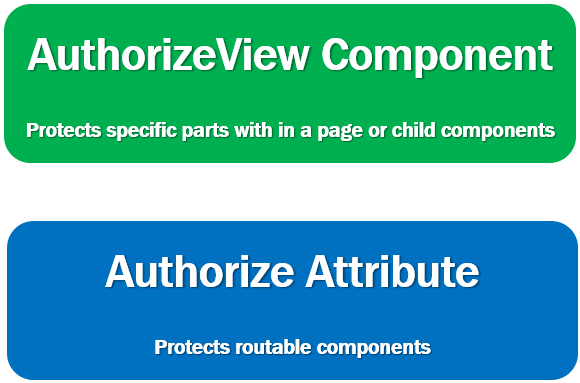
Blazor authorize attribute

In our previous video we discussed AuthorizeView component. This component is used to authorize the display of specific parts within a page, including child components.



We use [Authorize] attribute to protect routable components (i.e components with @page directive). We reach these components via the router and authorization is performed while being routed to these components.

So this means, [Authorize] attribute cannot be used to protect parts of a page or child components. For that we use AuthorizeView component and for routable components we use [Authorize] attribute.

Authorize attribute in Blazor and AuthorizeRouteView component

In the following example, [Authorize] attribute is used in it's simplest form, without any parameters (i.e roles or policies), so, it only checks if the user is authenticated.

If the user is authenticated, then the authorization is granted otherwise not.

@page "/"

@inherits EmployeeListBase

@attribute [Authorize]

@\*Rest of the component code\*@

Copy

Blazor authorize attribute not working

* For the authorization to work as expected, along with the [Authorize] attribute we must also use AuthorizeRouteView component instead of RouteView component.
* In App.razor file, use AuthorizeRouteView component instead of RouteView component.

<AuthorizeRouteView RouteData="@routeData" DefaultLayout="@typeof(MainLayout)" />

Copy

Role-based or Policy-based authorization

The [Authorize] attribute also supports role-based and policy-based authorization. For role-based authorization, use the Roles parameter:

@page "/"

@attribute [Authorize(Roles = "administrator, manager")]

<p>Only users in administrator or manager role are allowed access</p>

Copy

For policy-based authorization, use the Policy parameter:

@page "/"

@attribute [Authorize(Policy = "admin-policy")]

<p>Only users who satisfy admin-policy are allowed access</p>

Copy

Customize unauthorized content

If access is not allowed, "Not authorized" is the content that is displayed by default. However, we can customize this unauthorized content.

<CascadingAuthenticationState>

<Router AppAssembly="@typeof(Program).Assembly">

<Found Context="routeData">

<AuthorizeRouteView RouteData="@routeData" DefaultLayout="@typeof(MainLayout)">

<NotAuthorized>

<p>Sorry, you're not authorized to reach this page.</p>

<p>You may need to log in as a different user.</p>

</NotAuthorized>

</AuthorizeRouteView>

</Found>

<NotFound>

<LayoutView Layout="@typeof(MainLayout)">

<p>Sorry, there's nothing at this address.</p>

</LayoutView>

</NotFound>

</Router>

</CascadingAuthenticationState>