

## MemberShip e Roles

No ASP Clássico e em outras linguagem para web você tinha que fazer todo o trabalho de gerenciamento de usuários e grupos: Criar usuários, paginas de logins, trocas de senhas, recuperação de senhas, autenticação, criação de grupos de usuários, gerenciamento de grupos além de toda a estrutura de armazenamento necessária para isto tudo. Você também tinha que programar toda a parte de autorização de seu aplicativo: normalmente cada grupo de usuários estava restrito a um determinado numero de ações que poderiam ser executadas e ou paginas a qual ele poderia acessar.

Na primeira versão do ASP.NET muita coisa mudou, a programação de tudo isso citado no parágrafo anterior ficou um pouco mais fácil, mas ainda era preciso muito trabalho manual.

No ASP.NET 2.0, um novo recurso denominado MemberShip, veio para tornar tudo isso muito mais fácil. Mais do que isso, para revolucionar o gerenciamento de usuário, grupos e permissões. Além de uma série de controles prontos para gerenciamento de usuários e autenticação, o ASP.NET pode agora cuidar do armazenamento de tudo isso automaticamente através de um banco de dados do SQLExpress ou em qualquer outra fonte de dados que você quiser, inclusive o Active Directory.

## Definindo um Site para utilizar Forms Autentication

Uma aplicação Web pode ser totalmente publica, com um porta de noticias, ou pode requerer autenticação, como um InternetBanking, ou ainda pode ser um misto de ambos: Um site de comércio eletrônico onde você navega e compra de forma anônima e se autentica na hora do pagamento.

A autenticação em uma aplicação Web pode se dar de varias formas: Integrada com o Active Directory, onde poderão ser utilizadas as credencias que o usuário logou no diretório, Passport, que é um serviço pago mantido pela Microsoft e que já teve seu fim decretado, ou Forms, em que o usuário é autenticado através de informações digitadas em um formulário.

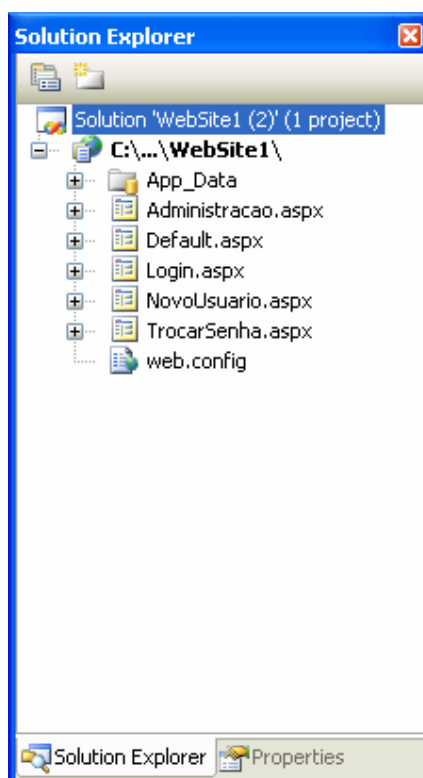
A forma de autenticação na mais comum na internet é Forms. Windows é util por exemplo, em uma aplicação de Intranet, em que podemos aproveitar as credencias do usuário no Diretório.



Sugiro que você acompanhe os exemplos passa a passo, já que uma etapa é diretamente dependente da outra.

Ao criar uma nova aplicação ASP.NET, por padrão ele não terá qualquer tipo de autenticação e seus acesso será irrestrito.

Para conhecermos os conceitos crie uma aplicação com cinco WebForms: Administracao.aspx, Default.aspx, Login.aspx, NovoUsuario.aspx e TrocarSenha.aspx. Observe a aplicação na imagem abaixo:




Rode a aplicação e note que você pode navegar livremente entre os formulários.

## Preparando o Arquivo web.config

Para definir autenticação Forms devemos fazer algumas alterações no arquivo web.config como no exemplo abaixo:



Você pode fazer as alterações necessárias no web.config através da ferramenta ASP.NET Configuration, que pode ser acionada através do ícone  no Solution Explorer.

```
<authentication mode="Forms">
  <forms name=".ASPXAUTH"
    loginUrl="login.aspx"
    defaultUrl="default.aspx">
  </forms>
</authentication>
```

Você está informando ao ASP.NET que vai utilizar autenticação do tipo Forms, que o nome do cookie de autenticação utilizado será .ASPXAUTH (padrão), que o formulário de autenticação será login.aspx e que o formulário padrão será default. Outras configurações podem ser adicionadas a este nó, estas exibidas são as mais relevantes.

Se você rodar a aplicação agora perceberá que ainda poderá navegar livremente entre as páginas. O que deu errado? Embora você tenha definido um método de autenticação, você ainda não restringiu o acesso de usuários anônimos a aplicação.

Isso deve ser feito com a adição de uma de um nó *autorization*, onde negamos acesso a usuários desconhecidos (não autenticados) representado pelo "?":

```
<authorization>
    <deny users="?" />
</authorization>
```

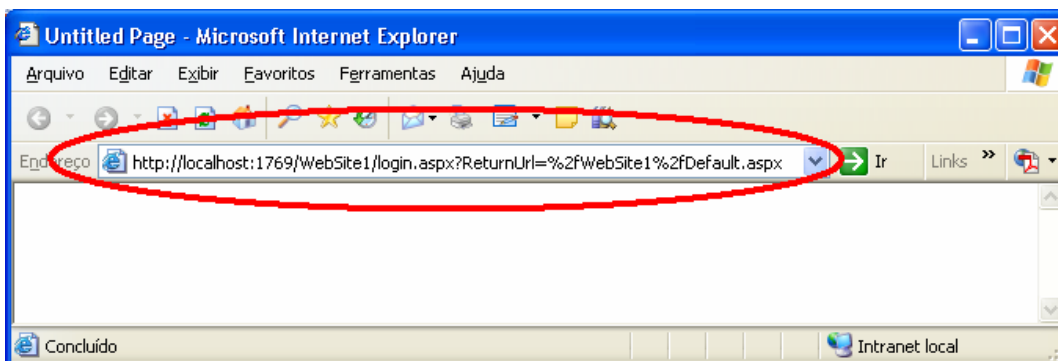


Ao final deste capítulo, apresento o web.config completo, da maneira que o seu deverá estar após todos os exemplos. Utilize-o para localizar onde determinada configuração deve se encaixar. Não copie o arquivo inteiro.

Agora defina a página clientes como página inicial e rode a aplicação.

Você deve observar duas coisas:

- Você foi redirecionado para a página login.aspx, pois é a página definida para autenticação em nosso arquivo de configuração.
- Na URL, o ASP.NET adicionou a página inicial solicitada, de forma que você possa ser redirecionado automaticamente após a autenticação



Agora estamos prontos para autenticar o usuário. Porém antes de autenticá-los precisamos criá-los, é o que vamos ver a seguir.

## Criando novos usuários com o controle CreateUserWizard

Criamos uma página para criação de usuários, NovoUsuario.aspx. Em aplicações este tipo de pagina normalmente tem seu acesso irrestrito, ou seja, usuários anônimos podem entrar livremente e se cadastrarem. Porém como definimos autenticação por formulários, nenhum usuário conseguirá acessá-las sem estar autenticado. Temos que torná-la publica, ou seja, de acesso anônimo.

Precisar definir algumas paginas, ou diretórios inteiros, com acesso anônimo e outras não é algo comum, portanto é algo que você vai usar com frequência em suas futuras aplicações.

Para definirmos uma determinada página como publica basta adicionarmos um nó location após o nó </system.web> já existente em nosso web.config, da seguinte forma:

```
<location path="NovoUsuario.aspx">
    <system.web>
```

```
<authorization>
  <allow users="*" />
</authorization>
</system.web>
</location>
```

Estamos definindo que a página NovoUsuario.aspx é de acesso anônimo. Defina NovoUsuario.aspx como página inicial e rode a aplicação. Note que você consegue acessa-la normalmente, embora o restante da aplicação ainda lhe redirecione para o Login.

