CKEditor

- Model: Use [UIHint("CKEditor")]

```
public partial class BaiViet
{
    [Key]
    [Display(Name="Mã bài viết")]
    public string maBaiViet { get; set; }
    [Display(Name="Tựa bài viết")]
    [Required(ErrorMessage="Phải nhập tiêu đề bài viết")]
    public string tuaBaiViet { get; set; }
    [Display(Name="Nội dung tóm tắt")]
    [Required(ErrorMessage = "Phải nhập nội dung tóm tắt")]
    public string noiDungTomTat { get; set; }
    [UIHint("CkEditor")]
    [Required(ErrorMessage = "Phải nhập nội dung chính")]
    [Display (Name="Nội dung chính")]
    [AllowHtml]
    public string noiDungChinh { get; set; }
    public Nullable<System.DateTime> ngayDang { get; set; }
    public string maLoai { get; set; }
```

1

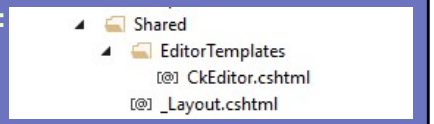
CKEditor

- Download CKEditor library: http://ckeditor.com/
- Copy CKEditor library to Script folder
- Create EditorTemplates in Shared Folder

2

CKEditor

- Create CKEditor.cshtml in View/Shared/ EditorTemplates:



- Code file:

```
<script src="~/Scripts/ckeditor/ckeditor.js"></script>
<textarea class="ckeditor" name="editor1"></textarea>
```

3

CKEditor

- Controller:

```
public class BaiVietController : Controller
{
    Personal_FSoftEntities1 db = new Personal_FSoftEntities1();
    //Action nhập bài viết
    public ActionResult NhapBaiViet()
    {
        ViewBag.LoaiBaiViet = new SelectList(db.LoaiBaiViets, "maLoai", "tenLoai");
        return View();
    }
```

4

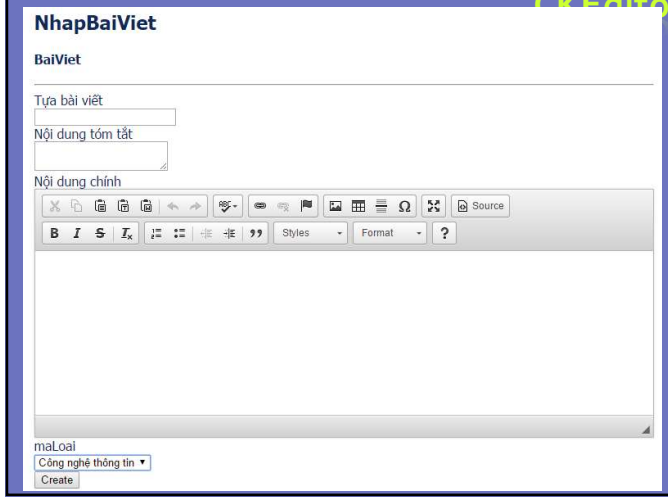
CKEditor

- Controller:

```
[HttpPost]
public ActionResult NhapBaiViet(BaiViet bv)
{
    bv.maBaiViet = "BV_" + DateTime.Now.ToString("yyyyMMddhhmmss");
    bv.ngayDang = DateTime.Now;

    if (ModelState.IsValid)
    {  //xử lý nhập bài viết
        db.BaiViets.Add(bv);
        db.SaveChanges();
        return RedirectToAction("Index");
    }//trường hợp sai dữ liệu, chọn lại Loại bài viết trên DropdownList
    ViewBag.LoaiBaiViet = new SelectList(db.LoaiBaiViets, "maLoai",
        "tenLoai", bv.maLoai);
    //trả lại trạng thái trên View với các thông tin bài viết
    return View(bv);
}
```

5

CKEditor



7

CKEditor

- View:

```
@model prjMVCExample.Models.BaiViet
@{
    ViewBag.Title = "NhapBaiViet";
    Layout = "~/Views/Shared/Admin_Layout.cshtml";
}
<h2>NhapBaiViet</h2>

@using (Html.BeginForm())
{
```

- ....

```
    @Html.LabelFor(model => model.noiDungChinh, new
{ @id = "control-label col-md-2" })
    <div class="col-md-10">
        @Html.TextAreaFor(model => model.noiDungChinh, new
{ @class = "ckeditor", @name = "editor1" })
        @Html.ValidationMessageFor(model => model.noiDungChinh)
```

6

Action Filter

Table of Contents

- Action Filters
  - [Authorize]
  - [AllowAnonymous]
  - [ValidateAntiForgeryToken]
  - [ValidateInput]
  - [HandleError]
  - [InitializeSimpleMembership]

9

Action Filter

- Action filter is used to perform the tasks occur at the time of implementation of an action different.
- Some Action Filter is available by MVC
  - [ValidateInput]
  - [Authorize]
  - [ValidateAntiForgeryToken]
  - [HandleError]

