Ak Bars Bank

AK BARS DIGITAL

ASP.NET Core 2.1

разрабатываем готовые к продакшн приложения

Поломошнов Дмитрий, senior backend developer
Ак Барс Цифровые Технологии

ASP.Net Core - это не только модно

Помимо основного функционала

- Версионирование
- Документирование API
- Надежность
- Логирование
- Мониторинг
- Управление отдельными фичами
- Безопасность

Версионирование

- Совместимые изменения

  Аккуратнее с изменениями, касающимися поведения

  → оставляем текущую версию

- Несовместимые изменения

  Новый контракт обратно несовместим со старым

  → увеличиваем версию (major/minor)

Версионирование

- Microsoft.AspNetCore.Mvc.Versioning
  - Позволяет легко управлять версиями HTTP методов
  - Поддержка нескольких версий API одновременно
  - Рекламирование новых версий и факта устаревания старых
- Разводим контроллеры разных версий по пространствам имен или сборкам

Версионирование

/api/v2/kittens

/api/kittens?api-version=2.0

ACCEPT: application/vnd.kittenserver.app-v2.0+json

Custom Header

Документирование

- Спецификация OpenAPI (Swagger) и навигатор по API
  - NSwag
- Генерация клиентских библиотек
  - NSwag
  - Autorest
  - Refit
  - Вручную

Документирование

- действия

  ActionResult<T>

  [ProducesResponseType(typeof(T), 200)]

- модели

  [Required]

  [DefaultValue(5)]

- XML документация

  <summary… />, <param.../>, <response.../>

Demo

Версионирование, Документирование

Надежность

Примеры внешних ресурсов

- Базы данных
- Сторонние вебсервисы
- Сервисы кеширования
- Системы обработки асинхронных сообщений
- Файловые хранилища

Паттерны обработки внешних ошибок

- Retry
- Circuit breaker
- Timeout
- Bulkhead isolation
- Fallback

Используем существующие механизмы повтора, если они есть

```
services.AddDbContext<KittenDbContext>(x =>
{
    string connectionString = Configuration.GetConnectionString("KittenDb");
    x.UseSqlServer(connectionString, o =>
    {
        o.EnableRetryOnFailure(
            maxRetryCount: 3,
            maxRetryDelay: TimeSpan.FromSeconds(20),
            errorNumbersToAdd: null);
    });
});
```

Polly

- Декларативная конфигурация
- Изолируем описание политик обработки ошибок от основной логики
- Независимо для каждого внешнего ресурса
- Логирование
- Переиспользуем настроенные политики

HttpClientFactory

- Инкапсуляция управления жизненным циклом HttpClientHandler
- Конфигурация pipeline для исходящих запросов
  - обработка исключительных ситуаций (Polly)
  - реализация общих подходов к логированию, авторизации
- Типизированные клиенты поверх HttpClient
- Гибкие возможности для интеграционного тестирования

Demo

HttpClientFactory

Логирование

- Структурированное логирование (json)
- Логирование исключений при запуске приложения
- Сохранение полезного контекста (BeginScope)
- Связывание внешних запросов и запросов в сторонние сервисы
- Асинхронная запись на диск

Health checks

- Проверка доступности приложения
- Проверка доступности зависимых ресурсов
- Набор встроенных проверок
- Легко создать собственные

Feature toggling

- Настройка доступности фич через параметры и переменные окружения
- Встроенные сценарии управления доступностью фич
- Простота запроса доступности фичи в коде

Безопасность

- Redirect to HTTPS
- HSTS

  Strict-Transport-Security: max-age=31536000; includeSubDomains
- Не выставляем наружу технические подробности ошибок

Sum up

- Версионирование
- Swagger
- HttpClientFactory
- Polly
- Health checks
- Logging
- Feature toggling
- HSTS

Спасибо за внимание!

dmitriy.polomoshnov@akbarsdigital.ru