

Как работает ограничение where T: Class1

Обобщения (generics) являются аналогом механизма языка C++ который называется шаблонами (templates).

Как и шаблоны, обобщения позволяют одинаковым способом производить одинаковые действия для различных типов данных.

Однако реализация обобщений в .NET несколько отличается от реализации шаблонов в компиляторе языка C++. Если программа на языке C# содержит обобщенный класс, то он будет скомпилирован в MSIL с обобщенным типом. Подстановка реальных типов вместо обобщенного типа будет произведена уже на этапе JIT-компиляции. Таким образом, работу с обобщениями поддерживает не только компилятор, но и сама среда .NET runtime.

Как и шаблоны в языке C++, обобщенные типы в языке C# указываются в треугольных скобках после имени класса. Обобщенные типы принято обозначать прописными латинскими буквами или начинать с прописной буквы.

В языке C# реализован механизм ограничений, позволяющий накладывать их на обобщенный тип T.

Where T: Class1 – тип T должен наследоваться от класса Class1

```
class Class1
{
    public int Id { get; private set; }
    public int Sum { get; set; }
    public Class1(int _id)
    {
        Id = _id;
    }
}
```

Для различных действий мы можем определить класс GenericClass2, который для выполнения всех операций будет использовать объекты класса Class1.

Но у класса Class1 может быть много наследников. И в этом случае в качестве универсального параметра можно установить тип Class1:

```
class GenericClass2 <T> where T: Class1
{
    public T FromClass1 { get; set; }
    public T ToClass1 { get; set; }
    public int Sum { get; set; }

    FromClass1.Sum -= Sum;
    ToClass1.Sum += Sum;}

```

С помощью выражения `where T : Class1` мы указываем, что используемый тип `T` обязательно должен быть классом `Class1` или его наследником. Благодаря подобному ограничению мы можем использовать внутри класса `GenericClass2` все объекты типа `T` именно как объекты `Class1` и соответственно обращаться к их свойствам и методам.

