối tượng và muốn sử dụng các biến theo kiểu tham chiếu thay vì kiểu giá trị.

Keyword with

- ✓Mục đích của record là tạo các đối tượng bất biến về giá trị và tham chiếu.
- ✓Ta có thể tạo record mới tương tự record đã có sử dụng biểu thức with(with expression).
- ✓Biểu thức with là một copy construction với khả năng sửa đổi giá trị các thuộc tính nếu muốn.
- ✓Kết quả là một đối tượng record mới được tạo ra trong đó mỗi thuộc tính đã được sao chép từ record đã tồn tại trước đó với vài tùy chọn sửa đổi thuộc tính.
- ✓Đối tượng record ban đầu không bị thay đổi gì.
- ✓Việc sao chép sẽ copy các thuộc tính kiểu value type và giữ nguyên tham chiếu của các thuộc tính kiểu tham chiếu. Do đó record gốc và record mới được tạo ra sẽ cùng tham chiếu tới cùng đối tượng trong thuộc tính kiểu tham chiếu.

Keyword with

✓Ví dụ:

```
DailyTemperature todayTemp = new DailyTemperature(30, 12);
var otherTemp = todayTemp with { };
var nextDayTemp = todayTemp with { HighTemp = 35 };
Console.WriteLine(todayTemp);
Console.WriteLine(otherTemp);
Console.WriteLine(nextDayTemp);
Console.WriteLine(todayTemp == nextDayTemp);
Console.WriteLine(todayTemp == otherTemp);
```

Microsoft Visual Studio Debug Console

```
DailyTemperature { HighTemp = 30, LowTemp = 12, Mean = 21 }
DailyTemperature { HighTemp = 30, LowTemp = 12, Mean = 21 }
DailyTemperature { HighTemp = 35, LowTemp = 12, Mean = 23.5 }
False
True
```

Keyword with

✓Ví dụ 2:

```
    Person person = new Person("Huong", "Pham", new string[] {"0972 032 178"});
    Person other = person with { FirstName = "Long" };
    Console.WriteLine(person);
    Console.WriteLine(other);
    Console.WriteLine(person.PhoneNumbers[0]);
    // thay đổi giá trị của số điện thoại
    other.PhoneNumbers[0] = "0359 259 315"; // ok
    Console.WriteLine(person.PhoneNumbers[0]);
    Console.WriteLine(other.PhoneNumbers[0]);
}
```

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

```
Microsoft Visual Studio Debug Console
Person { FirstName = Huong, LastName = Pham, PhoneNumbers = System.String[] }
Person { FirstName = Long, LastName = Pham, PhoneNumbers = System.String[] }
0972 032 178
0359 259 315
0359 259 315
```

Nội dung tiếp theo

Tính chất kế thừa