Agora basta acionar um controle CreateUserWizard a nossa página de criação de usuários. Note que este controle possui dezenas de propriedades que permitem sua personalização, inclusive a troca das mensagens e labels. Não vamos entrar em detalhes porque é quase tudo bem intuitivo, basta explorar o controle por alguns minutos.

Rode novamente a aplicação e crie um novo usuário, preenchendo todos os campos obrigatórios, como na imagem abaixo:

Sign Up for Your New Account

User Name:

Password:

Confirm Password:

E-mail:

Security Question:

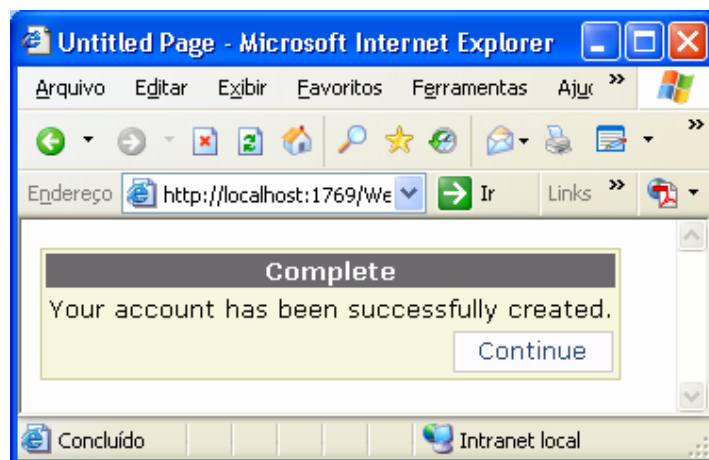
Security Answer:

Password length minimum: 7. Non-alphanumeric characters required: 1.

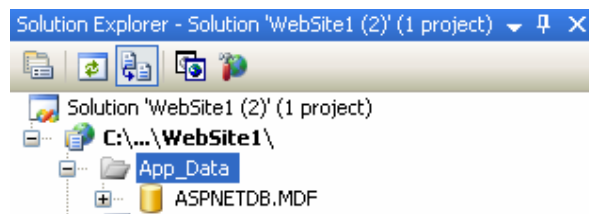
Concluído Intranet local

Por padrão, a senha deve conter no mínimo 7 caracteres, sendo no mínimo um caractere não alfanumérico ("#" por exemplo).

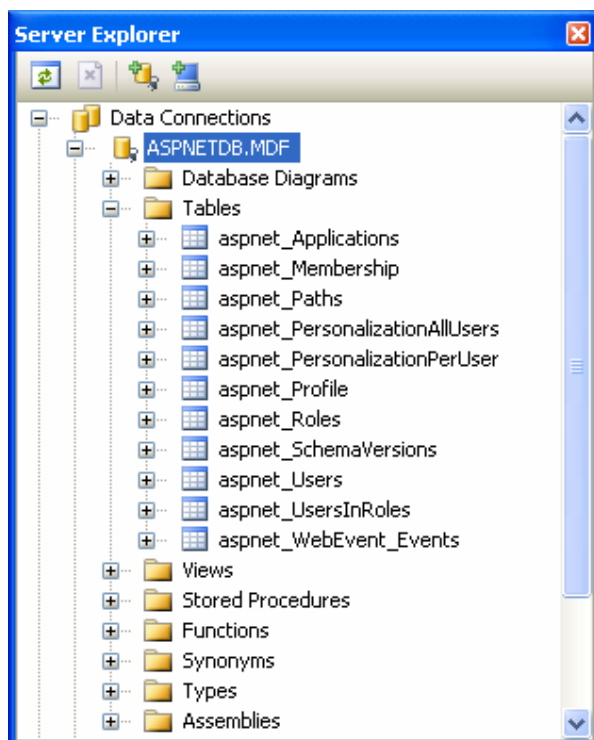
O ASP.NET informa que a criação do usuário ocorreu com sucesso.



Antes de continuarmos, vamos entender o que o ASP.NET fez internamente. Primeiramente abra o Solution Explorer, clique com o botão direito em cima da aplicação e clique em Refresh. Expanda o nó data. Note que foi criado um banco de dados SQL Server Express (mdb).



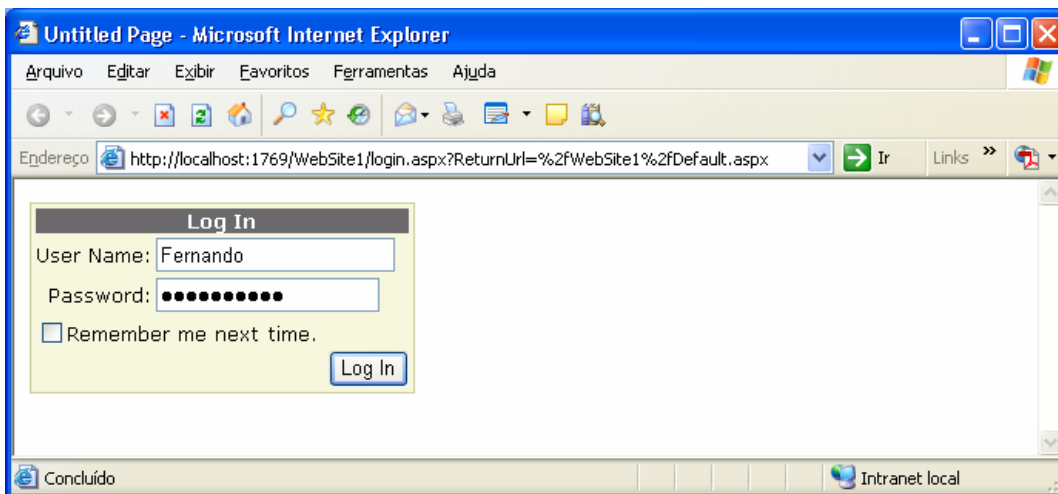
De um duplo clique sobre o banco para abri-lo no Server Explorer. Note que foi criada toda a estrutura de tabelas necessárias para o gerenciamento de usuários, grupos e permissões:



Se você examinar a tabela aspnet\_user, verá que o usuário criado esta armazenado nesta.

## Autenticando usuários com Login

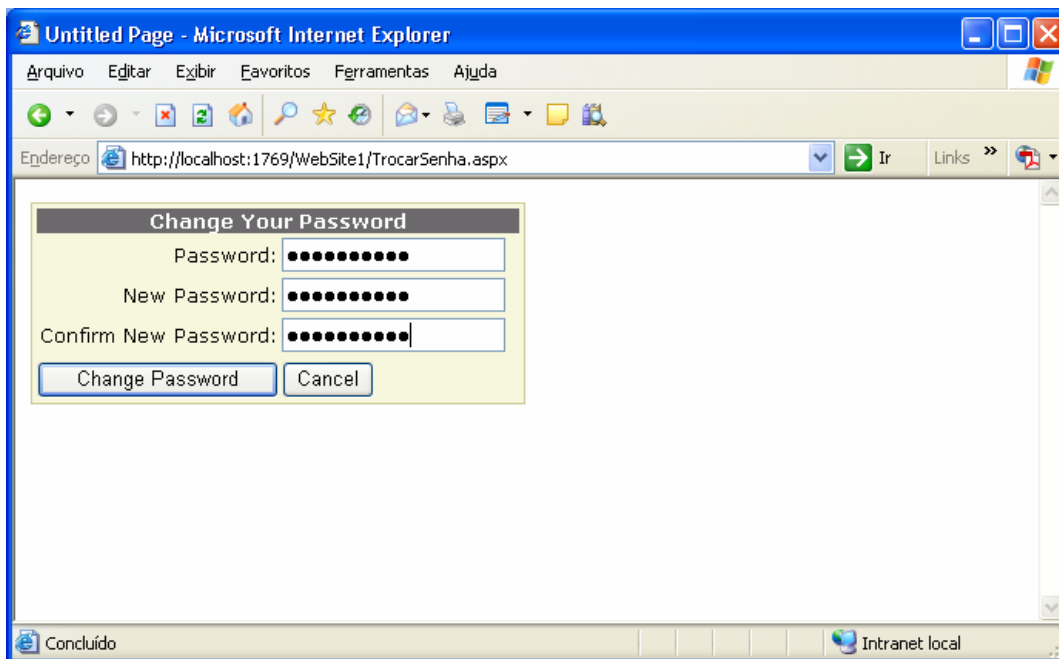
Criado nosso usuário, agora podemos autenticá-lo. Para isso vamos utilizar o controle Login, encontrado também na barra de ferramentas de mesmo nome. Coloque um controle na pagina Login.aspx, defina a pagina default.aspx com pagina inicial e rode a aplicação.



