

# Lighting Up Real-time Web Communications with SignalR



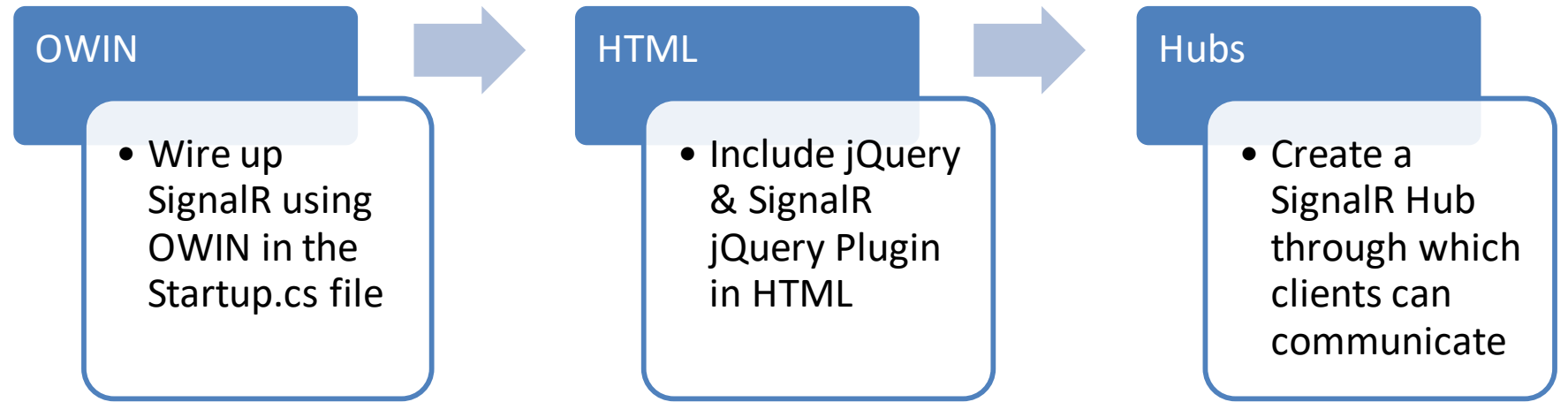
Jon Galloway | Technical Evangelist  
Brady Gaster | Program Manager, Azure SDK & Tools

## 02 | SignalR on the Web

Jon Galloway | Technical Evangelist  
Brady Gaster | Program Manager, Azure SDK & Tools

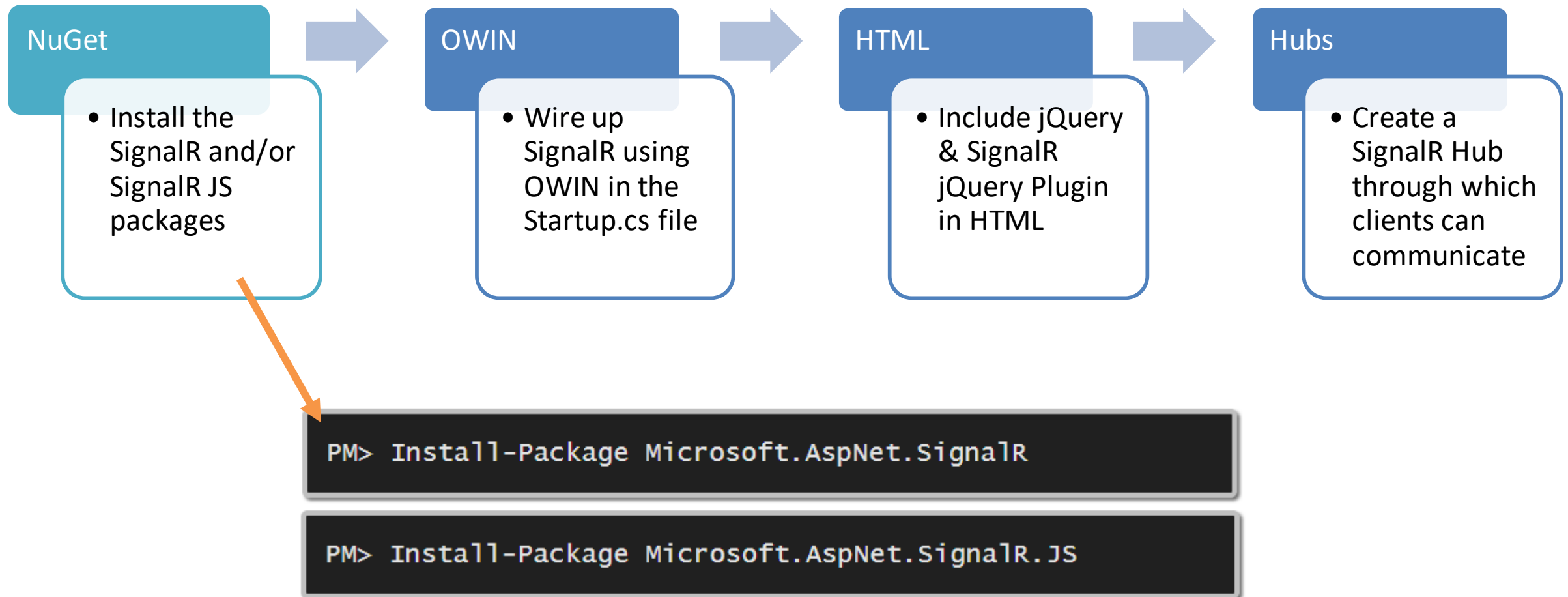
# Using SignalR in **New** ASP.NET Web Applications

