Deploying multiple virtual directories to a single Azure Website

Since the beginning of time, IIS has allowed you to configure multiple *virtual directories* and *applications* within a single website. There are a bunch of situations where this is useful, but let's go with a common example. Suppose I have a website than has a bunch of pages, and a REST API which is accessed from scripts in the website. There are a few ways you could set this up, assuming the use of ASP.NET MVC and Web API:

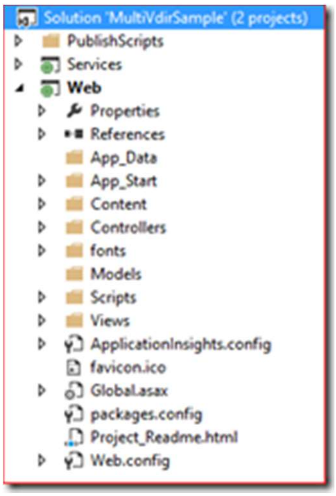
1. A single web project hosting both the web pages and API
2. One web project for the web pages, and another for the API, deployed to different websites on different servers
3. One web project for the web pages, and another for the API that is deployed to the same servers as an IIS application (/Services).

All of these options are valid, but the third option is tempting as it provides a degree of isolation between the two projects while minimising the deployment footprint. It also has the advantage of allowing you to share things like cookies and certain configuration between the two projects.

If you're hosting your solution in Microsoft Azure, you're probably already familiar with how to do this in a Web Role (using the <Sites> section in ServiceDefinition.csdef) or in a Virtual Machine (where you can set it all up yourself). But the best place for this solution is Azure Websites – and the good news is that it is also possible to replicate this configuration in a single Azure Website.

Set up the solution

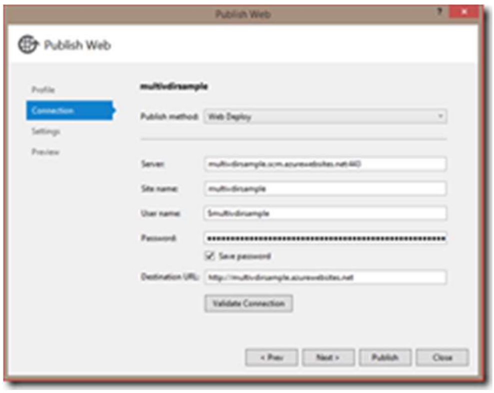
The first step is to set up your solution by creating separate web projects in Visual Studio. Here you can see I have the **Web** project for my website and the **Services** project for my API. I also changed the project properties for my **Services** project to run it in the /Services virtual directory on my local IIS Express web server for development purposes.



```
Solution 'MultiVdirSample' (2 projects)
▷ PublishScripts
▷ Services
◢ Web
   ▷ Properties
   ▷ References
     App_Data
   ▷ App_Start
   ▷ Content
   ▷ Controllers
   ▷ fonts
     Models
   ▷ Scripts
   ▷ Views
   ▷ ApplicationInsights.config
     favicon.ico
   ▷ Global.asax
     packages.config
     Project_Readme.html
   ▷ Web.config
```

Publish the root project

Once my solution is ready to deploy, the next step is to publish the root project (**Web**) to Azure websites. After choosing the appropriate subscription and website, I kept all the defaults in the Publish dialog.



multivdirsample

Publish method: Web Deploy

Server: multivdirsample.scm.azurewebsites.net:443

Site name: multivdirsample

User name: $multivdirsample

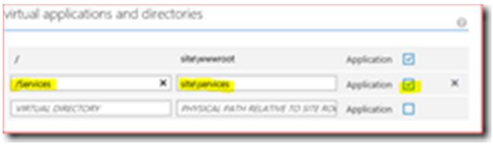
Password: ••

☑ Save password

Destination URL: http://multivdirsample.azurewebsites.net

Set up the Virtual Directory or Application

Now comes the important part. *Before* attempting to deploy the second project, go into the Azure Management Portal and configure the details for the IIS virtual directory or application in the website's configuration. You need so specify the name of the directory, the file system location, and whether you want to mark it as an IIS application. Be sure to save your settings to apply the changes to Azure.

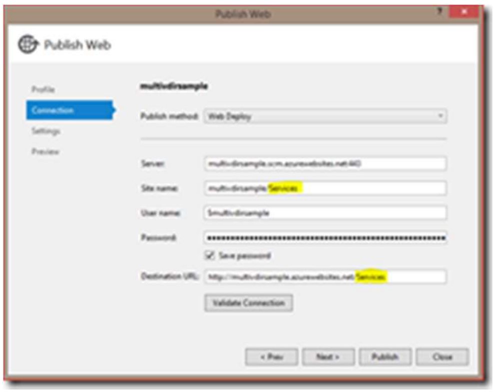


virtual applications and directories

/	site\wwwroot	Application	☑	
/Services	site\services	Application	☑	×
VIRTUAL DIRECTORY	PHYSICAL PATH RELATIVE TO SITE RO	Application	□	

Publish the child project(s)

Once your IIS virtual directory or application is set up, you can deploy your child projects. When using the Publish dialog, make sure you select the same Azure subscription and website that you used for your root project. Then you need to include the virtual directory path in the Site Name and Destination URL sections on the Connection tab.



multivdirsample

Publish method: Web Deploy

Server: multivdirsample.scm.azurewebsites.net:443

Site name: multivdirsample/Services

User name: $multivdirsample

Password: ••

☑ Save password

Destination URL: http://multivdirsample.azurewebsites.net/Services

Once the publishing is done, you’ll have your multi-project solution successfully deployed in separate virtual directories within a single Azure Website.