Se você informou o usuário e senha corretamente será redirecionado para a página Default e terá livre acesso a todas as demais páginas da aplicação.

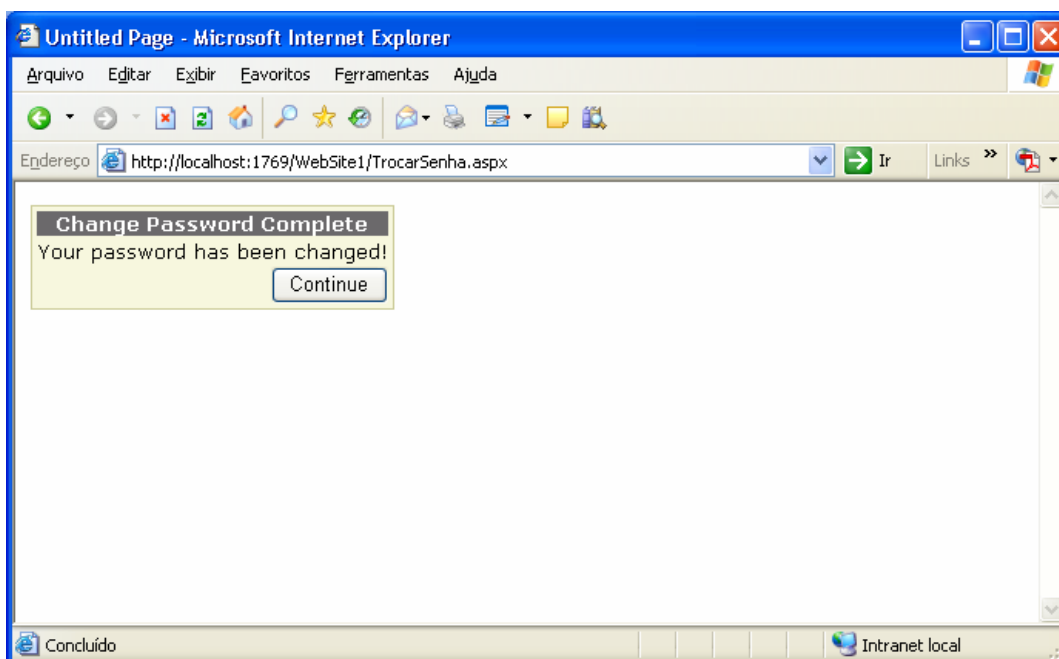
## Trocando a senha com ChangePassword

Outra funcionalidade pronta e encapsulada em um controle é o ChangePassword. Para utilizá-lo adicione o mesmo a pagina TrocarSenha.aspx, defina esta como pagina inicial e rode a aplicação, após o login você é redirecionado para uma troca de senha, onde você deve informar a senha atual, a nova senha e a confirmação da nova senha:



The screenshot shows a Microsoft Internet Explorer window titled 'Untitled Page - Microsoft Internet Explorer'. The address bar displays 'http://localhost:1769/WebSite1/TrocarSenha.aspx'. The main content area contains a form titled 'Change Your Password'. The form has three input fields: 'Password:', 'New Password:', and 'Confirm New Password:'. Each field is filled with a series of black dots. Below the input fields are two buttons: 'Change Password' and 'Cancel'. The status bar at the bottom indicates 'Concluído' and 'Intranet local'.

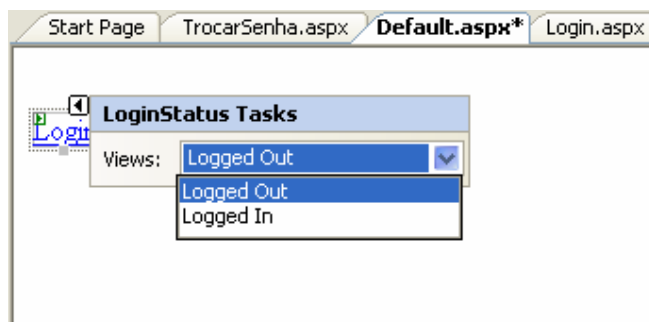
Uma mensagem informa que a senha foi trocada com sucesso:



The screenshot shows the same Microsoft Internet Explorer window. The main content area now displays a message box titled 'Change Password Complete' with the text 'Your password has been changed!'. Below the message is a single button labeled 'Continue'. The status bar at the bottom remains the same, showing 'Concluído' and 'Intranet local'.

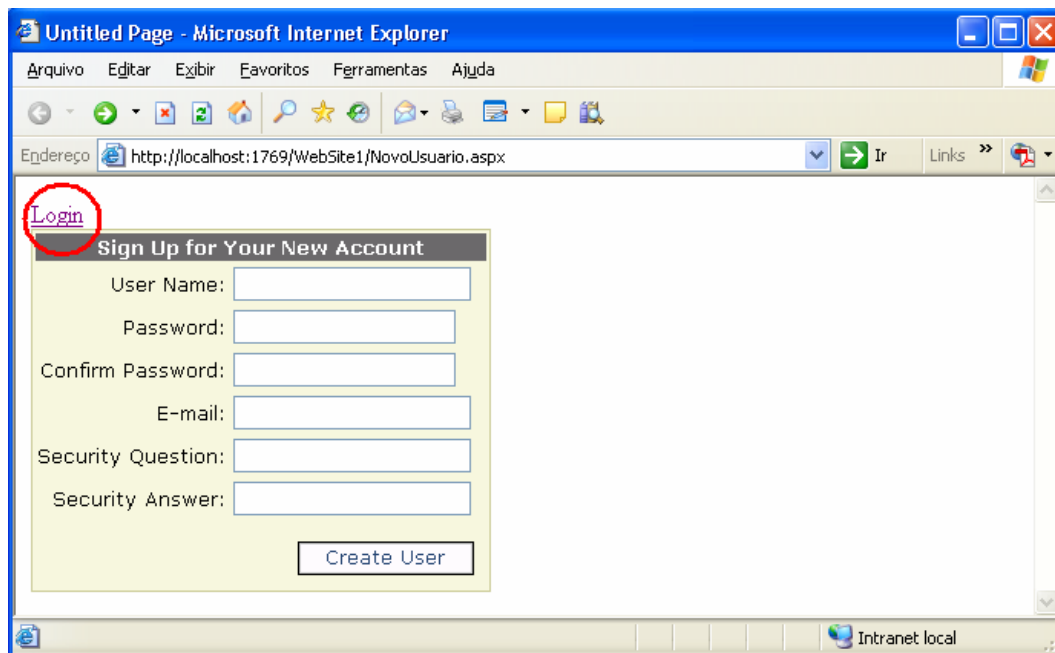
## Criando um atalho para Login ou Logout com LoginStatus

O controle LoginStatus permite criar um atalho para efetuar um login ou Logout do usuário. O controle possui duas visualização: Logged Out, que deve ser definida em situações em que o usuário não se encontra autenticado, e Logged In, quando o usuário já se autenticou no sistema, como você pode ver na imagem abaixo:



Normalmente você vai colocá-lo na visualização Logged Out em áreas publicas e como Logged In em áreas autenticadas. Vamos testá-lo das duas formas.

