继续Model验证的学习，顺道提及一下，今天面试的时候又遇到了几个新的问题，包括Code Smith的代码生成工具（类似T4模板，但更方便）、Dapper.net的轻量级ORM框架、服务的幂等性等，将慢慢的学习和分享。

首先介绍最重要的基于验证特性的声明式Model验证，ValidationAttribute是所有验证特性的抽象基类，主要内容如下表所示：

|  |  |
| --- | --- |
| 成员变量或函数 | 简介 |
| ErrorMessageResourceName | 错误消息所在资源项的名称 |
| ErrorMessageResourceType | 错误消息所在资源项的类型 |
| IsValid() | 在验证失败时返一个ValidationResult对象 |
| GetValidationResult() | 实际调用受保护的IsValid() |
| TypeId | 在需要多次使用同一验证特性时需要重写该属性，使得每一次的TypdId不相同。情形：假设需要控制不同级别人员用于不同工资范围，定岗定薪。 |

继承ValidationAttribute的验证特性类如下所示：

|  |  |
| --- | --- |
| 特性名称 | |
| RequiredAttribute | RangeAttribute |
| StringLengthAttribute | Max/MinLengthAttribute |
| RegularExpressionAttribute | CompareAttribute |
| CustomValidationAttribute |  |

在验证过程中起作用的是DataAnnotaionsModelValidator对象的Validate方法，首先根据容器对象创建出表示验证上下文的ValidationContext对象，并采用ModelMetadata的DisplayName来做为上下文的名称，然后调用Attribute属性的GetValidatioResult方法进行最终的验证。同时MVC还定义了一个System.Web.Mvc.DataAnnotationsModelValidator<TAttribute>的泛型类，我们常用的RequiredAttribute均继承于该类。

数据特性验证的提供器其包含一个静态的验证工厂集合ValidatableFactories，是一个以类型Type为key，指定委托DataAnnotationsValidatableObjectAdapterFactory为value的字典，比较少见的方式。该提供器的静态构造方法中已将常见特性的验证提供器加入，并提供静态注册方法注册新的验证提供器。之后的内容蒋大师分享了两种扩展，一个是将ValidationAttribute应用在Action的参数上，和J2EE中Spring MVC的方式一致，以及实现同一个Model类型实现多种方式等，就不一一介绍了。

最后，简要介绍客户端验证，在不用Razor引擎的前提下，这部分的主要价值就体现在关于JQuery插件的扩展，关于javascript，提到最多的概念就是PE（Progressive Enhancement）渐进性增强和非入侵式Unobtrusive，最主要对的意思就是页面可以在不支持JS的情况下显示基本内容，再浏览器允许的情况增强显示效果。这里用到了最常见的前端验证框架文件jquery.validate.js，可以通过设置class的内联方式来完成验证，也可以直接通过validate方法来设置验证，代码如下：

|  |
| --- |
| $(document).ready(function () {  $("form").validate({  rules: {  name: { required: true },  birthday: { required: true, date: true },  blogAddress: { required: true, url: true },  emailAddress: { required: true, email: true },  },  message: {  name: { required: "请输入姓名" },  birthday: { required: "请输入出生日期", date: "请输入一个合法的日期" },  blogAddress: { required: "请输入姓名", url: "请输入一个合法的URL" },  emailAddress: { required: "请输入Email地址", email: "请输入一个合法的Email地址" },  }  });  }); |

在框架中，基于JQuery的Model验证其实就是根据数据的验证特性生成相应的js代码，指定的html元素具有”data-val”属性和一系列的以“data-val-”为前缀的属性。并在之后一个<span>元素，该元素的CSS样式为“field-validation-valid”，当验证失败时替换为”field-validation-error”。在表示客户端验证的ModelClientValidationRule中，有一个ValidationParameters来表示验证参数名和参数值。之后蒋大师又介绍了一个自定义验证的例子，我只节选出js作为自己学习JQuery插件的练习。

|  |
| --- |
| jQuery.validator.addMethod("agerange", function (value, element, params) {  value = value.replace(/(^\s\*)(\s\*$)/g, "");  if (!value) {  return true;  }  var minAge = params.minage;  var maxAge = params.maxage;  var birthDateArray = value.split("-");  var birthDate = new Date(birthDateArray[0], birthDateArray[1], birthDateArray[2]);  var currentDate = new Date();  var age = currentDate.getFullYear() - birthDate.getFullYear();  return age >= minAge && age <= maxAge;  });  jQuery.validator.unobtrusive.adapters.add("agerange", ["minage", "maxage"], function (options) {  options.rules["agerange"] = {  minage: options.params.minage,  maxage:options.params.maxage  };  options.messages["agerange"] = options.message;  }); |

注：本文主要供自己学习，不妥之处望见谅。

参考资料：

[1]蒋金楠. ASP.NET MVC4框架揭秘[M]. 上海:电子工业出版社, 2012. 283-319