SignalR is included in the Visual Studio 2013 ASP.NET templates

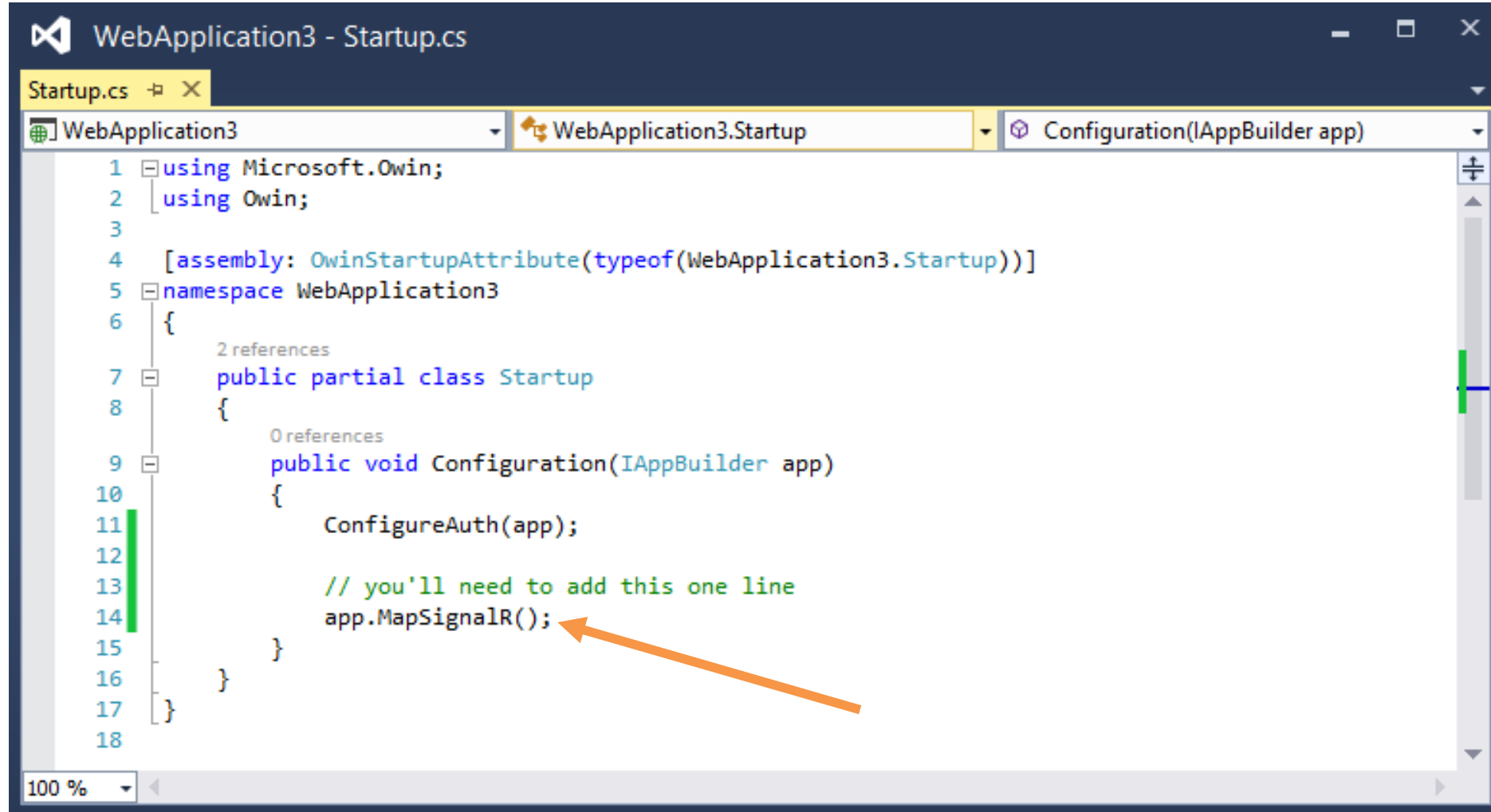


# Using SignalR in Existing ASP.NET Web Applications

SignalR is distributed as numerous NuGet packages, one of which provides the JavaScript client