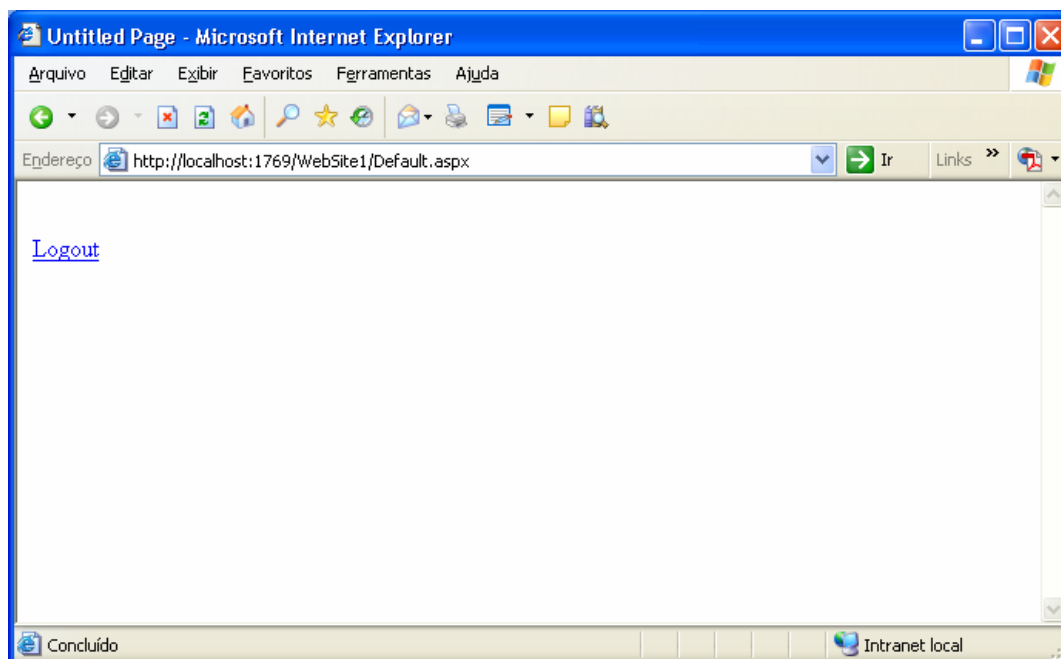
Definimos anteriormente a pagina NovoUsuario.aspx como publica, portanto a mesma poderá ser acessada sem autenticação. Coloque um controle LoginStatus nesta pagina com a visão Logged out, defina esta pagina como página inicial e rode a aplicação:



Ao clicar em Login o usuário é redirecionado para a página de login da aplicação.

Agora vamos testá-lo na visão Logged In. Coloque um controle LoginStatus na página default.aspx, defina sua visão como Logged In e defina esta página como inicial. Ao rodar a aplicação você é direcionado para o Login, pois default.aspx não permite acesso anônimo. Após a autenticação você é redirecionado para a página default, onde o controle pode ser visualizado:

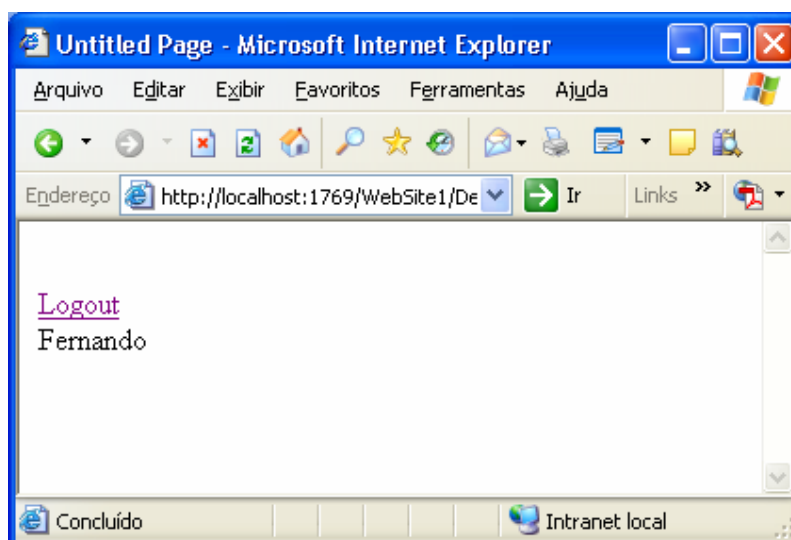




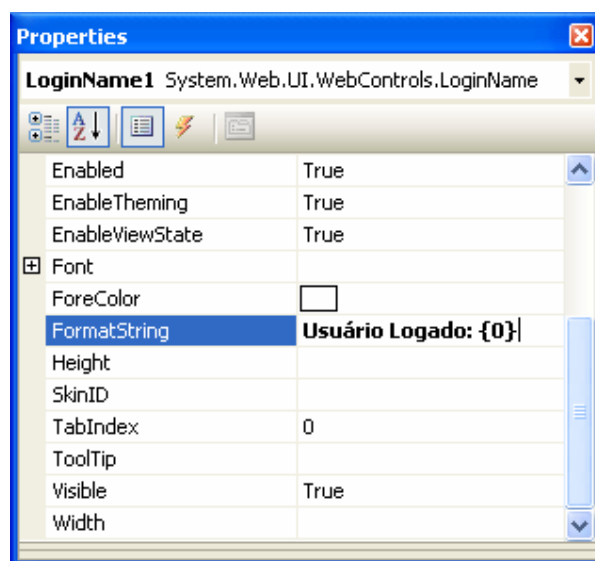
Clicando sobre o controle seu cookie de autenticação é excluído, e você passa a ser um usuário não autenticado, portanto a aplicação o redireciona para a página de login.

## Informando o usuário autenticado com LoginName

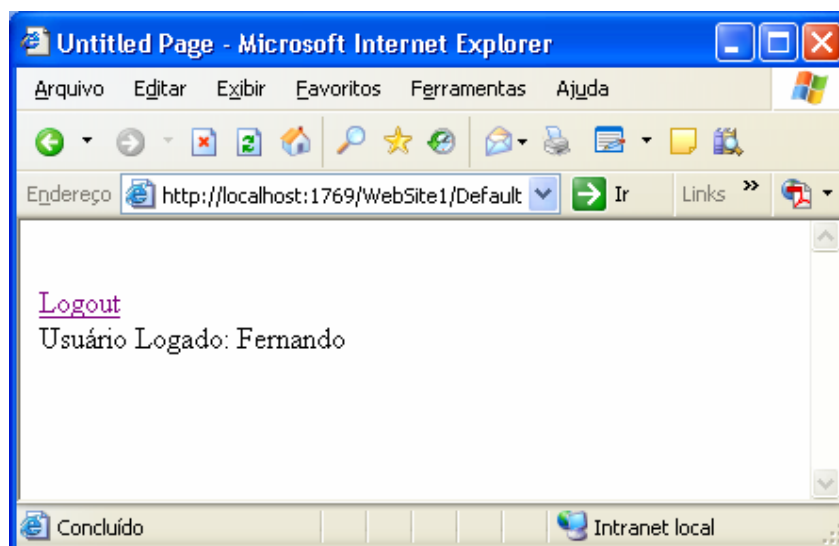
O controle LoginName permite que seja exibido em qualquer local o nome do usuário autenticado. Basta colocá-lo sobre a página. Abaixo você pode ver o mesmo sobre a página default.aspx, acessada obviamente após a autenticação:



Para personalizar a mensagem você pode utilizar a propriedade FormatString, que por padrão esta preenchida como {0}. Para exibir uma mensagem como "Usuário Logado: Fernando" basta alterar a propriedade da seguinte forma:



Observe o resultado em tempo de execução:



Se você colocá-lo em uma página publica e acessá-la sem autenticação, nada será exibido.

## Recuperando senhas com PasswordRecovery

O controle PasswordRecovery permite recuperar senhas esquecidas. Para utilizá-lo basta colocar o controle sobre a pagina. O usuário terá que informar o nome de usuário e responder a pergunta cadastrada durante a criação do usuário. A senha será enviada para o E-mail também cadastrado durante a criação do usuário.

