Из цикла лекций **«Технологии разработки Internet-приложений»** для студентов 4-го курса кафедры Компьютерных технологий физического факультета Донецкого национального университета

Технологии разработки **Internet-приложений**

ASP.NET приложения –

ASP.NET-процесс, пул, домен приложения, компиляция, конвейер HTTP

проф. В.К.Толстых, www.tolstykh.com

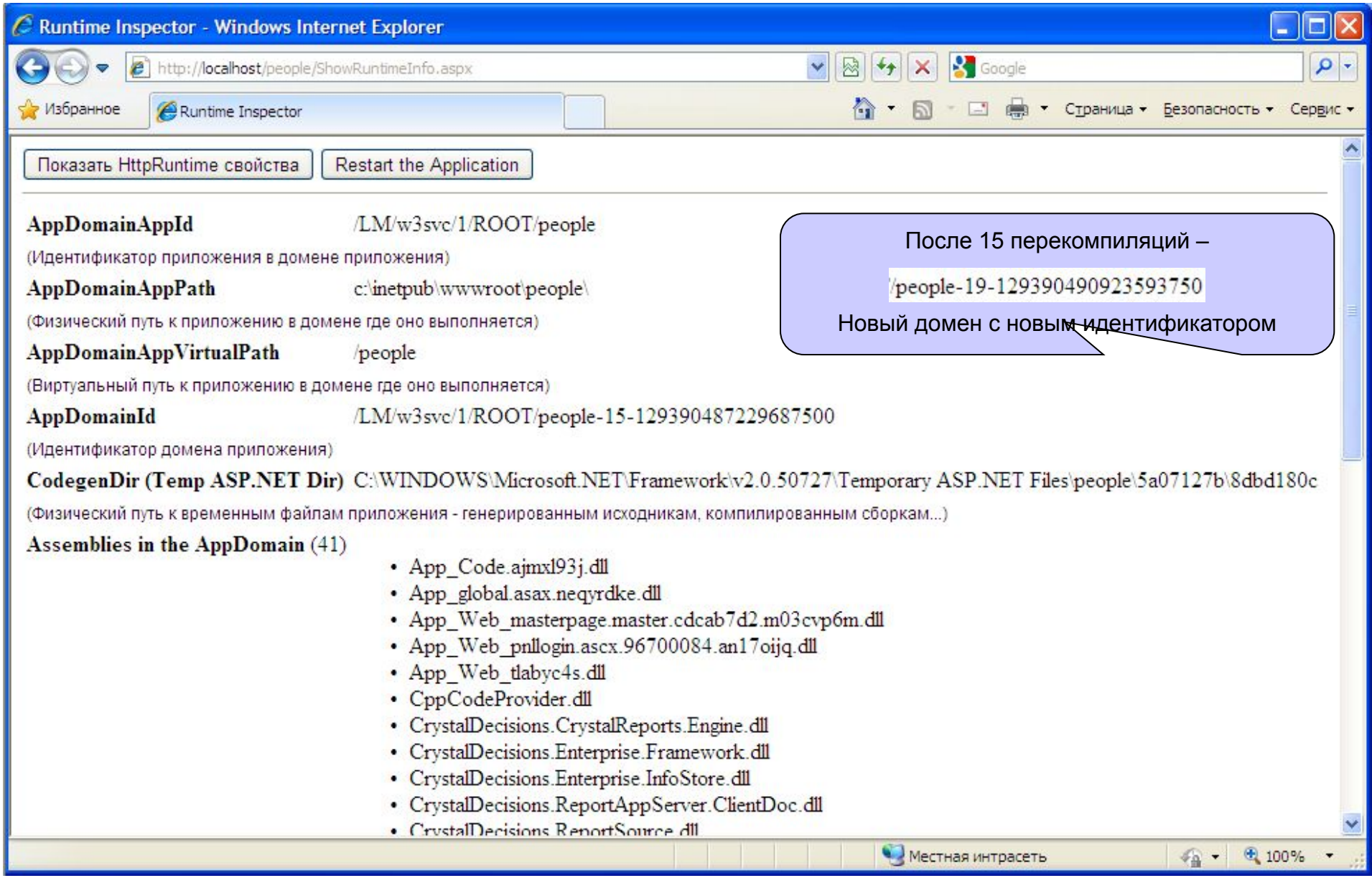
Терминология

Рабочий процесс (РП) – **w3wp.exe** обслуживает любые приложения, которые могут выполняться на данном Web-сервере. **РП** ничего не знает об ASP.NET. **РП** запускает конвейер HTTP для приложения посредством создания экземпляра **HttpRuntime** (выполнение HTTP-запроса) и загружает CLR.

AppPool (пул приложений) – это группа приложений, для выполнения которых используется одна и та же копия **РП** и запрещается доступ от других **РП**. Все запросы к приложениям в пуле ставятся в одну очередь этого пула.