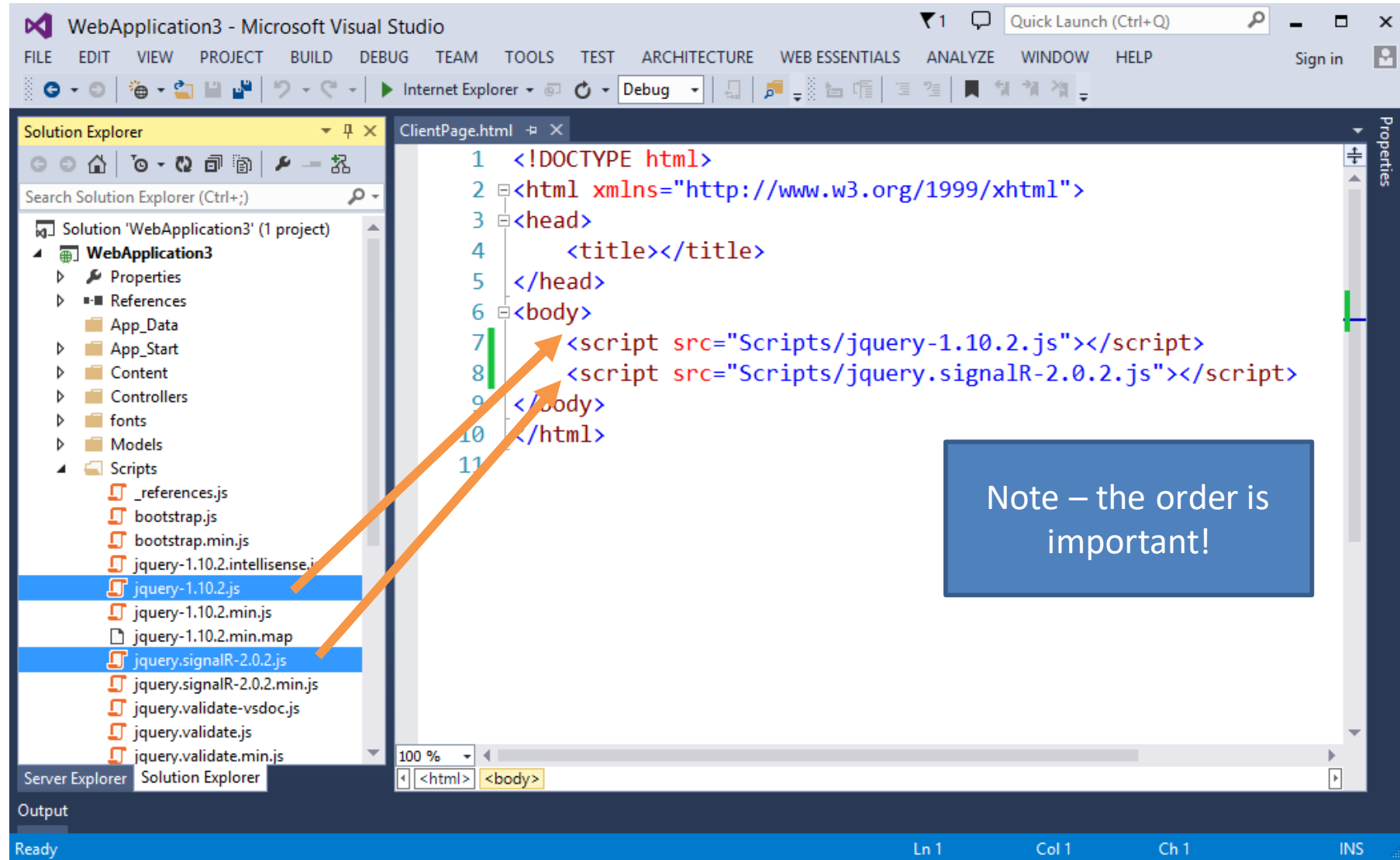


# Wire up SignalR using OWIN

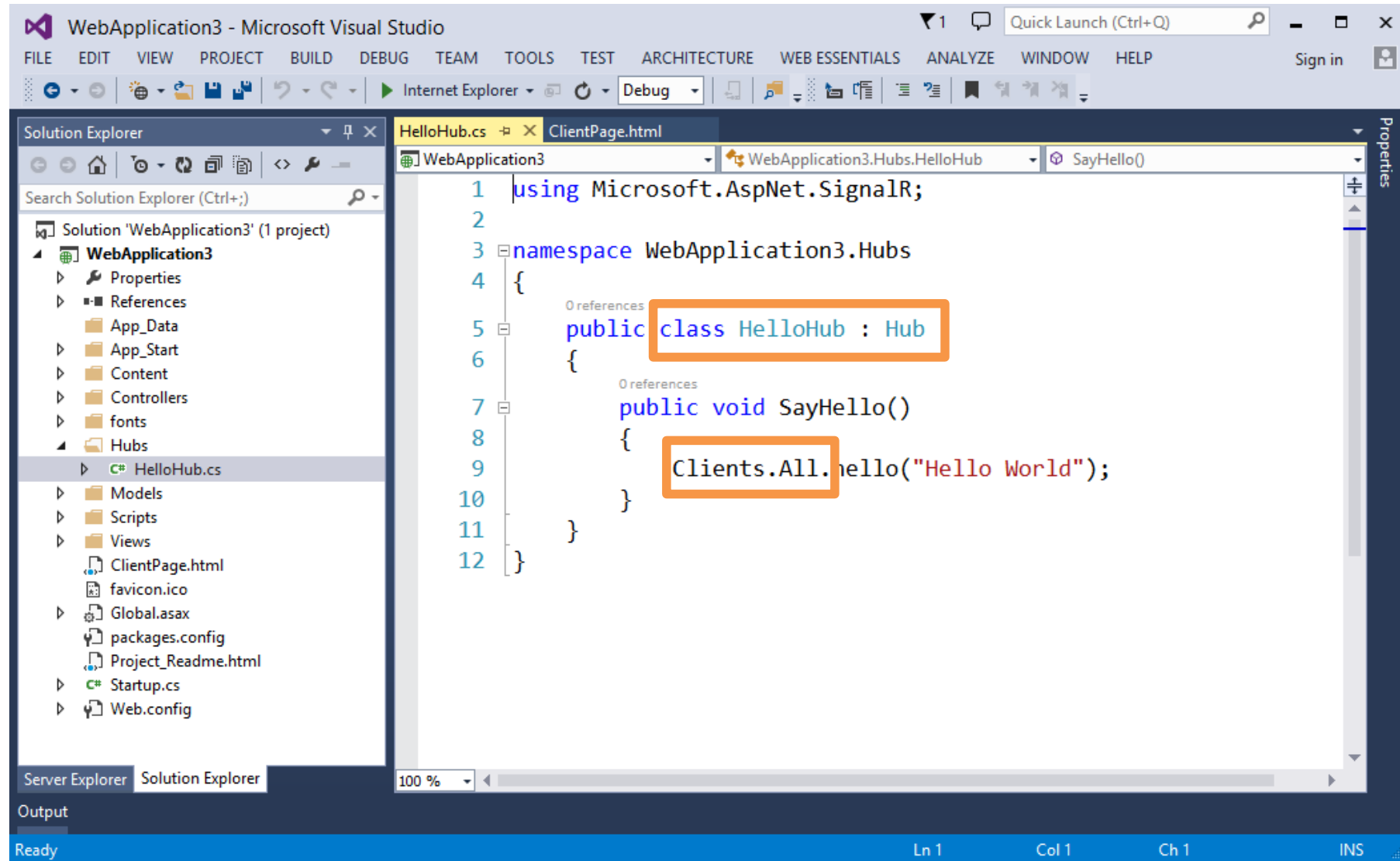


```
1 using Microsoft.Owin;
2 using Owin;
3
4 [assembly: OwinStartupAttribute(typeof(WebApplication3.Startup))]
5 namespace WebApplication3
6 {
7     2 references
8     public partial class Startup
9     {
10         0 references
11         public void Configuration(IAppBuilder app)
12         {
13             ConfigureAuth(app);
14
15             // you'll need to add this one line
16             app.MapSignalR();
17         }
18     }
19 }
```

