Bài 8.5: Cấu trúc DateOnly

- ✓ Mục đích và các thuộc tính
- ✓ Các phương thức thường dùng
- ✓ Ví dụ minh họa

Mục đích sử dụng và các thuộc tính

- ✓Cấu trúc readonly DateOnly dùng để mô tả thông tin về ngày tháng năm dương lịch.
- ✓Giá trị hợp lệ trong khoảng 1/1/1 đến 31/12/9999 dương lịch sau công nguyên.
- ✓DateOnly và TimeOnly được hỗ trợ từ C# 10.
- ✓Các thuộc tính:
  - ✓ Day: lấy ra giá trị ngày của đối tượng hiện thời.
  - ✓ DayNumber: lấy số ngày của ngày hiện tại tính từ ngày 1/1/1.
  - ✓ DayOfWeek: lấy ngày của tuần từ đối tượng hiện thời.
  - ✓ DayOfYear: lấy ngày của năm tính từ ngày 1/1 của năm trong đối tượng hiện thời.
  - ✓ MaxValue, MinValue: lấy ngày tối đa/tối thiểu có thể tạo từ struct.
  - ✓ Month: lấy tháng từ đối tượng hiện thời.
  - ✓ Year: Lấy năm từ đối tượng hiện thời.

Các constructor

- ✓ DateOnly(int year, int month, int day): tạo đối tượng DateOnly với ngày tháng năm cho trước.
- ✓ DateOnly(int year, int month, int day, Calendar calendar): tạo đối tượng của DateOnly với calendar cho trước.

Các phương thức khác

Phương thức và mô tả
AddDays(int value); thêm value ngày vào số ngày trong đối tượng hiện thời. Trả về đối tượng DateOnly mới có số ngày bằng số ngày hiện tại và số ngày thêm vào.
AddMonths(int value); thêm value tháng vào đối tượng hiện thời. Trả về một đối tượng DateOnly mới sau khi bổ sung value tháng vào.
AddYear(int value); thêm value năm vào đối tượng hiện thời và trả về một đối tượng DateOnly mới.
CompareTo(DateOnly other); CompareTo(Object other); So sánh giá trị của other với đối tượng hiện thời và trả về số nguyên thể hiện sự khác biệt(sớm hơn, muộn hơn hay bằng với) so với tham số nhận được.
Equals(DateOnly other); Equals(Object other); Trả về true nếu đối tượng hiện thời tương đương với other. Trả về False nếu ngược lại.
FromDateTime(DateTime dateTime); trả về đối tượng DateOnly từ phần ngày tháng năm của đối tượng dateTime.

Các phương thức khác

FromDayNumber(int dayNumber); trả về đối tượng của DateOnly từ số ngày cung cấp trong tham số.
GetHashCode(); trả về mã băm của đối tượng hiện thời.
Parse(string str); trả về đối tượng DateOnly tương ứng string đã cung cấp. Ngoại lệ nếu str là null. Parse(string s, IFormatProvider? provider, DateTimeStyles style); chuyển đổi string s sang DateOnly sử dụng provider và style cho trước. Ngoại lệ nếu s null và s không chứa kí tự hợp lệ thể hiện ngày tháng năm.
ToDateTime(TimeOnly time); trả về một đối tượng DateTime với thông tin ngày tháng năm như đối tượng hiện thời và thời gian như tỏng tham số. ToDateTime(TimeOnly time, DateTimeKind kind); tương tự như trên nhưng sử dụng thêm tham số kind.
ToShortDateString(); trả về string ngày tháng năm rút gọn của đối tượng hiện thời.
ToLongDateString(); trả về string đầy đủ về ngày tháng năm đại diện cho đối tượng hiện thời.
ToString(); chuyển đối tượng hiện thời thành string đại diện tương ứng ở dạng ngắn. ToString(IFormatProvider? provider); tương tự trên nhưng sử dụng provider.
TryParse(string? S, out DateOnly result); chuyển string s sang đối tượng DateOnly result tương ứng và trả về một giá trị bool chỉ ra rằng liệu việc chuyển đổi có thành công hay không.

Các toán tử so sánh

- ✓Phép == so sánh tự tương đương giữa hai đối tượng.
- ✓Phép > xác định xem đối tượng thứ nhất có muộn hơn đối tượng thứ hai không.
- ✓Phép >= xác định xem liệu một đối tượng DateOnly có tương đương hay muộn hơn đối tượng DateOnly ở toán hạng bên phải không.
- ✓Phép < xác định xem đối tượng ở toán hạng trái có sớm hơn đối tượng ở toán hạng bên phải không.
- ✓Phép <= xác định xem đối tượng ở toán hạng trái có sớm hơn hoặc tương đương đối tượng ở toán hạng bên phải không.
- ✓Phép!= xác định sự không tương đương của hai đối tượng DateOnly.

Ví dụ

```
Console.OutputEncoding = Encoding.UTF8;
DateOnly dateOnly = new DateOnly(2025, 10, 15);
Console.WriteLine("==> Thông tin về đối tượng DateOnly vừa tạo: ");
Console.WriteLine($"Ngày: {dateOnly.Day}");
Console.WriteLine($"Tháng: {dateOnly.Month}");
Console.WriteLine($"Năm: {dateOnly.Year}");

// thực hiện các thao tác khác
Console.WriteLine("==> Sau khi thêm 17 ngày vào đối tượng Date ở trên: ");
var otherDate = dateOnly.AddDays(17);
Console.WriteLine($"Ngày: {otherDate.Day}");
Console.WriteLine($"Tháng: {otherDate.Month}");
Console.WriteLine($"Năm: {otherDate.Year}");

Console.WriteLine("==> Sau khi thêm 17 tháng vào đối tượng Date ở trên: ");
otherDate = dateOnly.AddMonths(17);
Console.WriteLine($"Ngày: {otherDate.Day}");
Console.WriteLine($"Tháng: {otherDate.Month}");
Console.WriteLine($"Năm: {otherDate.Year}");

// So sánh các đối tượng của DateOnly
Console.WriteLine($"{dateOnly} > {otherDate} ? {dateOnly > otherDate}");
Console.WriteLine($"{dateOnly} == {otherDate} ? {dateOnly == otherDate}");
Console.WriteLine($"{dateOnly} <= {otherDate} ? {dateOnly <= otherDate}");
Console.WriteLine(dateOnly.ToLongDateString());
```

Ví dụ

```
==> Thông tin về đối tượng DateOnly vừa tạo:
Ngày: 15
Tháng: 10
Năm: 2025
==> Sau khi thêm 17 ngày vào đối tượng Date ở trên:
Ngày: 1
Tháng: 11
Năm: 2025
==> Sau khi thêm 17 tháng vào đối tượng Date ở trên:
Ngày: 15
Tháng: 3
Năm: 2027
10/15/2025 > 03/15/2027 ? False
10/15/2025 == 03/15/2027 ? False
10/15/2025 <= 03/15/2027 ? True
Wednesday, October 15, 2025
```

Nội dung tiếp theo

Tìm hiểu struct TimeOnly