**Demo Script**

1. **Introduction**
2. Welcome to DevConnections
3. Today we are going to be covering:
   1. TODO: Fill in this summary
4. My name is Sayed and I'm a PM on the ASP.NET team
5. **Template updates for Web Forms**
6. File New project, pick Web Forms and no other options
7. With **One ASP.NET** every project is equivalent, we can really mix and match
8. You can add MVC/Web API/Web Forms easily to during project create
9. I'll also show you how to add MVC to an existing Web Forms project in a minute
10. **Auth Overview**
11. On the One ASP.NET dialog you can configure the Auth options for the project that will be created. By default Individual Auth is used, which I'll use for this demo
12. **Azure Updates**
13. Mention the Host in the Cloud checkbox, and that it's really easy to dev/test with MA now
14. Create the project
15. CTRL+F5 to see the app running
16. **Responsive Design**
17. The templates use Bootstrap by default now. One of the benefits is responsive design
18. Show the site full screen and then re-size to show responsive ness
19. Show the result in the **Win Phone emulator**? Note: its slow
20. With Bootstrap we can update the theme very easily
21. Go to bootswatch.com and download updated version of bootstrap.css and then refresh the browser to see the updates
22. **Auth demo**
23. CTRL+F5 to get the site running
24. Register a new user
25. It's really easy to add additonal providers such as google, facebook, microsoft account. I'll show you how to add google as a provider to your site
26. Go to startup.cs and uncomment the google section
27. CTRL+F5 to get the project up and running
28. Go to user details and add google auth to the existing user
29. Mention that you can add other providers as well
30. **Bundling**
31. Web Forms templates now enable bundling by default
32. If you've created a web forms project in VS2013 you may have noticed that under Scripts\ there is a WebForms\ folder. Instead of serving Web Forms scripts from scriptresource.axd we now serve directly from the site contents. This is to enable bundling.
33. Show bundle.cs and describe how bundles are declared
34. Show sitemaster to see how bundle is added to the page
35. CTRL+F5 and view source to see the result
36. Change web.config and set Debug="false"
37. CTRL+F5 and view source to see the result, now we are using .min files
38. Go back to sitemaster.aspx and EnableCDN support via attribute
39. CTRL+F5 to see the result again and notice that jQuery is being served from CDN
40. Show the CDN fallback
41. **WebForms Scaffolding**
42. We are going to use scaffolding to help us get started with an application to help us manage contacts. Let's get started
43. Right click on Models folder and Add New Class
44. Add Contact model
45. Build
46. Add New Scaffolded Item and select Web Forms as the node
47. Create new pages
48. Let's take a look at what was created for us
    1. DynamicData templates under DynamicData\
    2. Contact\ folder with pages
49. **Async**
50. Now we are going to talk a bit about asnyc
51. Why would you want to do asnyc?
52. For the web async is all about throughput. If your thread is waiting for the results of a DB call it cannot serve other requests. With async we can free up those threads and serve other requests while the DB returns the result.
53. **History** of asnyc (very brief)
54. We've been working on async for quite some time now. We've had the following major features related to async: APM, get list from **aspconf click video**
55. I'm happy to say that we've finally nailed async in .NET!
56. I'm going to show you two techniques to rip off content from other sites
57. **Web scrapping**
58. Let's create a simple http handler that will scrape the contents of a given site and return it
59. In the web project, Add a new **Generic Handler**
60. I've already got the synchronous version as a snippet let me show it to you
61. Drop in the snippet to implement the generic handler
    1. Snippet should have code for twitter.com and bing.com
62. Talk through the code quickly and CTRL+F5 to see the result
63. Now lets create an async version
64. Remove the code from the handler and extend HttpTaskAsyncHandler instead of implementing IHttpHandler
65. Replace the code with the async version
66. Bring in a snippet for the FlushBufferAsync() method
67. **Image downloader**
68. OK I've just shown how you can scrape a website to get the HTML result now let's take a look at how we can grab images from another location. First I'll show how you might have done this in the past, and then show the async version.
69. Let's get started on that
70. Add new Web Form->ShowImages
71. In ShowImages.aspx add a asp:ListView with an Item Template to show all the images
72. Go into showimages.cs and create a method to return the images from the images\ folder.
73. CTRL+F5 to see the result and that its working
74. OK now we are going to add the part where we can add new images to the site
75. Add New Web Form->**AddImages**
76. Add a TextBox with TextMode = Multiline and set the default value to the dropbox link
77. Add code to addimages.cs to download the images
78. CTRL+F5
79. Add an image
80. Go to showimages to see the result
81. Go back to addimages, snap brower to left
82. Snap windows explorer to the right and open the images folder
83. In browser add 5 images to the download list and hit download
84. As you can see each image is being downloaded and saved before starting the process of downlading the next image
85. Let's see how we can improve this with async
86. The first thing I'm going to do is to add the Async="true" attribute to the page
87. Instead of WebClient I'll use HttpClient.
88. Install-Package Microsoft.Net.Http to the project
89. Update the code to use async/await and
90. Now let's take a look at the behavior when I try and add 5 images
91. OK you probably noticed that the result seemed to be the same
92. OK that wasn't exactly what we were expecting. Let's take a close look at the code
93. We have modified the code so that the downloading happens asynchronously but not in parallel. So we still wait for the first image to be downloaded before we move on. We've still improved the situation because while you app is waiting it can serve other requests
94. The issues here is the usage of await let's refactor
95. Update GetImage to return a task
96. Update GetImages to loop through all tasks and call await Task.WhenAll
97. CTRL+F5 to demo again with 5 images

**Additional ideas**

1. Mention to not use Task.Run,Task.Factory, etc in asp.net
2. Show how to use cancellation tokens and how to combine more than 1 token
   1. See aspconf presentation
3. Show how to make an async page lifecycle event (i.e. wrap the method in using RegisterPageAsync & PageAsyncTask)
4. Demo EF6 with async
   1. Grab EF6 web forms sample here <http://www.asp.net/mvc/tutorials/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>
   2. F5 to see it running and then update some of the code to use async
   3. Rowan's talk: <http://channel9.msdn.com/Events/TechEd/NorthAmerica/2013/DEV-B337#fbid=>
5. Show how to use Web Essentials for Inspect Element from the browser to