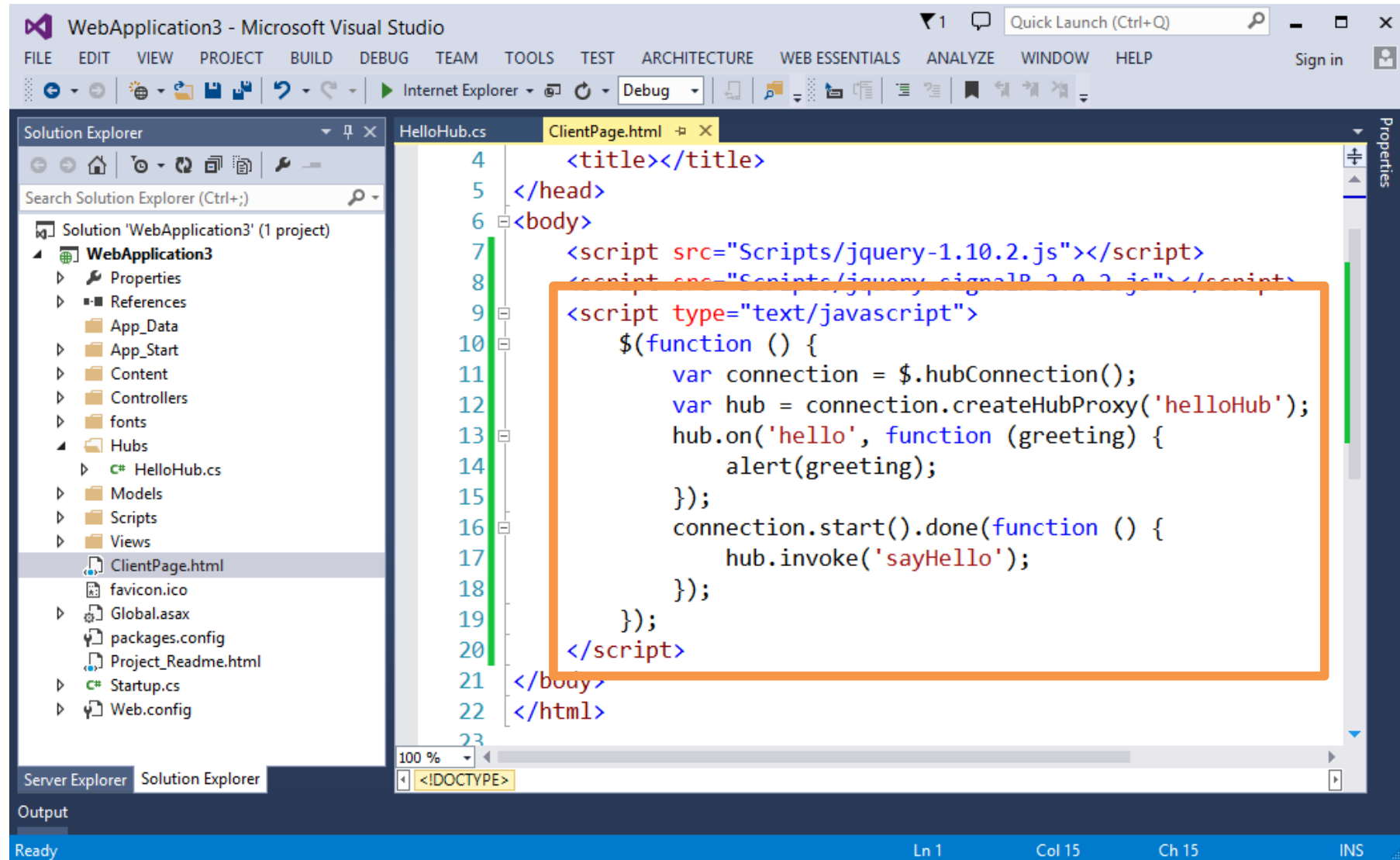
# Include the jQuery Scripts



# Create a Hub



# Connect the client to the Hub using JavaScript





# DEMO

---

Installing SignalR into an Existing ASP.NET App

# Using SignalR with AngularJS

Jon Galloway | Technical Evangelist  
Brady Gaster | Program Manager, Azure SDK & Tools

# Connect the client to the Hub using JavaScript

- Hubs are wrapped by Angular factories
- Multiple controllers can make use of a single factory
- Angular factories are created as singletons
- All controllers can share single-instance Hub proxies

