Несколько сайтов на одном биндинге.

Виртуальные директории,

Applications,

Деплоинг,

Дефолтные странитцы,

Конфигурация,

Как создать распределенные сайты

HttpHandler

**Notes**

C:\Windows\System32\inetsrv\Config\applicationhost.config

C:\Windows\System32\drivers\etc\hosts

* **system.applicationHost** - Contains configuration settings for sites, applications, virtual directories, and application pools. These are centralized settings that cannot be distributed.

Modules(IhttpModule methods Init, Dispose) – subscribe handlers on events lifecycle for (for example add some handlers to response) –> web.config(system.webserver -> modules) or iis manager -> <add/remove name=”” type=”{namespace.class}” precondition=” integratedMode,managedHandler”/> - precondition – defines use cases for module like (managedHandler use only by managed handlers)

Benefits of integrated modules

 The new request-processing architecture consists of an ordered list of native and managed modules that perform specific tasks in response to requests.

First, all file types can use features that were originally available only to managed code. For example, you can now use ASP.NET Forms authentication and Uniform Resource Locator (URL) authorization for static files, Active Server Pages (ASP) files, and all other file types in your sites and applications.

Second, this design eliminates the duplication of several features in IIS and ASP.NET. For example, when a client requests a managed file, the server calls the appropriate authentication module in the integrated pipeline to authenticate the client. In previous versions of IIS, this same request would go through an authentication process in both the IIS pipeline and in the ASP.NET pipeline.

Third, you can manage all of the modules in one location, instead of managing some features in IIS and some in the ASP.NET configuration. This simplifies the administration of sites and applications on the server.

HttpHandlers(IHTTPHandler methods ProcessingRequest(contetxt) prop IsReusable -> )

=

<site name="MySite" id="3">

<application path="/" applicationPool="MySite">

<virtualDirectory path="/" physicalPath="d:\mysite\www" />

</application>

</site>

<site name="Contoso" id="2" serverAutoStart="true">

<application path="/">

<virtualDirectory path="/" physicalPath="C:\Contoso\Content" />

</application>

<application path="/CRM">

<virtualDirectory path="/" physicalPath="C:\Contoso\Content\CRM" />

<virtualDirectory path="/Images" physicalPath="E:\Images" />

</application>

<bindings>

<binding protocol="http" bindingInformation="\*:80:www.contoso.com" />

</bindings>

</site>

Каждый <site> содержит коллекцию <application>. Всегда будет по крайней мере одно приложение, которое определяет корневое приложение, /.

Атрибут applicationPool указывает, какой пул приложений использовать.

Обратите внимание, что существует единственный дочерний элемент: virtualDirectory.

Каждый application имеет дочерний набор элементов virtualDirectory, и в этой коллекции обычно будет хотя бы один элемент.

По умолчанию <virtualDirectory> в корневом приложении сообщает нам:

* это это корень (path="/") и
* что он физически находится в файловой системе в d:\mysite\www (physicalPath="d:\MySite\www").

path каждого virtualDirectory относится к path, указанному в родительском пути application.