AppDomain – домен приложения, виртуальная папка в которой готовятся ответы клиенту на запросы каких-либо ресурсов данного приложения. Непосредственно виртуальная папка домена приложения содержит классы приложения в виде откомпилированных **.dll-**сборок, готовых обрабатывать запросы клиентов. Невозможно выгрузить сборку не выгрузив весь домен. Каждая *перекомпиляция* проекта (или пересохранение файлов **Global.asax**, **web.config**) не удаляет старые сборки проекта в его домене, а создаёт новые копии. Только после 15 перекомпиляций удаляется весь домен со сборками и создаётся новый домен с новым идентификатором. Использование доменной модели позволяет разграничить память, выделяемую каждому приложению. В рамках одного домена приложения совместно используются, находящиеся в памяти глобальные данные приложения (**Application**, **Session**, **Cache**...). Управление доменами осуществляется через объект `ApplicationManager`.

Информация о домене приложения



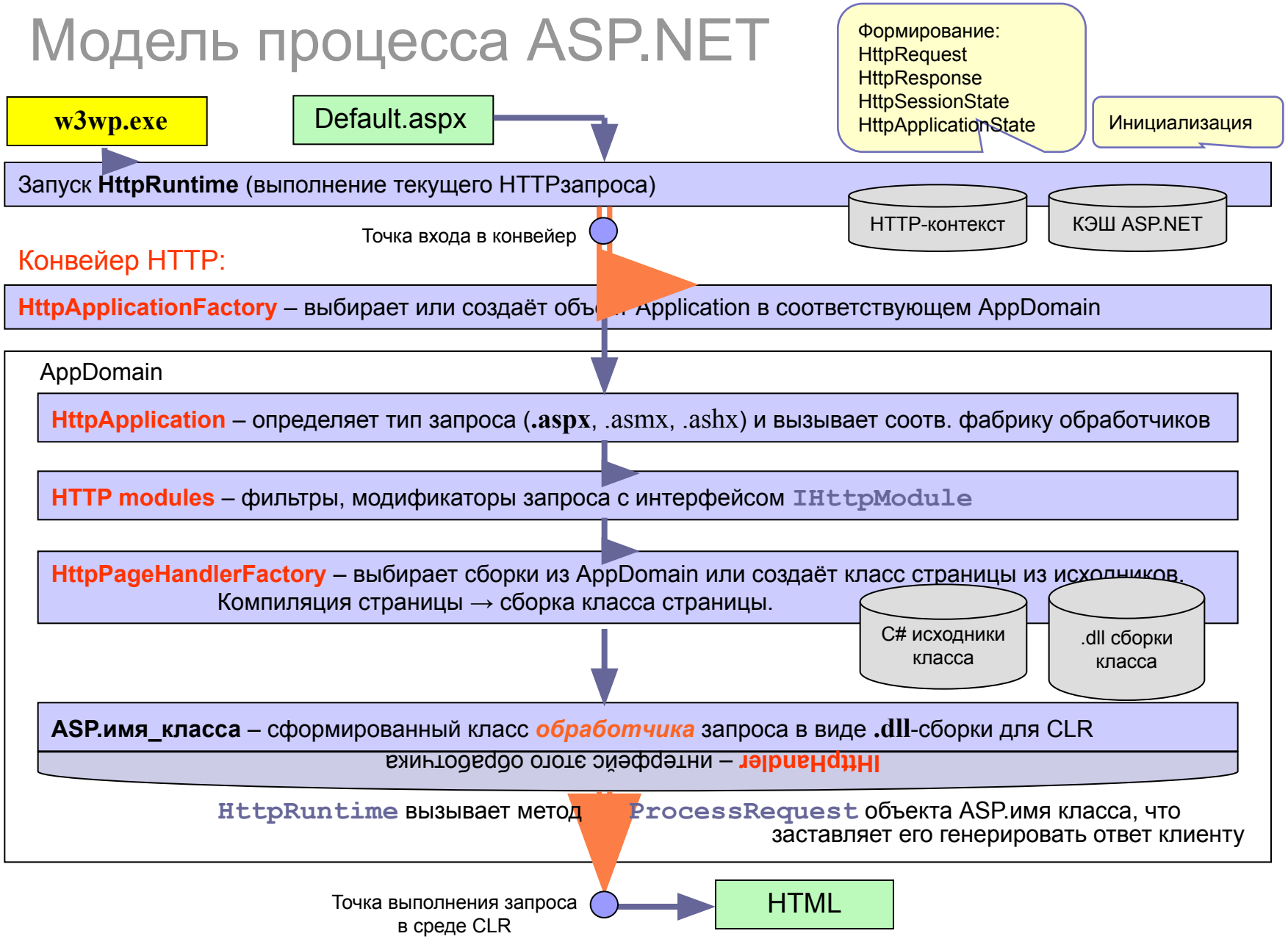
…терминология

Application – приложение, экземпляр класса **`HttpApplication`**, который создаётся вместе со своим доменом AppDomain при первом обращении к каким-либо ресурсам приложения и формируется на базе классов в сборках своего AppDomain. Если готовых сборок нет, то HTTP-конвейер, создаёт требуемые сборки. События, формирующие Application, задаются в файле **Global.asax**. Именно поэтому изменения файла **Global.asax** приводят к построению проекта заново.