# DEMO

---

Using SignalR and AngularJS Together

# DEMO

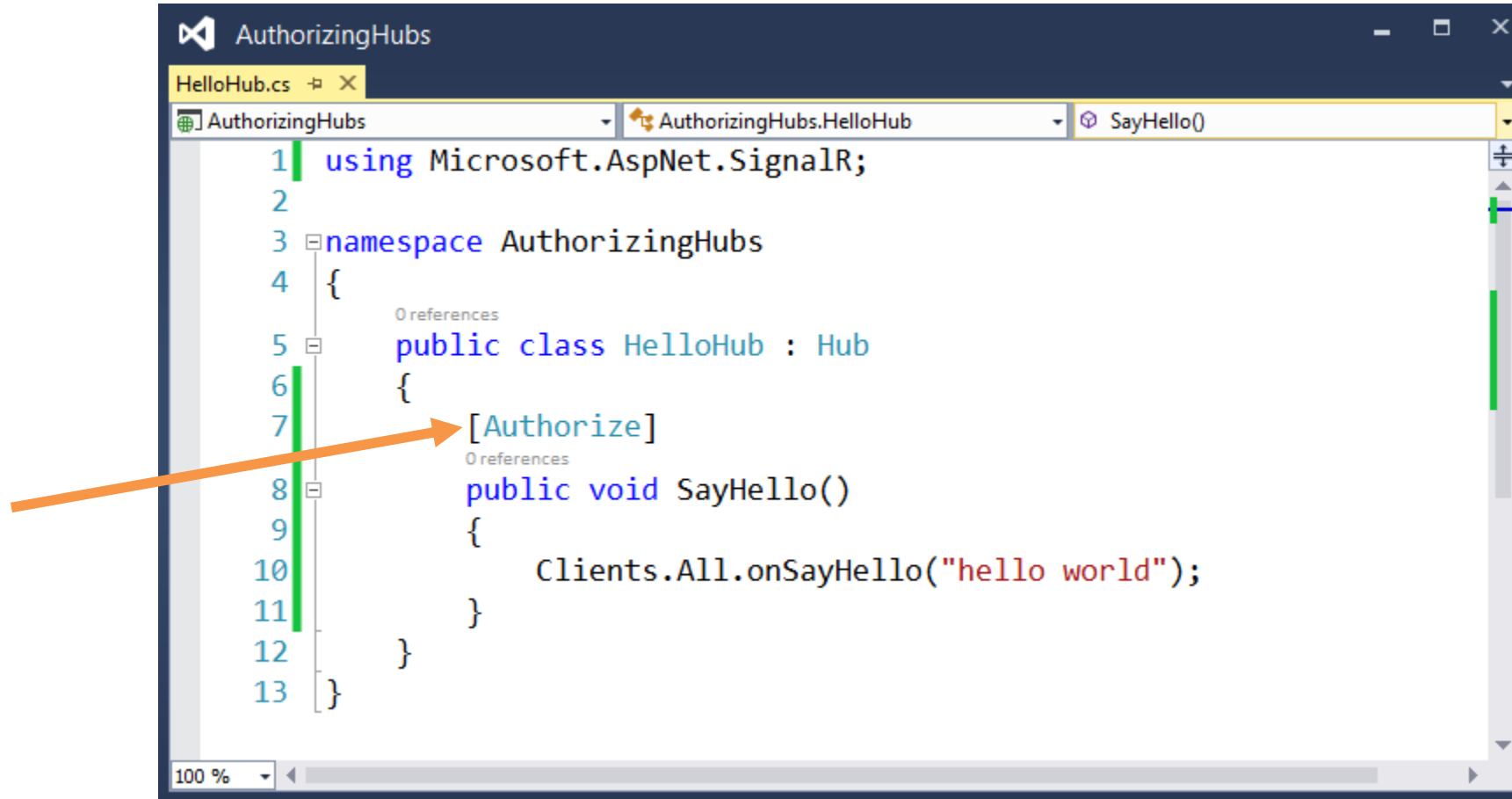
---

Building a Real-world Angular/SignalR App

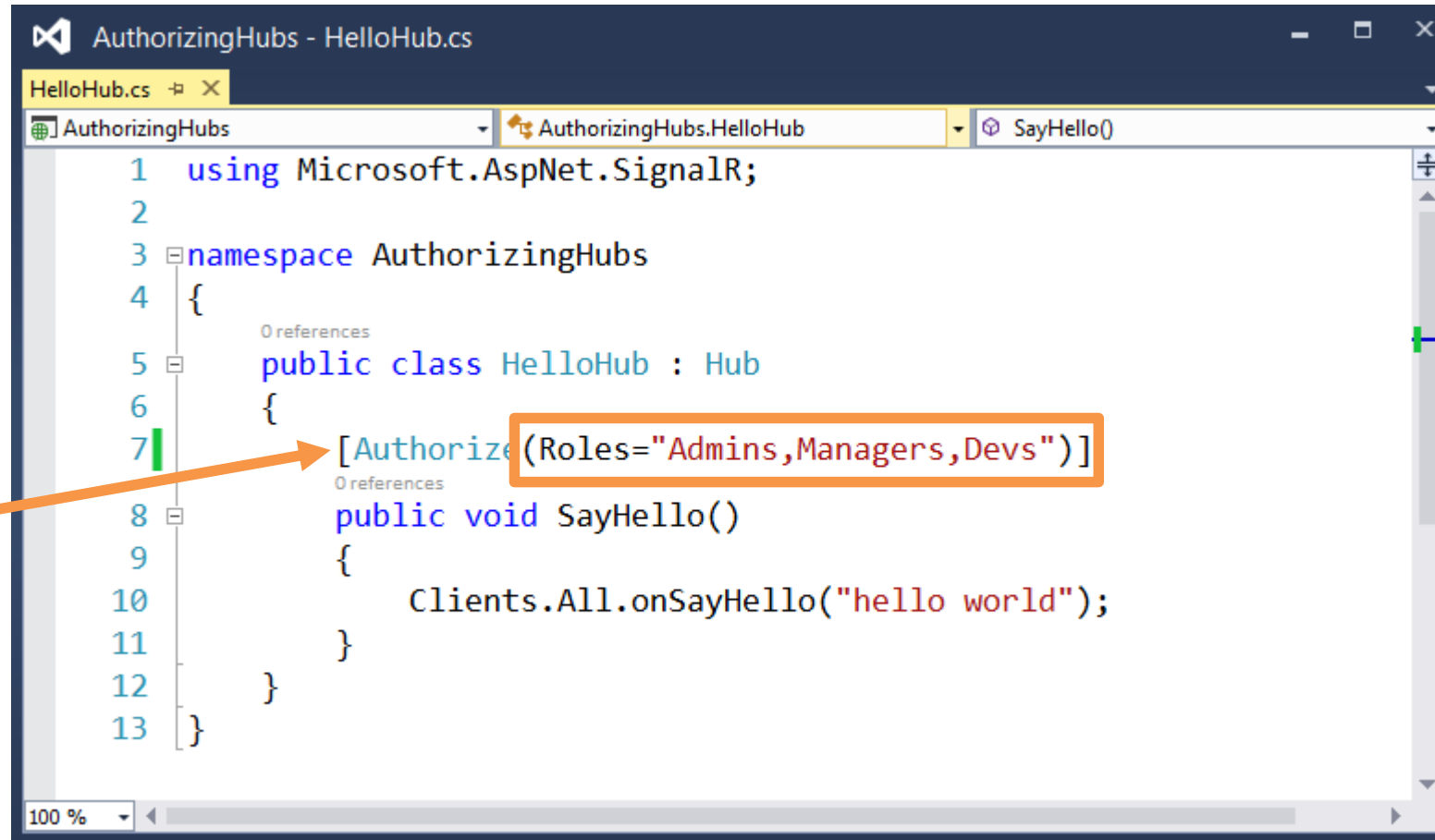
# Enabling Authorization with SignalR

Jon Galloway | Technical Evangelist  
Brady Gaster | Program Manager, Azure SDK & Tools

