**使用数据注解**

ABP支持数据注解特性，假设我们正在开发一个Task应用服务，它用来创建一个Task并需要一个输入，如下所示：

public class CreateTaskInput

{

public int? AssignedPersonId { get; set; }

**[Required]**

public string Description { get; set; }

}

public class TaskAppService : ITaskAppService

{

public void CreateTask(**CreateTaskInput** input)

{

}

}

就像你看到的，没有写验证代码，因为ABP自动完成验证。ABP会验证如果输入为null，就抛出AbpValidationException

**正常化**

有时我们需要在验证后，执行一个额外的操作来整理DTO参数。ABP定义了IShouldNormailize接口，包含Normalize方法，如果你实现了这个接口，Normalize方法在验证后（在方法被调用前）被调用。

public class GetTasksInput : **IShouldNormalize**

{

public string Sorting { get; set; }

**public void Normalize()**

**{**

**if (string.IsNullOrWhiteSpace(Sorting))**

**{**

**Sorting = "Name ASC";**

**}**

**}**

}

**禁用验证**

对于自动验证类（查看“简介”小节），你可以使用这些特性来控制验证：

* DisableValidation特性：可用于 控制器，DTO的类，方法或属性 上来禁用验证。
* EnableValidation特性：在一个被禁用验证的类里，可以在方法上使用这个特性让当前方法的输入验证可用。

**自定义验证**

如果数据注解不够你用，你可以实现ICustomValidate接口，如下所示：

public class CreateTaskInput : **ICustomValidate**

{

public int? AssignedPersonId { get; set; }

public bool SendEmailToAssignedPerson { get; set; }

[Required]

public string Description { get; set; }

**public void AddValidationErrors(CustomValidatationContext context)**

**{**

**if (SendEmailToAssignedPerson && (!AssignedPersonId.HasValue || AssignedPersonId.Value <= 0))**

**{**

**context.Results.Add(new ValidationResult("AssignedPersonId must be set if SendEmailToAssignedPerson is true!"));**

**}**

**}**

}

ICustomValidate接口定义了AddValidationErrors方法，如果有验证错误，我们必须添加ValidationResult对象到context.Results列表。在验证进度里，如果有需要，你可以用context.IocResolver来[解析依赖](http://www.cnblogs.com/kid1412/p/5980068.html" \l "Resolving" \t "http://www.cnblogs.com/kid1412/p/_blank)。