Para testar o controle primeiro você precisa informar um servidor smtp em seu arquivo web.config, a fim de enviar o e-mail, para isso adicione o seguinte nó, logo após o nó location criado anteriormente:

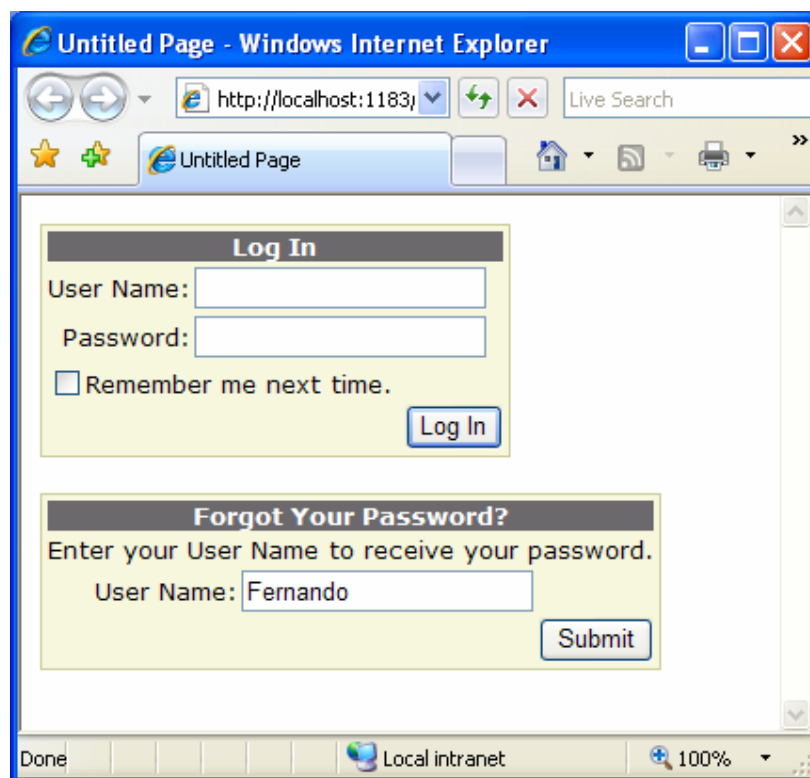
```
<system.net>  
  <mailSettings>
```

```
<smtp from="fulano@portal.com">
  <network
    host="smtp.portal.com"
    port="25"
    userName="user"
    password="123" />
  </smtp>
</mailSettings>
</system.net>
```

Obviamente as informações acima são fictícias, você deve colocar informações de um servidor smtp válido.

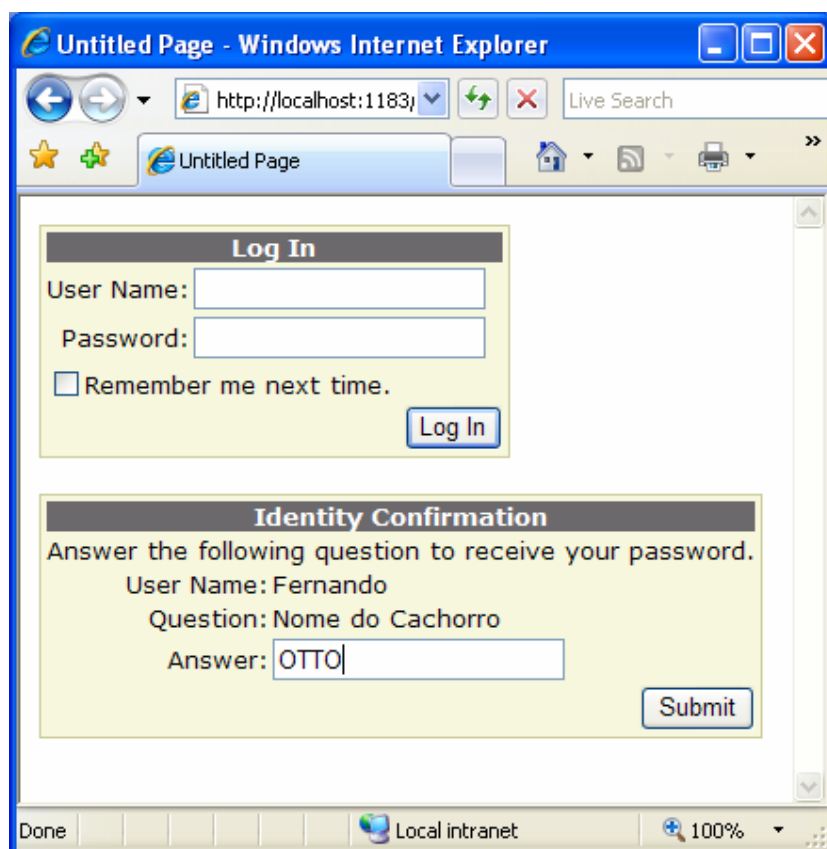
O próximo passo é colocar um controle PasswordRecovery em sua página de login.aspx, que é local mais apropriado por ser onde normalmente o usuário descobre que esqueceu a senha e também por ter acesso publico:

Defina login.aspx como página inicial e rode a aplicação. Para recuperar a senha, primeiramente você vai ter que informar o nome do usuário:



The screenshot shows a Windows Internet Explorer window titled "Untitled Page - Windows Internet Explorer". The address bar shows "http://localhost:1183". The page content includes two main sections: "Log In" and "Forgot Your Password?". The "Log In" section has fields for "User Name:" and "Password:", a checkbox for "Remember me next time.", and a "Log In" button. The "Forgot Your Password?" section has a heading "Forgot Your Password?", a sub-heading "Enter your User Name to receive your password.", a "User Name:" field with the text "Fernando" entered, and a "Submit" button. The status bar at the bottom shows "Done", "Local intranet", and "100%".

Em seguida, deve responder a pergunta secreta cadastrada durante o login:



The screenshot shows a Windows Internet Explorer browser window titled "Untitled Page - Windows Internet Explorer". The address bar displays "http://localhost:1183". The page content consists of two main sections:

**Log In**

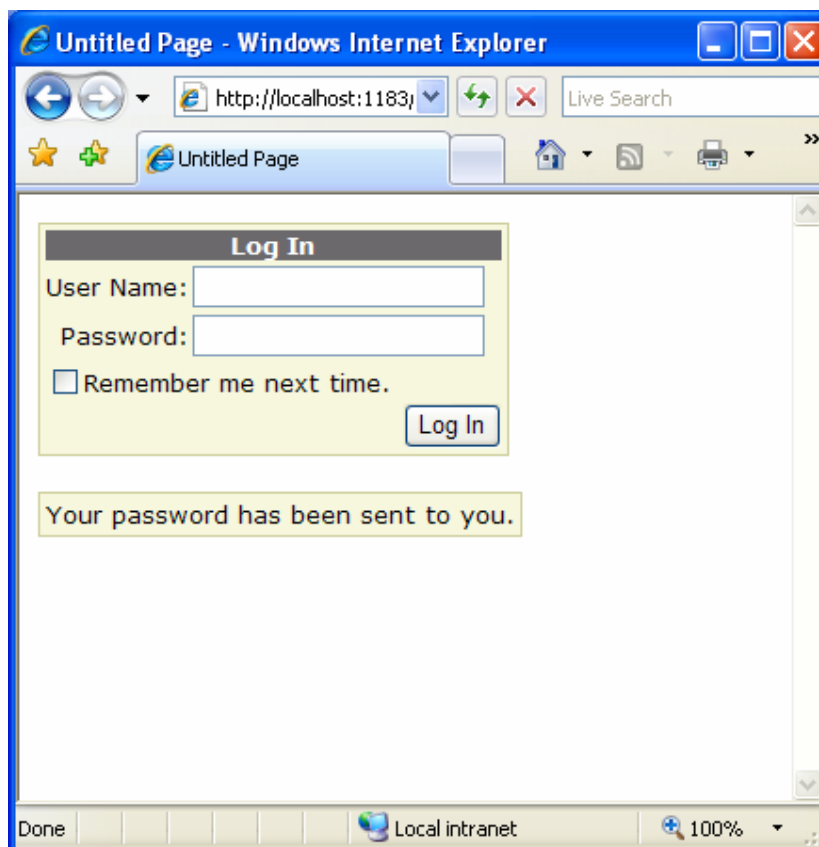
User Name:   
Password:   
☐ Remember me next time.

