# Wisej Self Hosting

# Wisej Host Service

## 1 Overview

Wisej allows you to serve Wisej Web Applications on Windows machines **without IIS and without Cassini**.

Wisej.HostService.exe is a small single executable (~380K) that can run as a process or as a service and serve a single Wisej Application, plus unlimited sub-applications to local or external users.

## 2 How To Use it

Copy *Wisej.HostService.exe* to the root folder of your Wisej application, at the same level as *web.config* and run it. That's it, it will immediately serve the Wisej application to port 8080 by default.

**Note:** you may get an "Access Denied" error if you are not running on a Windows Server and don't have Administrator privileges because Windows blocks attempts at listening to all domains (*). In which case add "-d:localhost" to accept local connections, or run as administration.

### 2.1 Command Line Arguments

| Argument | Description |
|---|---|
| **(none) or -start** | Starts the host process for the current Wisej application: where the process is located. |
| **-stop** | Stops the process for the current Wisej application. |
| **-p:{port} or -port:{port}** | Changes the port. The default port is 8080. |
| **-d:{domain name} or -domain:{domain name}** | Limits the domain: i.e. * = all, or localhost to accept local connections only. The default is *. |
| **-i or -install** | Installs Wisej.HostService as a Windows service for the current Wisej application.<br><br>You can mix "-i" with "-p:" and "-n:" and "-d:". |
| **-u or -uninstall** | Uninstalls Wisej.HostService from the Windows services. |

| | |
|---|---|
| **-n:{name} or -name:{}** | Changes the name of the service, otherwise Wisej sets the name to “Wisej.HostService: “ + {Name of Wisej Application Folder} |

You can run several instances of Wisej.HostService to host multiple Wisej applications, as long as each instance listens to a different port.

# 3 CUSTOMIZATION

We provide the full source code for the *Wisej.HostService project*. You can modify and customize the host configuration however you like and create a custom service.

The place to customize the web server is in *WisejHost.Start()*:

```
WebApp.Start(options, (builder) =>
{
	/** TODO: Add additional middleware handlers here. */

	// ============================================
	// FileServer middleware processes all static
	// resources: html, png, etc...
	//
	// It should run before the Wisej middleware to
	// serve known static files.
	// --------------------------------------------
	builder.UseFileServer(new FileServerOptions
	{
		EnableDefaultFiles = true,
		EnableDirectoryBrowsing = true,
		FileSystem = new PhysicalFileSystem(HttpRuntime.AppDomainAppPath)
	});

	// ============================================

	// ============================================
	// Wisej Middleware processes all .wx, .aspx requests
	// through the System.Web pipeline.
	//
	// It will process all files not handled by the
	// FileServer middleware using the classic
	// pipeline and any handler/module added to
	// web.config.
	// --------------------------------------------
	builder.UseWisej();
	// ============================================

	/** TODO: Add additional middleware handlers here. */
});
```

You can add any additional OWIN middleware that you may want in your hosting service. There are middleware modules for authentication, logging, tracing, caching, redirecting. And, course, you can add your own middleware.

# 4 How Does it Work

We didn't recreate a new web server to replace IIS, nor we used Cassini.

Wisej Standalone and Wisej Self Hosting features use the Microsoft implementation of OWIN (Open Web Interface for .NET), called Katana: https://www.asp.net/aspnet/overview/owin-and-katana.

Wisej added a WisejMiddleware implementation to route requests to the Wisej Handler and you can add any OWIN middleware and recompile the project.

However, Wisej also added full support for ASP.NET/MVC applications, including all ASP.NET controls, WebForms, etc. Basically, our implementation can fully replace IIS in a **single executable**.

## 4.1 Single Executable

We like simplicity. The *Wisej.HostService* projects are configured to merge all the OWIN and Katana assemblies into a single executable using *ILMerge* and some code we added to let the host know where to load its classes.

# 5 Conclusion

Wisej Self Hosting is an easy and inexpensive way to host Wisej applications on machines of any size.

Last, you can use it to host pure ASP.NET/MVC (including WebForms) applications.