# Allowing Authorized Users Only



# Controlling Access by Role



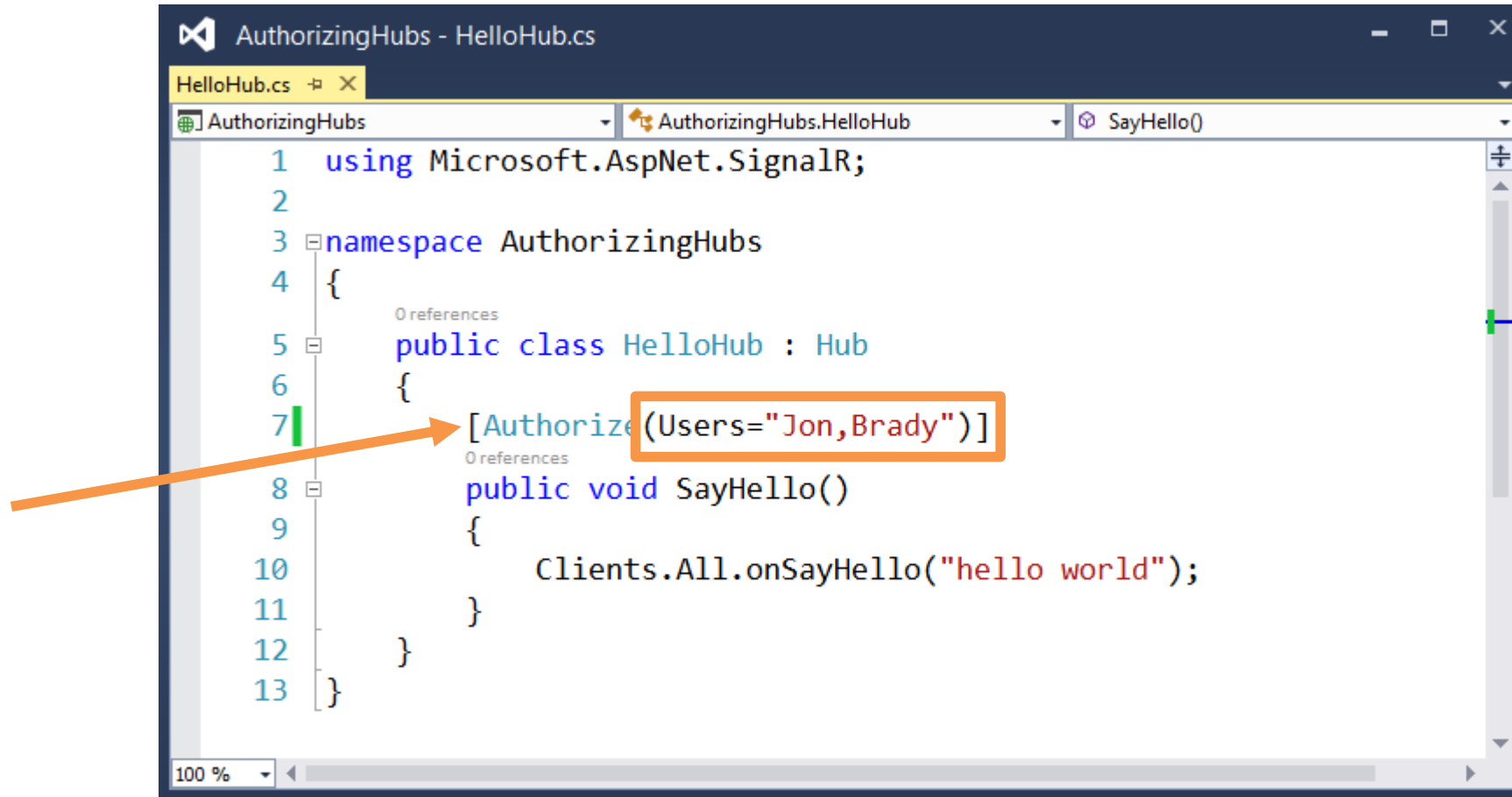
The screenshot shows a Visual Studio window titled "AuthorizingHubs - HelloHub.cs". The code editor displays the following C# code:

```
1 using Microsoft.AspNet.SignalR;  
2  
3 namespace AuthorizingHubs  
4 {  
5     public class HelloHub : Hub  
6     {  
7         [Authorize(Roles="Admins,Managers,Devs")]  
8         public void SayHello()  
9         {  
10             Clients.All.onSayHello("hello world");  
11         }  
12     }  
13 }
```

