Как работает ограничение where T: Interface1?
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.
В языке Java существует механизм, аналогичный тому, что и в языке C#, который также называется обобщениями.

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 интересный механизм ограничений, позволяющий накладывать их на обобщенный тип T.

where T: Interface1 - Тип T должен реализовывать интерфейс Interface1
Создадим интерфейс:

```
interface I1
{
string I1_method();
}
```

Создадим обобщенный класс, содержащий ограничение:

```
class GenericClass2<T> where T : I1
{
private T i;
//Конструктор
public GenericClass2(T param) { this.i = param; }
//Приведение к строке
public override string ToString()
{
return i.I1_method();
}
}
```

Ограничение задается непосредственно при объявлении класса:

```
class GenericClass2<T> where T : I1
```