**Identity Confirmation**

Answer the following question to receive your password.  
User Name: Fernando  
Question: Nome do Cachorro  
Answer:

The status bar at the bottom shows "Done", "Local intranet", and "100%" zoom.

Se tudo correr bem, a aplicação informa que o e-mail foi enviado:



Ok, agora é só abrir o e-mail. Você vai receber uma mensagem semelhante a listada abaixo:

**Please return to the site and log in using the following information. User Name: Fernando Password: rnaIrYqSBLAbjs**

Bom, temos um problema. A senha retornada não é nem parecida com a que você cadastrou. o que deu errado?

O motivo é simples: É uma boa prática armazenar senhas no forma de Hashs. Um hash é um conjunto de caracteres obtidos através da aplicação de um algoritmo de criptografia sobre alguma informação. Um hash tem algumas características peculiares:

- Tem tamanho fixo, ou seja, se você aplicar um hash sobre a bíblia ou sobre o seu nome, o tamanho do hash deverá ser o mesmo.
- Deve ser único, ou seja, nenhum hash gerado a partir de uma informação diferente deve ser igual a outro qualquer.
- É irreversível. Ao contrário da criptografia simétrica em que podem embaralhar alguns dados e fazer o caminho inverso, com um hash não podemos obter a informação original.

O uso de hash é ideal para o armazenamento se senhas. No login, o sistema gera um hash a partir da senha informada e compara com o valor armazenado. A grande vantagem é que a senha original não precisa ser armazenada, corremos menos riscos. A desvantagem é que se o usuário esquecer a senha, a única saída é criar

outra. Foi o que ASP.NET fez, gerou uma nova senha e enviou para o E-mail do usuário.

Opcionalmente você pode alterar este comportamento, para que as senhas sejam armazenadas Criptografadas ou sem qualquer mecanismo de criptografia, porém é recomendável manter o comportamento padrão do ASP.NET.

## **Conteudos para usuários anônimos e autenticados em uma mesma página**

É comum também quereremos exibir para nossos usuários conteudos especificos conforme sua situação na Site. Por exemplo, em uma determinada parte da página, se o usuário esta logado exibimos seu nome de usuário, se não estiver logado, colocamos uma caixa de login.

O controle LoginView tem exatamente este objetivo. Ele é formado de dois templates: Anonymous e LoggedIn. Para testa-lo abra a pagina NovoUsuario.aspx e coloque um LoginView, no template Anonymous coloque um controle Login, e no template LoggedIn coloque um controle LoginName Login.

Defina a página NovoUsuario.aspx como inicial e rode a aplicação. Note que é exibido o conteudo do template Anonymous, pois o usuário ainda não efetuou login:

The screenshot shows a Windows Internet Explorer browser window titled "Untitled Page - Windows Internet Explorer". The address bar displays "http://localhost:1183". The page content includes a "Login" link at the top left. Below it is a "Sign Up for Your New Account" form with fields for "User Name:", "Password:", "Confirm Password:", "E-mail:", "Security Question:", and "Security Answer:", followed by a "Create User" button. Below this is a "Log In" form with fields for "User Name:" and "Password:", a checkbox for "Remember me next time.", and a "Log In" button. The status bar at the bottom shows "Don", "Local intranet", and "100%".

[Login](#)

**Sign Up for Your New Account**

User Name:

Password:

Confirm Password:

E-mail:

Security Question:

Security Answer:

**Log In**

User Name:

Password:

☐ Remember me next time.

Don Local intranet 100%

Agora defina Login.aspx como página inicial, efetue o login e redirecione para NovoUsuario.aspx (digite a URL no navegador). Note que agora é exibido o conteúdo do template LoggedIn:

Logout Fernando

**Sign Up for Your New Account**

User Name:

Password:

Confirm Password:

E-mail:

Security Question:

Security Answer:

Create User

Fernando



## Praticando

Crie uma aplicação web com funcionalidades de Cadastro de Usuários, Login, troca de senha e recuperação de senha. Crie ainda uma página com exibição de conteúdo diferenciado para usuários autenticados e usuários anônimos.

## Gerenciado usuários manualmente

O ASP.NET prove ainda uma API com diversas classes que nos permitem fazer tudo o que foi estudado até o momento manualmente (e muito mais).

Para exemplificar, vamos listar na página Default.aspx todos os usuário do sistema em um ListBox através do método GetAllUser. Para isso, adicione um Listbox a sua pagina e no evento Load da página escreva o seguinte código:

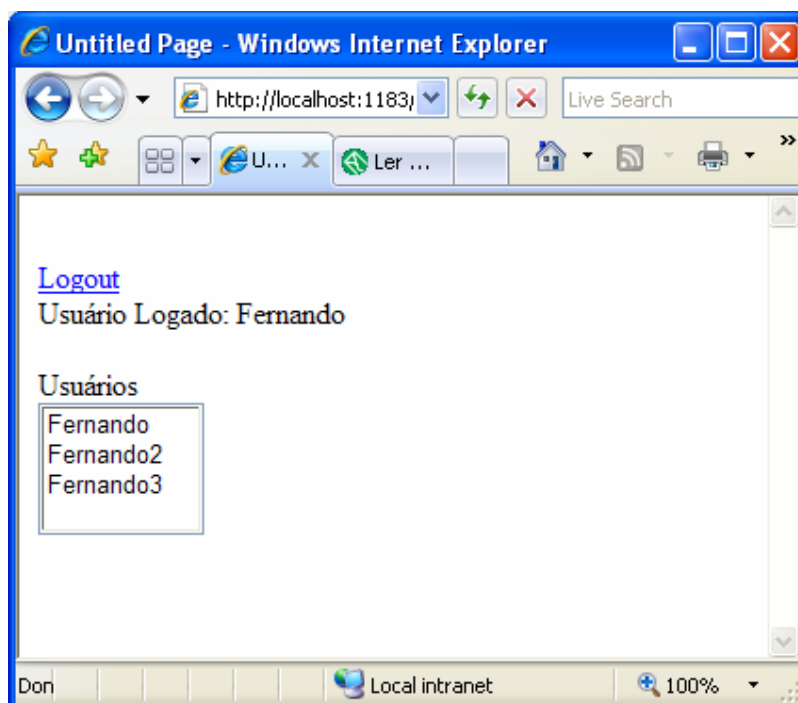
**Vb**

```
If Not Page.IsPostBack Then  
    ListBox1.DataSource = Membership.GetAllUsers
```



```
ListBox1.DataBind()  
End If
```

```
c#  
if (!Page.IsPostBack)  
{  
    ListBox1.DataSource = Membership.GetAllUsers();  
    ListBox1.DataBind();  
}
```



Listar todas as classes com seus métodos e exemplos requer um curso inteiro, por isso vou listar abaixo as principais classes e seus membros mais importantes, com suas respectivas funções:

## Classe Membership

Funções específicas de gerenciamento de usuários.

CreateUser: Permite Criar um novo usuário.

DeleteUser: Permite excluir um usuário.

FindUserByEmail: Retorna uma coleção de usuários com o E-mail especificado.

GeneratePassword: Gera uma senha aleatória.

GetAllUsers: Retorna uma coleção de usuários.

GetUserNameByEmail: Retorna um usuário cujo E-mail equivale ao especificado.

UpdateUser: Atualiza o usuário.

ValidateUser: Verifica se o usuário e sua senha são válidos.

## Classe MembershipUser

Trás informações específicas de um usuário

CreationDate: Data e hora em que o usuário foi criado.

Email: Retorna ou determina o E-mail do usuário.

IsOnLine: Retorna se o usuário esta on-line.

LastActivityDate: Retorna a data e hora da última atividade do usuário

LastLoginDate: Retorna a data do ultimo login do usuário.

## Autorização e Funções

Até agora vimos alguns métodos de gerenciamento de usuários. Porém normalmente usuários têm funções diferentes em um negócio, e sua aplicação deverá ser capaz de gerenciar isto. Embora neste quesito já não existam controles prontos, o gerenciamento de funções é também muito mais fácil e rápido no ASP.NET 2.0.