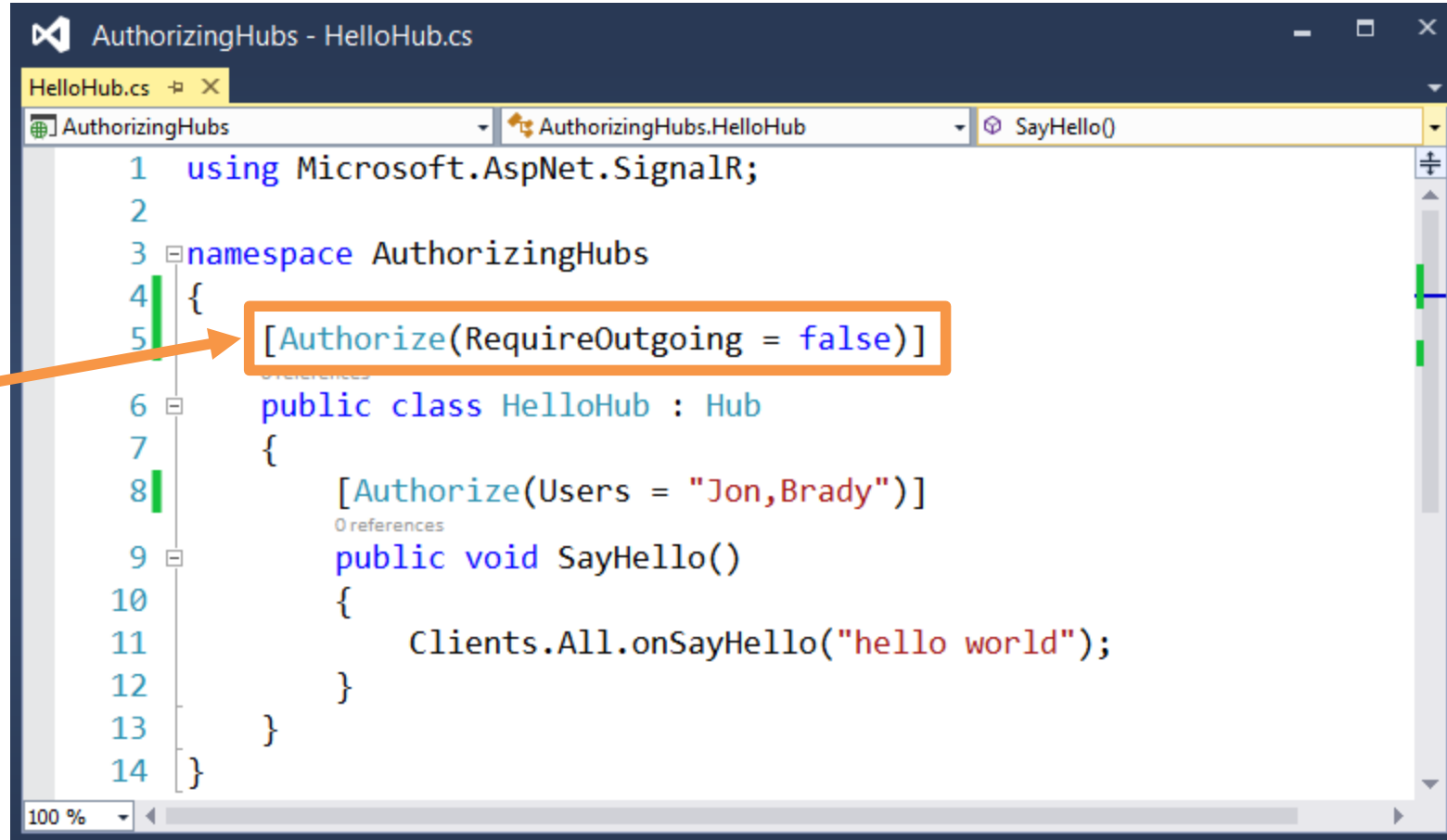
An orange arrow points from the left margin to the `[Authorize(Roles="Admins,Managers,Devs")]` attribute on line 7. The attribute is enclosed in an orange rectangular box. The code is color-coded: keywords are blue, strings are red, and comments are green. The interface includes a breadcrumb trail at the top: "AuthorizingHubs" > "AuthorizingHubs.HelloHub" > "SayHello()". The status bar at the bottom shows "100 %".



# Controlling Access to Specific Users



# Inbound is restricted, but outbound is open



```
1 using Microsoft.AspNetCore.SignalR;
2
3 namespace AuthorizingHubs
4 {
5     [Authorize(RequireOutgoing = false)]
6     public class HelloHub : Hub
7     {
8         [Authorize(Users = "Jon,Brady")]
9         public void SayHello()
10         {
11             Clients.All.onSayHello("hello world");
12         }
13     }
14 }
```

The screenshot shows a Visual Studio window titled "AuthorizingHubs - HelloHub.cs". The code editor displays the following C# code:   
1 using Microsoft.AspNetCore.SignalR;   
2   
3 namespace AuthorizingHubs   
4 {   
5 [Authorize(RequireOutgoing = false)]   
6 public class HelloHub : Hub   
7 {   
8 [Authorize(Users = "Jon,Brady")]   
9 public void SayHello()   
10 {   
11 Clients.All.onSayHello("hello world");   
12 }   
13 }   
14 }   
An orange arrow points from the left margin to the attribute [Authorize(RequireOutgoing = false)] on line 5, which is also enclosed in an orange rectangular box. The breadcrumb at the top shows the path: AuthorizingHubs > AuthorizingHubs.HelloHub > SayHello(). The status bar at the bottom indicates 100% zoom.

# DEMO

---

Requiring Authorization to Access Hubs

Let's *all* participate in an  
*interactive* demo together

# DEMO

---

From Where are You Watching?

# Calling SignalR from ASP.NET Server-side Code

```
MapR
LocationController.cs
MapR
MapR.Controllers.LocationController
Post(Location location)

0 references
7 public class LocationController : ApiController
8 {
9     [HttpPost]
10    public void Post([FromBody] Location location)
11    {
12        var mappingHub = GlobalHost.ConnectionManager.GetHubContext<MappingHub>();
13
14        var hubEventParameters = new
15        {
16            Latitude = location.Latitude,
17            Longitude = location.Longitude
18        };
19
20        mappingHub.Clients.All.locationReceived(hubEventParameters);
21    }
22 }
```

# Resources

Title	URL
Authentication & Authorization for SignalR Hubs	<a href="http://aka.ms/signalr-auth">http://aka.ms/signalr-auth</a>

## Next Section: SignalR on the Client

Jon Galloway | Technical Evangelist  
Brady Gaster | Program Manager, Azure SDK & Tools





# Microsoft

© 2013 Microsoft Corporation. All rights reserved. Microsoft, Windows, Office, Azure, System Center, Dynamics and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries. The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.