webAula

Padrão de estrutura de projetos C#.Net

Revisões

Autor	Data
Renan Moreira Rotondo	29/07/2013
Renan Moreira Rotondo	25/09/2013

Sumário

Introdução

Esta documentação foi criada para ser a norteadora da criação/reorganização de projetos de aplicativos C#.Net criados na webAula. A ideia é que ela se torne uma referência para o dia-a-dia dos profissionais do setor, com o objetivo de aperfeiçoar o serviço e tornar mais simples o entendimento das estruturas de projeto dos aplicativos da empresa.

Estrutura básica

A base do projeto será pautada em uma estrutura modular. **Módulo** é a parte do sistema responsável por uma tarefa bem definida e que pode ser acoplado a um sistema para permitir ao mesmo executar a tarefa disponibilizada pelo módulo.

Módulo é uma parte do sistema que utiliza a mesma arquitetura tecnológica do sistema, é responsável por atividades que satisfaz um assunto bem definido, as atividades do módulo utilizam tarefas e componentes comuns do sistema, um módulo ou vários módulos compõem um sistema. Um módulo também é representado por um grupo de componentes de software que atende a um assunto bem definido.

O módulo realiza um assunto no processo de informação que o sistema propõe atender.

Exemplo: imagine um site que possui 3 tipos de recurso: Fórum, Banner e Calendário de eventos. Cada um desses recursos representará um módulo na estrutura do projeto.

Regras de nomenclatura

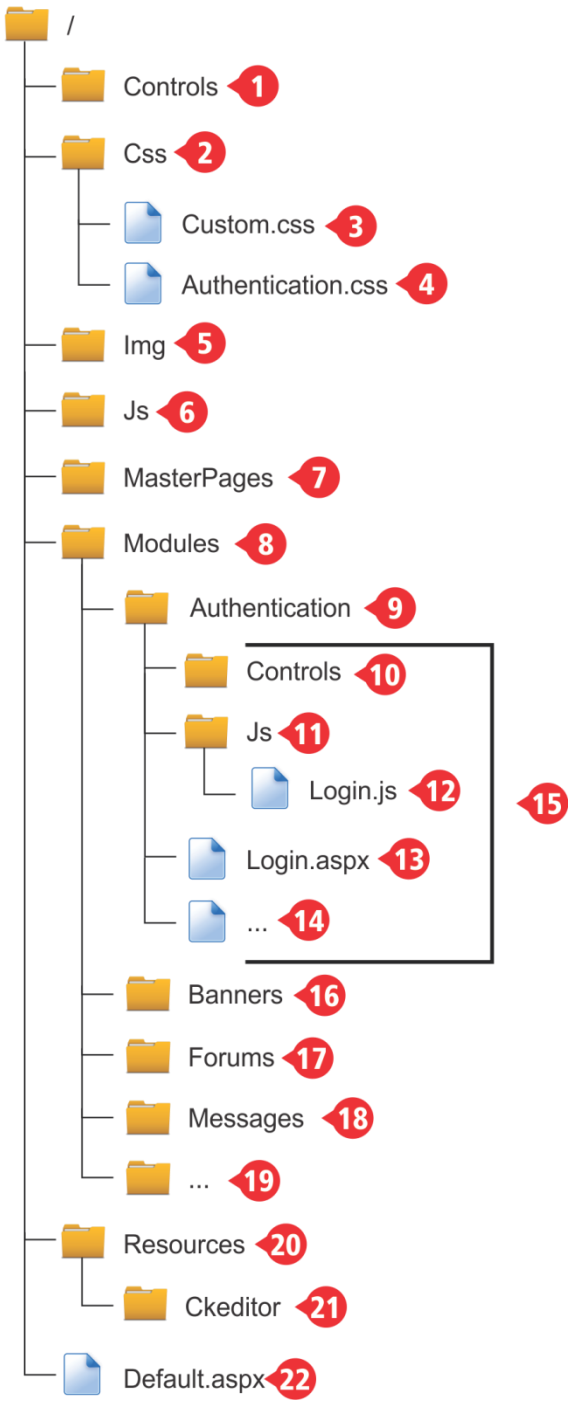
Algumas premissas para a criação e organização projetos C#.NET deverão ser seguidas. São elas:

- A nomenclatura dos arquivos deve seguir o padrão Upper CamelCase para as junções de palavras (primeira letra da palavra maiúscula, primeira letra das palavras anexadas maiúscula). http://pt.wikipedia.org/wiki/CamelCase
- A nomenclatura das classes Css devem seguir o padrão Lower camelCase (primeira letra da palavra minúscula, primeira letra das palavras anexadas maiúscula)..
- Os arquivos, pastas e estilos de cascata (CSS) deverão ter nome em inglês (Ex: 'ListTopics.aspx', 'TopicDetail.aspx', '.boxCalendar', Modules > Authentication)
- Evite manter arquivos de backup na estrutura do projeto no repositório de arquivos. Esses arquivos acabam se transformando em lixo e são replicados para os clientes (Ex: listTopics___.aspx, listTopics_v2.aspx)
- Evite manter comentadas estruturas HTML que não serão aproveitadas no código. Lembre-se que para backup de informações antigas, temos um repositório de arquivos com o histórico de versões.

- Os nomes dos arquivos devem ser facilmente identificáveis e devem explicitar a sua real funcionalidade. (Ex: tela de visualizar dados de um usuário = userView.aspx).
- Sempre utilize nomenclaturas sem acentos, espaços ou caracteres especiais.
- Atribuir um ID específico para blocos de código que contemplem um grupo de informações, uma área ou uma funcionalidade. (Ex: #header, #content, #footer, #boxFeaturedNews). Com isso, fica mais fácil criar especificidades nos códigos CSS.
- Sempre trabalhar os CSSs de modo que eles possuam alto grau de especificidade. (Ex: #content .nav {color: #ff0000}; .message.active {background: black})

Estrutura de pastas

A estrutura de pastas da do projeto seguirá o modelo esquemático disposto na figura abaixo:



Legenda:

1) **Controls:** Pasta onde ficarão armazenados os arquivos de controle padrão do projeto. (Ex: include do menu principal, include do rodapé principal, include da biblioteca webAula Standards – http://wastandards.webaula.com.br , etc.)

2) **Css:** Pasta onde ficarão armazenados os arquivos CSS genéricos do projeto.(Ex: estilos referentes ao cabeçalho e rodapé do site). Poderão haver arquivos com extensão SASS (http://sass-lang.com/), caso esse tipo de tecnologia esteja sendo usado. As declarações CSS pertencentes a um determinado módulo, deverão estar separadas das declarações genéricas presentes no arquivo Custom.css.

3) **Custom.css**: Arquivo com os estilos básicos do projeto.

4) **Authentication.css**: Exemplo de arquivo com os estilos do módulo denominado authentication.

5) **Img**: Pasta onde ficarão armazenadas as imagens genéricas do projeto. (Ex: logomarca do cliente, plano de fundo do corpo da página, etc.)

6) **Js**: Pasta onde ficarão armazenados os métodos javascript padrão do projeto. (Ex: funções javascript genéricas, usadas em todo o projeto, biblioteca jquery, etc.).

7) **MasterPages**: Pasta onde ficarão armazenadas as páginas mestre do projeto.

8) **Modules**: Pasta onde ficarão estocados os módulos do projeto. (Ex: Na imagem, foram inseridos 4 exemplos – módulo de autenticação, módulo de banners, módulo de fóruns, módulo de mensagens)

9) **Pasta de módulo**: Pasta de um módulo em específico

10) **Controls**: Pasta com os arquivos de inclusão exclusivos do módulo. (Ex: includes de boxes e de widgets do módulo).

11) **Js**: Pasta com os arquivos javascript exclusivos das telas utilizadas pelo módulo. Cada tela terá um arquivo javascript correspondente. Esse javascript deverá ter o nome da tela ao qual ele se refere.

12) **Login.js:** Arquivo javascript da tela de login, usada como exemplo na imagem.

13) **Login.aspx:** Arquivo de tela modelo do módulo em questão.

14) **...**: Demais arquivos de tela do módulo.

15) **Estrutura de pastas e arquivos de um módulo**: Essa estrutura deve se repetir em todos os módulos do projeto

16) **Pasta de módulo**: Pasta de um módulo em específico

17) **Pasta de módulo**: Pasta de um módulo em específico

18) **Pasta de módulo**: Pasta de um módulo em específico

19) **Demais pastas de módulos do projeto.**

20) **Resources:** pasta que armazena os recursos e aplicativos de terceiros de grande porte, usados no projeto

21) **Ckeditor:** Exemplo de recurso

22) **Default.aspx:** Página inicial do projeto