

Конвейер обработки запросов ASP.NET MVC 3

IIS/ASP.NET

Входящий запрос HTTP

Много чего происходит тут, пока запрос дойдёт до модуля маршрутизации

application.PostResolveRequestCache += new EventHandler(this.OnApplicationPostResolveRequestCache);

Инициализирует модуль и подготавливает его для обработки запросов

Сопоставляет HTTP-запрос маршруту, извлекает обработчик для текущего маршрута и задаёт его как обработчик HTTP для текущего запроса

```

//...
//Обработчик маршрута
IRouteHandler routeHandler = routeData.RouteHandler;
//...
//Обработчик HTTP-данных
IHttpHandler httpHandler = routeHandler.GetHttpHandler(requestContext);
//...
//Задаёт обработчик для запроса
context.RemapHandler(httpHandler);
    
```

Обработчик маршрута

Фактически отсюда начинается обработка запроса MVC

Стандартный обработчик запроса MVC

ProcessRequest(HttpContext httpContext)

ProcessRequestInit(HttpContextBase httpContext, out IController controller, out IControllerFactory factory)

factory = ControllerBuilder.GetControllerFactory();

Создаётся объект фабрики контроллеров

Собственная реализация фабрики контроллеров

Фабрика контроллеров с контейнером DI

DefaultController Factory : IControllerFactory

controller = factory.CreateController(RequestContext, controllerName);

Создаётся объект контроллера

Controller : ControllerBase, IController

controller.Execute(RequestContext);

Выполняет контекст запроса

Execute(RequestContext requestContext)

Execute(RequestContext requestContext)

ExecuteCore()

Свойство ActionInvoker контроллера можно изменить, для пропуска некоторых фильтров, путём наследования от стандартного класса контроллера

```

//...
string actionName = RouteData.GetRequiredString("action");
if (!ActionInvoker.InvokeAction(ControllerContext, actionName))
{
    HandleUnknownAction(actionName);
}
//...
    
```

Свойство ActionInvoker вызывает метод действия контроллера, полученный из таблицы маршрутизации

InvokeAction(ControllerContext controllerContext, string actionName)

IAuthorizationFilter

Не прошёл авторизацию. код ответа 401

InvokeAuthorizationFilters(controllerContext, filterInfo.AuthorizationFilters, actionDescriptor);

InvokeActionMethodWithFilters(controllerContext, filterInfo.ActionFilters, actionDescriptor, parameters);

InvokeActionMethodWithFilters(controllerContext, filterInfo.ActionFilters, actionDescriptor, parameters);

IActionFilter

IExceptionFilter

ActionResult

Фильтров может быть несколько

Фильтр обработки ошибок пропустил

Инкапсулирует результат метода действия

OnActionExecuting(ActionExecutingContext filterContext)

OnActionExecuted(ActionExecutedContext filterContext)

OnResultExecuting(ResultExecutingContext filterContext)

OnResultExecuted(ResultExecutedContext filterContext)

InvokeActionResultWithFilters(controllerContext, filterInfo.ResultFilters, postActionContext.Result);

ViewResult

IViewEngine

Стандартные представления: WebForm или Razor

Результат отличный от ActionResult, например обычная строка

Собственный движок представления

Не ViewResult, например JSON

Выполняются перед и после метода результата действия

Ответ клиенту

Маршрутизация

Обработчик запроса

Фабрика контроллеров

Контроллеры и действия