HTTP-конвейер превращает HTTP-запрос клиента в dll-экземпляры класса CLR запрошенного ресурса, который затем в JIT-компиляторе CLR генерирует *нативный* код для конкретного процессора, после работы в котором получается готовый ответ клиенту в виде HTML-разметки для Web-браузера или XML-разметки для Web-сервиса.

`HttpRuntime` обеспечивает поддержку исполнения текущего ASP.NET приложения: формирование HTTP-контекста из запроса, инициализация КЭШ и монитора изменений файлов приложения, вызов метода **`ProcessRequest`** для запуска в CLR полученного обработчика запроса.

Модель процесса ASP.NET



Задача конвейера HTTP

– найти в нужном домене приложения управляемый класс (.NET класс разработчика), который необходим для реализации текущего запроса. Например, если был запрошен файл **Default.aspx** с атрибутом `ClassName = _Default`, то в **AppDomain** данного приложения будет отыскиваться класс `ASP._Default`. Если такой класс имеется, то будет создан его экземпляр в CLR – ***обработчик запроса*** с интерфейсом `IHttpHandler` и вызван его метод `ProcessRequest`, который сгенерирует ответ клиенту.

Если необходимый управляемый класс в **AppDomain** приложения отсутствует, или он устарел, то класс динамически создаётся фабрикой обработчиков (`...HandlerFactory`).

Компонент конвейера HTTP : `HttpApplicationFactory`

– это фабрика создания приложений и поддержки пула приложений. `HttpApplicationFactory` выясняет имеется ли уже для виртуальной папки сайта, к которой обращён запрос, необходимый **AppDomain**. Если – да, т. е. приложение уже выполняется, то `HttpApplicationFactory` выбирает из пула соответствующее приложение и передаёт ему запрос. В противном случае создаётся новый объект `HttpApplicationFactory` со своим **AppDomain** и ему передаётся запрос.

Компонент конвейера HTTP : **HttpApplication**

– это приложение ASP.NET, – динамически создаваемый экземпляр класса **HttpApplication**. **HttpApplication** определяет тип запроса (страница ASP.NET – файл **.aspx**, Web-сервис – файл **.asmx**, пользовательский обработчик HTTP-данных, не имеющий пользовательского интерфейса – файл **.ashx**) и привлекает соответствующую фабрику для создания обработчика запроса.

Компонент конвейера HTTP : **HTTP modules**

– это классы, реализующие интерфейс **IHttpModule** и обрабатывающие события времени выполнения (Runtime). HTTP-модули позволяют выполнять различные действия на любом этапе жизненного цикла страницы или приложения. В HTTP-модуле можно подписаться на любое событие жизненного цикла и обрабатывать его, реализуя при этом какую-то свою логику. Обычно, модули используют как фильтры приложения. Каждый модуль запускается методом **Init**. В этом методе каждый HTTP-модуль подписывается на события жизненного цикла приложения (**BeginRequest** , **AuthenticateRequest** … – всего 26). После этих действий все загруженные модули HTTP остаются в памяти и выгружаются только тогда, когда выгружается домен приложения.

Компонент конвейера HTTP : ...HandlerFactory

– это фабрики создания обработчиков HTTP-запросов. Обязанности фабрики – либо найти **.dll**-сборку в **AppDomain**, где содержится класс запрошенного ресурса, либо – динамически создать эту сборку по требованию. Исходный код C# класса генерируется путём сканирования частичных классов запрашиваемого ресурса (**.aspx**, **.asmx**, **.ashx**) и временно сохраняется в папке **%SystemRoot%\Microsoft.NET\Framework** версия **\Temporary ASP.NET Files\папка_приложения** (если директива `@Page` не имеет атрибут `debug=true`, то эти файлы удаляются сразу после обработки запроса). Далее сформированный класс компилируется. Полученная сборка фиксируется в **AppDomain** (кэшируется во временной папке рядом с папкой приложения) и в виде обработчика запроса загружается в память где работает CLR.

Например:
`PageHandlerFactory` – фабрика создания обработчиков для страниц ASP.NET (**.aspx**),
`WebServiceHandlerFactory` – фабрика создания обработчиков для Web-сервисов (**.asmx**),
и т. д.

