Configurations in Entity Framework Core

You learned about default Conventions in EF Core in the previous chapter. Many times we want to customize the entity to table mapping and do not want to follow default conventions. EF Core allows us to configure domain classes in order to customize the EF model to database mappings. This programming pattern is referred to as Convention over Configuration.

There are two ways to configure domain classes in EF Core (same as in EF 6).

1. By using Data Annotation Attributes
2. By using Fluent API

Data Annotation Attributes

Data Annotations is a simple attribute based configuration method where different .NET attributes can be applied to domain classes and properties to configure the model.

Data annotation attributes are not dedicated to Entity Framework, as they are also used in ASP.NET MVC. This is why these attributes are included in separate namespace *System.ComponentModel.DataAnnotations*.

The following example demonstrates how the data annotations attributes can be applied to a domain class and properties to override conventions.

```
[Table("StudentInfo")]
public class Student
{
    public Student() { }

    [Key]
    public int SID { get; set; }

    [Column("Name", TypeName="ntext")]
    [MaxLength(20)]
    public string StudentName { get; set; }

    [NotMapped]
    public int? Age { get; set; }


    public int StdId { get; set; }

    [ForeignKey("StdId")]
    public virtual Standard Standard { get; set; }
}
```

Data annotation attributes are the same in EF 6 and EF Core. Visit Data Annotations chapter in the EF 6 section for more information.

Fluent API

Another way to configure domain classes is by using Entity Framework Fluent API. EF Fluent API is based on a Fluent API design pattern (a.k.a Fluent Interface) where the result is formulated by method chaining.

Learn about Fluent API in the next chapter.