

مجموعه نرم‌افزاری تاپ سادی

سربارگذاری عملگرها در C# برای کلاسها و داده‌ها

معرفی

سربارگذاری عملگرها این ویژگی را دارد که معنای عملگرهای C# را با توجه به کلاسهایی که خودمان ساخته ایم تعریف کنیم. سربارگذاری عملگرها یکی از مهمترین و جذاب ترین امکانات C# است چون گسترش داده‌ها یا همان Type Extensibility را عرضه می‌کند. گسترش داده‌ها ویژگی است که به ما امکان می‌دهد تا به محیط برنامه نویسی خود انواع داده‌های جدید اضافه کرده و بدون آنکه کوچکترین نقطه ضعفی داشته باشیم، از آنها استفاده کنیم. مثلاً می‌توان یک نوع داده جدید تعریف کرد و سپس عملگرها را با توجه به آن داده سربارگذاری کرد. پس از انجام این کارها آن نوع داده جدید کاملاً به خود متکی خواهد بود و می‌توان هر نوع داده درون ساخته C# استفاده کرد.

سربارگذاری عملگرها در C# تفاوتی با C++ دارد که تفاوت مهم آن در این است که در C# عملگرها وابسته به نمونه نیستند و این عملگرها بصورت static تعریف خواهند شد و این عمل تا حدی شبیه به سربارگذاری عملگرها توسط توابع دوست (friend) در C++ می‌باشد. بنابراین استفاده از عملگر this در داخل این توابع امکان پذیر نمی‌باشد. لازم است یادآوری کنم که در C++ در سربارگذاری عملگرها وقتی از سربارگذاری نمونه استفاده می‌شود، عملگر سمت چپ همان خود نمونه یا کلاس بوده و کلاس یا داده سمت راست عملوند به عنوان پارامتر به تابع ارسال می‌گردد.

بر خلاف C++ در C# اجازه سربارگذاری تمامی عملگرها به کاربر داده نمی‌شود و همچنین استفاده از بعضی عملگرها وابسته به سربارگذاری عملگرهای دیگر می‌باشد. در جدول زیر این محدودیتها به صورت خلاصه آورده شده است.

عملگرها	قابلیت سربارگذاری
- , ++ , ~ , ! , - , + false , true , -	عملگرهای یکتایی که قابل سربارگذاری هستند
& , % , / , * , - , + << , >> , ^ , ,	عملگرهای دوتایی که قابل سربارگذاری هستند
, &&	این عملگرها قابل سربارگذاری نیستند ولی اگر عملگرهای & و بارگذاری شده باشند قابل استفاده می‌باشد. (توضیح داده خواهد شد.)
[]	این عملگر قابل سربارگذاری نیست ولی می‌توان به عنوان indexer استفاده کرد (توضیح داده خواهد شد.)
()	این عملگرها قابل سربارگذاری نیستند ولی می‌توان اعمال casting را انجام داده که تحت عنوان implicit و explicit توضیح داده خواهد شد.
=& , =% , =/ , +- , =+ =<< , =>> , ^= , = ,	این عملگرها قابل سربارگذاری نیستند ولی در صورتی که عملگر متناظر تعریف شده باشد، قابل استفاده است. مثلاً اگر عملگر + سربارگذاری شده باشد می‌توان از += استفاده کرد.
new , <- , :? , . , = sizeof , , , is	این عملگرها قابل سربارگذاری نیستند.

typedef	
, => , < , > , != , == =<	این عملگرها قابل سربرگذاری هستند ولی باید به صورت زوج تعریف شوند به عنوان مثال اگر عملگر == سربرگذاری شود، عملگر != نیز باید سربرگذاری شود. این عمل برای زوجهای (<,>) و (<=,>=) نیز صادق است.

در ادامه به توضیح هر یک از موارد فوق می پردازیم.

۱- عملگرهای یکتایی + , - , ! , ~ , ++ , -- , true , false

چنانچه اشاره شد این عملگرها قابل سربرگذاری هستند. به غیر از عملگرهای true و false بقیه عملگرها در C++ نیز وجود داشتند. عملگرهای true و false درست یا غلط بودن یک object را نشان می دهند به عنوان نمونه به مثال زیر توجه کنید.

```
Digit    D;
D = new Digit();
if(d)      // call true(d) function
    // Message for True Object
```

همچنین در زیر مثالی از بارگذاری عملگر یکتایی – ارائه شده است.

```
class Test{
    int    x , y ;
    public static Test operator - (Test t){
        Test Temp;
        Temp = new Test();
        Temp.x = -t.x;
        Temp.y = -t.y;
        return Temp;
    }
}
```

۲- عملگرهای یکتایی + , - , * , / , % , & , | , ^ , << , >>

این عملگرها نیز قابل سربرگذاری هستند که برای توضیح

برای خواندن بقیه مقاله به [سایت دوقلوهای برنامه نویس](#) مراجعه فرمایید

برگرفته از سایت جالب برنامه نویس

www.PishgamSoft.com