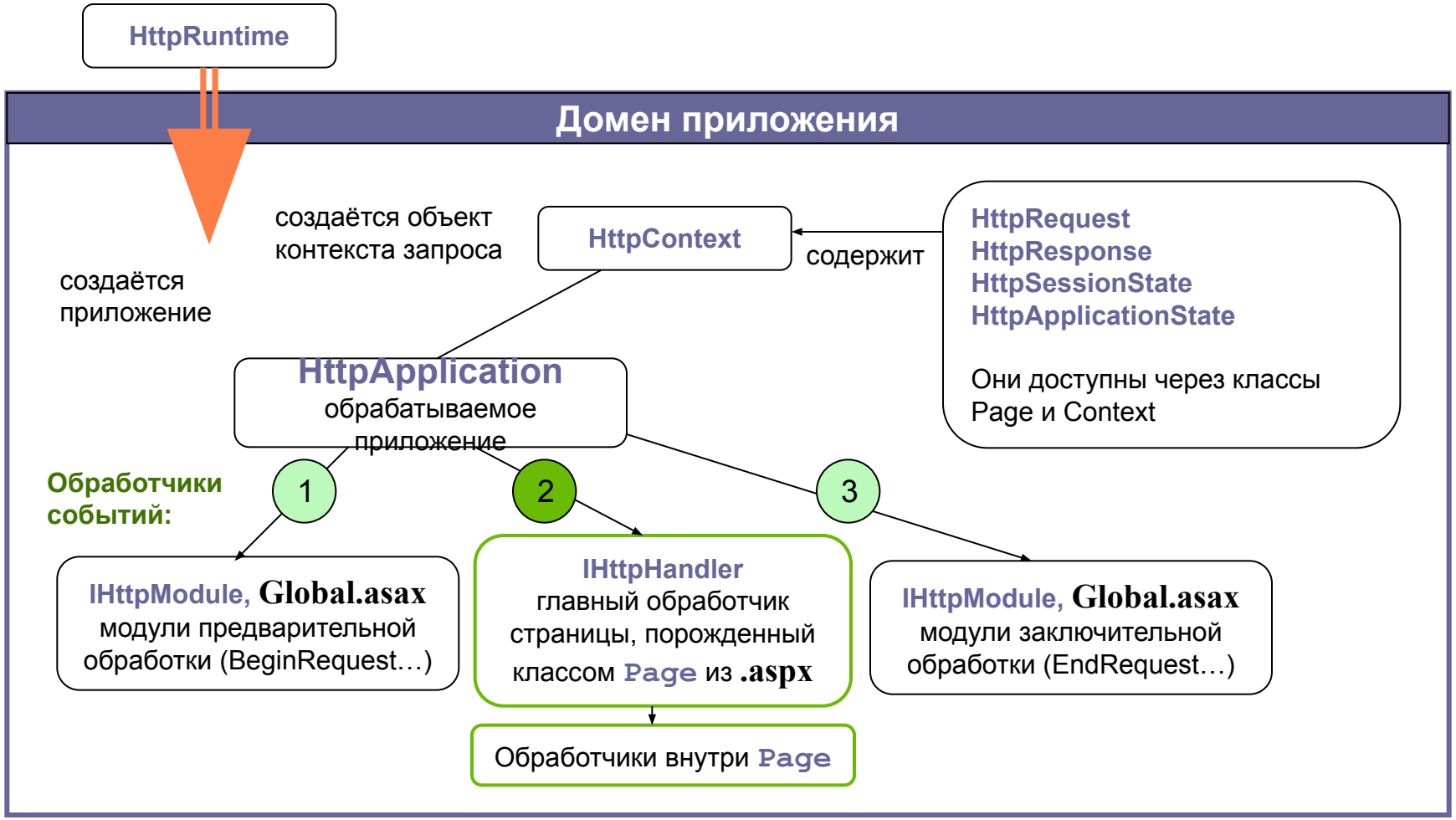
Необходимая фабрика определяется объектом `HttpApplication`, а связь фабрики с типом ресурсов задаётся в файлах **machine.config** и корневом **web.config** в узле `<httpHandlers>` .

Компонент конвейера HTTP : IHttpHandler

– это обработчик HTTP-запроса с интерфейсом `IHttpHandler`, созданный какой-либо предыдущей фабрикой. Например, при запросе страницы **Default.aspx** с атрибутом `ClassName=_Default` **.dll**-сборка обработчика в CLR будет представлять объект **ASP._Default**, являющийся экземпляром класса `System.Web.UI.Page`.

Приложение в конвейере HTTP с событиями

Главная ***точка входа*** в HTTP-конвейер – это модуль **HttpRuntime** каждого ASP.NET приложения.



Обработчики событий приложения могут быть в файле **Global.asax** или HTTP-модулях с интерфейсом **IHttpModule**.