Para testarmos o gerenciamento e uso de funções vamos executar as seguintes tarefas:

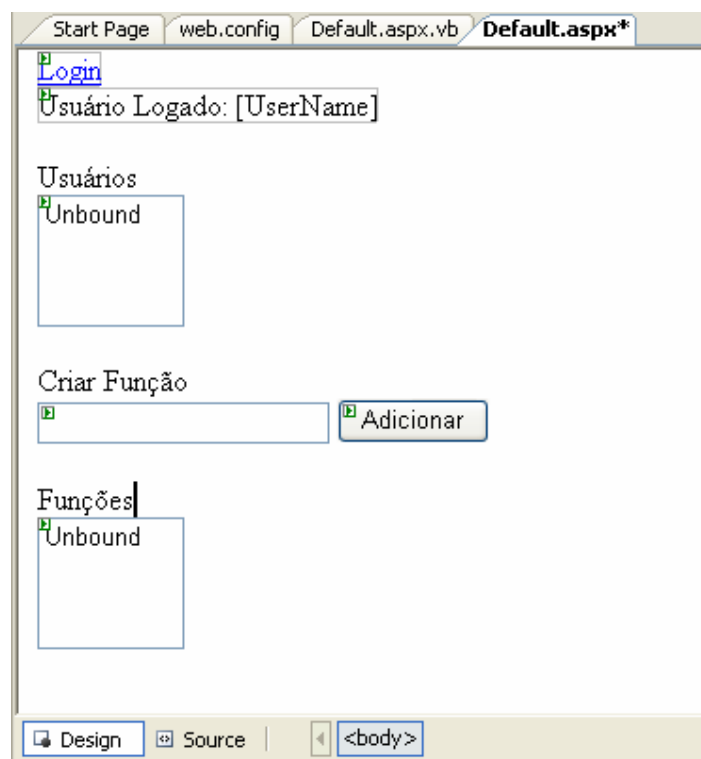
- Criar um cadastro de funções em nossa página Default.aspx. Cadastrar a função admin.
- Criar um controle de atribuição de funções a usuários também na pagina Default. Adicionar seu usuário a esta função.
- Definir que nossa página administracao.aspx só poderá ser acessada por um usuário pertencente a função admin.
- Testar a aplicação.

## Criando um cadastro de Funções

Antes de tudo devemos habilitar o gerenciamento de funções em nossa aplicação, que por padrão é desabilitado. Para isso adicione o seguinte nó dentro de <system.web>:

```
<roleManager enabled="True" />
```

Agora adicione a página Default.aspx um controle textbox com um rótulo Grupo e um button com um rótulo adicionar logo ao lado. Adicione também um novo controle ListBox (já deve existir um para exibição de usuários) com o rótulo Funções. Sua página, em tempo de design, deve se parecer com a imagem abaixo.



Agora vamos adicionar algumas linhas de código. Altere seu método Load da pagina de acordo com a listagem abaixo:

```
VB  
If Not Page.IsPostBack Then  
    ListBox1.DataSource = Membership.GetAllUsers  
    ListBox1.DataBind()  
    Roles_DataBind()  
End If
```

```
C#  
if (!Page.IsPostBack)  
{  
    ListBox1.DataSource = Membership.GetAllUsers();  
    ListBox1.DataBind();  
    Roles_DataBind();  
}
```

Crie um método Roles\_dataBind, que atualiza as funções no listbox:

```
VB  
Sub Roles_DataBind()  
    ListBox2.DataSource = Roles.GetAllRoles  
    ListBox2.DataBind()  
End Sub
```

```
C#  
void Roles_DataBind()  
{  
    ListBox2.DataSource = Roles.GetAllRoles();  
    ListBox2.DataBind();  
}
```

Agora de um duplo clique sobre o botão Incluir, e sobre o manipulador de evento criado adicione o seguinte código:

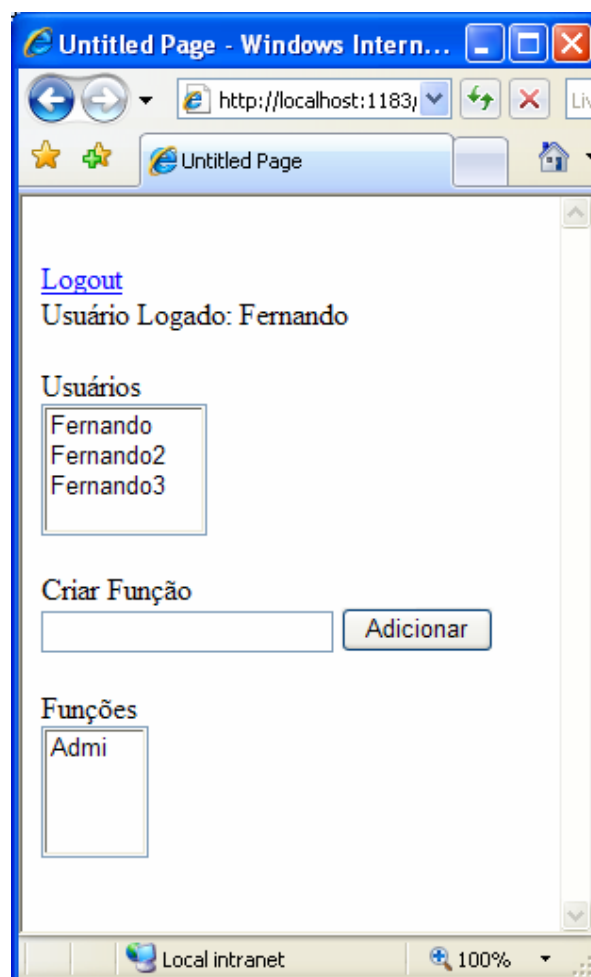
**Vb**

```
Roles.CreateRole(TextBox1.Text)  
Roles_DataBind()
```

**C#**

```
Roles.CreateRole(TextBox1.Text);  
Roles_DataBind();
```

Tudo o que foi feito é adicionar uma nova função a partir do texto informado no controle textbox. Rode a aplicação, informe admin como nome para a função clique em Incluir:

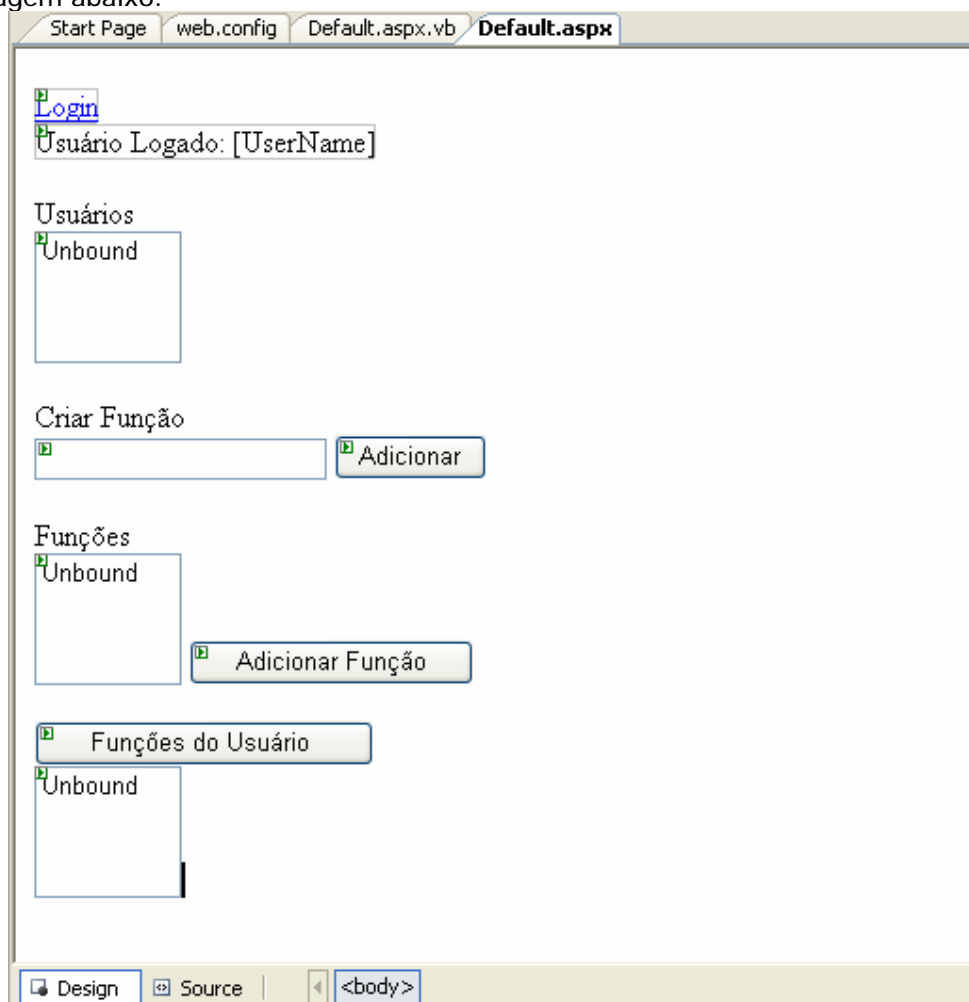


## Criando um controle de atribuição de funções a usuários

Para adicionar uma função a um usuário, vamos verificar o usuário selecionado, a função seleciona e em seguida fazer a adição, tudo através do clique de um botão Adicionar Função. Para exibir a função de um determinado usuário, um clique em

outro botão Funções do Usuário deveria exibir em um listbox as funções atribuídas ao usuário selecionado.

Primeiramente adicione ao seu WebForm default.aspx um botão de texto Adicionar Função, um botão de texto Funções do Usuário ao lado do listbox de usuários, e um novo listbox abaixo deste botão. Seu formulário agora deverá se parecer com a imagem abaixo:



Agora um pouco de código. No manipulador do evento do botão Adicionar Função coloque:

**VB**

```
Roles.AddUserToRole(ListBox1.SelectedItem.Text, _  
    ListBox2.SelectedItem.Text)
```

**C#**

```
Roles.AddUserToRole(ListBox1.SelectedItem.Text, _  
    ListBox2.SelectedItem.Text);
```

Já para o botão Funções do usuário:

**VB**

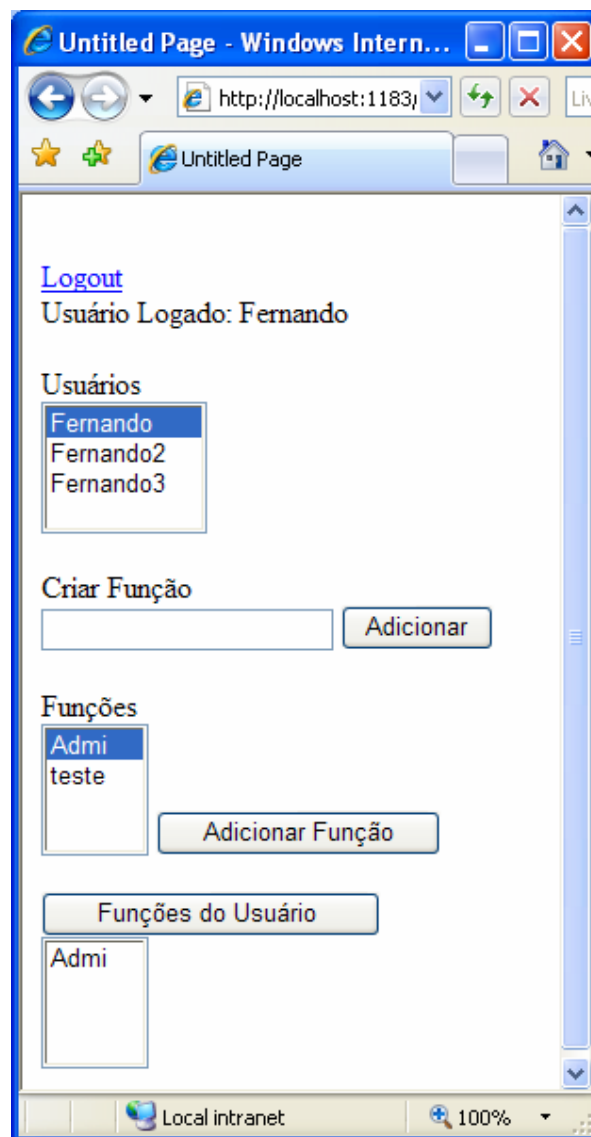
```
ListBox3.DataSource =  
Roles.GetRolesForUser(ListBox1.SelectedItem.Text)  
ListBox3.DataBind()
```

c#

```
ListBox3.DataSource =  
Roles.GetRolesForUser(ListBox1.SelectedItem.Text);  
ListBox3.DataBind();
```

O que o código acima faz é simplesmente pegar as funções do atribuídas ao usuário selecionado no Listbox1 e listadas no listbox3.

Rode a aplicação e adicione a função admin ao seu usuário:



## Definindo que nossa página Administracao.aspx só poderá ser acessada por um usuário pertencente a função admin

Para que apenas usuários do grupo admin possam acessar a página Administracao.aspx, tudo a fazer é adicionar um novo nó location em nosso web.config, da seguinte forma:

```
<location path="Administracao.aspx">
  <system.web>
    <authorization>
      <allow roles="admin"/>
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

Observe que a estrutura do nó é semelhante a utilizada anteriormente para tornar a página NovoUsuario.aspx, porém aqui ao invés de autorizarmos um usuário estamos autorizando uma função, e negando acesso a todos os demais usuários. Para testar crie um novo usuário e não o adicione a função admin, faça o login com o mesmo e tente acessar a página Administracao.aspx. Faça o mesmo com seu usuário que foi incluído na função admin.

## Arquivo web.config

Abaixo segue o arquivo web.config final, utilizado em nossos exemplos até aqui. Apenas comentários foram removidos por praticidade.

```
<?xml version="1.0"?>
<configuration
  xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <appSettings/>
  <connectionStrings/>
  <system.web>
    <compilation debug="true" strict="false" explicit="true"/>
    <pages>
      <namespaces>
        <clear/>
        <add namespace="System" />
        <add namespace="System.Collections" />
        <add
          namespace="System.Collections.Specialized" />
        <add namespace="System.Configuration" />
        <add namespace="System.Text" />
        <add
          namespace="System.Text.RegularExpressions" />
        <add namespace="System.Web" />
        <add namespace="System.Web.Caching" />
        <add namespace="System.Web.SessionState" />
        <add namespace="System.Web.Security" />
        <add namespace="System.Web.Profile" />
        <add namespace="System.Web.UI" />
        <add namespace="System.Web.UI.WebControls" />
        <add
          namespace="System.Web.UI.WebControls.WebParts" />
        <add namespace="System.Web.UI.HtmlControls" />
      </namespaces>
    </pages>
```

```
<roleManager enabled="True"/>
  <authentication mode="Forms">
    <forms name=".ASPXAUTH" loginUrl="login.aspx"
defaultUrl="default.aspx">
    </forms>
  </authentication>
  <authorization>
    <deny users="?" />
  </authorization>
</system.web>
<location path="Administracao.aspx">
  <system.web>
    <authorization>
      <allow roles="admin" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
<location path="NovoUsuario.aspx">
  <system.web>
    <authorization>
      <allow users="*" />
    </authorization>
  </system.web>
</location>
<system.net>
  <mailSettings>
    <smtp from="fulano@portal.com">
      <network
        host="smtp.portal.com"
        port="25"
        userName="user"
        password="123" />
      </smtp>
    </mailSettings>
  </system.net>
</configuration>
```



## Praticando

Crie uma aplicação ASP.NET com três páginas: Publica.aspx, Funcionario.aspx e Gerente.aspx. A página publica poderá ser acessada por qualquer usuário. A página Publica.aspx deverá ter seu acesso irrestrito a qualquer usuário. A página Funcionario.aspx deverá ser acessada apenas por usuários do grupo funcionário e do grupo gerentes. A página do grupo gerentes deverá ser acessada apenas por gerentes. Crie todas as demais páginas para criação e gerenciamento de usuários e grupos, bem como de login.