10

Action Filter

- Also we can write Action Filter for private purposes
  - [InitializeSimpleMembership]
- Filters can be used to filter:
  - A single action
  - The actions in a Controller
  - All actions of the project

11

[ValidateInput]

- HTML data is sended to Action
  - True: can send HTML data
  - False: receive error message
- Example:

```
public class HomeController : Controller
{
    [ValidateInput(false)]
    public ActionResult Send(Mail mail)
    {
```

12

[Authorize]

- [Authorize] is used to validate accessing the action with appropriate authorities
- There are 3 types:
  - **[Authorize]**: allow access after logging
  - [**Authorize(users="u1,u2,…")**]: allow access with user u1, u2,…
  - [**Authorize(roles="r1,r2,…")**]: allow access with users have roles r1, r2, …

13

[Authorize]

Must login

```
[Authorize]
[InitializeSimpleMembership]
public class AccountController : Controller
{
    public ActionResult ChangePassword()...

    [AllowAnonymous]
    public ActionResult Login(string returnUrl)...
```

allow anonymous access

15

[Authorize]

```
[Authorize(Roles="Administrator")]
public ActionResult Manage()
{
    return View();
}
```

```
[Authorize(Users = "thphuong")]
public ActionResult Config()
{
    return View();
}
```

```
[Authorize]
public ActionResult MyAccount()
{
    return View();
}
```

14

[ValidateAntiForgeryToken]

- [ValidateAntiForgeryToken]: prevent request forgery
-

```
<form action="/home/savedata" method="post">
    <%= Html.AntiForgeryToken() %>
    <input type="submit" value="Submit from App1" />
</form>
```

```
[HttpPost]
[ValidateAntiForgeryToken]
public ViewResult SaveData()
{
    return View();
}
```

16

[HandleError]

- Simply Try…Catch approach

```
public ActionResult TestMethod()
{
    try
    {
        //....
        return View();
    }
    catch (Exception e)
    {
        //Handle Exception;
        return View("Error");
    }
}
```

- *Problem with the above approach is we cannot reuse the exception handling logic across multiple action methods.*

17

[HandleError]

1. Global level Exception Handling
1. Web.config:

```
<system.web>
  <customErrors mode="On" defaultRedirect="Error" />
```

- **TestMethod:**

```
public ActionResult TestMethod()
{
    int x = 0;
    x /= x;
    return View();
}
```

18

[HandleError]

1. FilterConfig class

```
public class FilterConfig
{
    public static void RegisterGlobalFilters(GlobalFilterCollection filters)
    {
        filters.Add(new HandleErrorAttribute());
    }
}
```

2. **Where is RegisterGlobalFilters method invoked?**

19

[HandleError]

1. In Global.asax file in Applicaion_Start RegisterGlobalFilters method is invoked.

```
public class MvcApplication : System.Web.HttpApplication
{
    protected void Application_Start()
    {
        AreaRegistration.RegisterAllAreas();
        WebApiConfig.Register(GlobalConfiguration.Configuration);
        FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
        RouteConfig.RegisterRoutes(RouteTable.Routes);
    }
}
```

20

[HandleError]

- HandleError at controller level?

```
[HandleError(View = "TestMethod")]
public class TestController : Controller
{
    public ActionResult TestMethod()
    {
        int x = 0;
        x /= x;
        return View();
    }
}
```

21

[HandleError]

- HandleError at Action level

```
public class TestController : Controller
{
    [HandleError(View = "TestMethod")]
    public ActionResult TestMethod()
    {
        int x = 0;
        x /= x;
        return View();
    }
}
```

22

[HandleError]

- View:

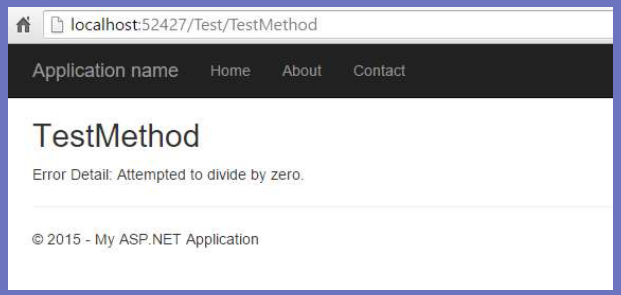
TestMethod.cshtml

```
@model System.Web.Mvc.HandleErrorInfo

@{
    ViewBag.Title = "TestMethod";
}

<h2>TestMethod</h2>
<div>
    Error Detail: @Model.Exception.Message
</div>
```

23

[HandleError]

- Web Result:



24

[HandleError]

- Different views for Different Exceptions
  - For that we will use overloaded version of HandleErrorAttribute constructor as follows.

```
[HandleError(ExceptionType=typeof(DivideByZeroException),View="TestMethod")]
[HandleError(ExceptionType=typeof(NullReferenceException),View="NullError")]
public class TestController : Controller
{
```

25