

Introduction to .NET

Florin Olariu

“Alexandru Ioan Cuza”, University of Iași

Department of Computer Science

Agenda

- ▶ Best practices in Web API's
- ▶ Fluent Validations
- ▶ AutoMapper
- ▶ Starting Microservices or “SOA done right!”
- ▶ Web API - Demo
- ▶ Questions

Best practices in Web API's

- ▶ Semantic best practices
- ▶ Technical best practices

Best practices in Web API's - Semantic best practices

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
 - ▶ GET - <http://localhost:16991/v1/cities>
 - ▶ GET - <http://localhost:16991/v1/cities/1/poi>
 - ▶ POST - <http://localhost:16991/v1/cities/1/poi>
 - ▶ PUT - <http://localhost:16991/v1/cities/1/poi/3>

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.

DO	DO NOT
Use GETs to get data.	Use status codes in a way that is not expected. If returning a set of objects, don't return 404 if the set is empty—return the empty set. That's what people expect.
Use PUTs/POSTs to change/add data.	Use GETs to alter data or PUTs/POSTs to only get data.
Pick a few good status codes to use consistently and use them consistently and correctly throughout your API.	Use every status code that tangentially relates to what you're trying to tell your consumer. Use error messages to describe invalid conditions.
Use the same general endpoint structure throughout.	

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's

`[Route("v1/cities")]`

`[ApiController]`

`public class PoiController : ControllerBase`

`{`

`}`

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's.
- ▶ KISS.

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's.
- ▶ KISS.
 - ▶ Don't expose more than you think needs exposing.

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's.
- ▶ KISS.
 - ▶ Don't expose more than you think needs exposing.
 - ▶ Don't use 43 different status codes.

Best practices in Web API's - Semantic best practices

- ▶ Think nouns, not verbs.
- ▶ Be consistent.
- ▶ Versioning Web API's.
- ▶ KISS.
 - ▶ Don't expose more than you think needs exposing.
 - ▶ Don't use 43 different status codes.
 - ▶ Keep your DTOs simple.

Best practices in Web API's - Technical best practices

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.

- ▶

```
public class Poi
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
}
```

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.

- ▶

```
public class Poi
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
}
```

- ▶

```
public class PoiForCreatingDto
{
    public string Name { get; set; }
    public string Description { get; set; }
}
```

- ▶

```
public IActionResult CreatePoi(int cityId, [FromBody] PoiForCreatingDto poi)
```

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.
- ▶ Validate everything.

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.
- ▶ Validate everything.

- ▶

```
public class PoiForCreatingDto
{
    [Required]
    [MaxLength(30)]
    public string Name { get; set; }
    [Required]
    [MaxLength(200)]
    public string Description { get; set; }
}
```

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.
- ▶ Validate everything.

```
[HttpPost("{cityId}/poi")]
public IActionResult CreatePoi(int cityId, [FromBody] PoiForCreatingDto poi)
{
    if (poi == null)
    {
        return BadRequest();
    }
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    } ... }
```

Best practices in Web API's - Technical best practices

- ▶ Use DTOs to move data back and forth.
- ▶ Validate everything.
- ▶ Keep your controllers as thin as possible. Enforce separation of concerns. Separation of concerns means things are testable.

Fluent Validations



Fluent Validations

- ▶ “A small validation library for .NET that uses a fluent interface and lambda expressions for building validation rules.”
 - ▶ <https://github.com/JeremySkinner/FluentValidation>

Fluent Validations

- ▶ “A small validation library for .NET that uses a fluent interface and lambda expressions for building validation rules.”
 - ▶ <https://github.com/JeremySkinner/FluentValidation>
- ▶ Using: Install-Package FluentValidation.AspNetCore or Nuget package in Visual Studio

Fluent Validations

► “A s
exp

► Usir
Stuc

```
using FluentValidation;

public class CustomerValidator: AbstractValidator<Customer> {
    public CustomerValidator() {
        RuleFor(customer => customer.Surname).NotEmpty();
        RuleFor(customer => customer.Forename).NotEmpty().WithMessage("Please specify a first name");
        RuleFor(customer => customer.Discount).NotEqual(0).When(customer => customer.HasDiscount);
        RuleFor(customer => customer.Address).Length(20, 250);
        RuleFor(customer => customer.Postcode).Must(BeAValidPostcode).WithMessage("Please specify a valid postcode");
    }

    private bool BeAValidPostcode(string postcode) {
        // custom postcode validating logic goes here
    }
}

Customer customer = new Customer();
CustomerValidator validator = new CustomerValidator();
ValidationResult results = validator.Validate(customer);

bool validationSucceeded = results.IsValid;
IList<ValidationFailure> failures = results.Errors;
```

AutoMapper

The background of the slide features abstract, overlapping geometric shapes in various shades of blue, ranging from light sky blue to deep navy blue. These shapes are primarily located on the right side and bottom of the frame, creating a modern, dynamic aesthetic.

AutoMapper

- ▶ “AutoMapper uses a fluent configuration API to define an object-object mapping strategy. AutoMapper uses a convention-based matching algorithm to match up source to destination values. AutoMapper is geared towards model projection scenarios to flatten complex object models to DTOs and other simple objects, whose design is better suited for serialization, communication, messaging, or simply an anti-corruption layer between the domain and application layer.” ->

<http://docs.automapper.org/en/stable/index.html>

AutoMapper

- ▶ <https://dotnetcademy.net/Learn/2/Pages/2>

Starting Microservices or “SOA done right!”

Starting Microservices or “SOA done right!”



The image shows a screenshot of an Amazon product listing. On the left, there is a vertical strip of five small thumbnail images showing different views and colors of the banana-shaped storage boxes. The main image is a large, high-quality photo of three banana-shaped storage boxes in pink, green, and yellow, arranged in a slightly overlapping manner. To the right of the image, the product title is 'Benran Outdoor Travel Cute Banana Protector Storage Box (Pink, Green, Yellow, Pack of 3)'. Below the title is a star rating of 4.5 stars and the text '26 customer reviews'. The price is listed as '\$10.00 & FREE Shipping'. Below the price, there is a promotional message: 'Get \$50.00 off instantly: Pay \$0.00 upon approval for the Amazon Rewards Visa Card. Learn more'. At the bottom, it says 'Only 9 left in stock.' and 'This item does not ship to Garlasco, Italy. Please check other sellers who may ship internationally. Ships from and sold by Ziyue.'

BENRAN

Benran Outdoor Travel Cute Banana Protector Storage Box (Pink, Green, Yellow, Pack of 3)

★★★★★ 26 customer reviews

Price: \$10.00 & FREE Shipping

i Get \$50.00 off instantly: Pay \$0.00 upon approval for the Amazon Rewards Visa Card. [Learn more](#)

Only 9 left in stock.

This item does not ship to Garlasco, Italy. Please check other sellers who may ship internationally.

Ships from and sold by Ziyue.

- ▶ Copyright : Mauro Servienti => “SOA done right!”

Starting Microservices or “SOA done right!”

- ▶ Who is in charge of changing, for example, the price?
- ▶ Who is responsible for business rules affecting shipping costs?
- ▶ Where do we store stars and customers' reviews?

Starting Microservices or “SOA done right!”



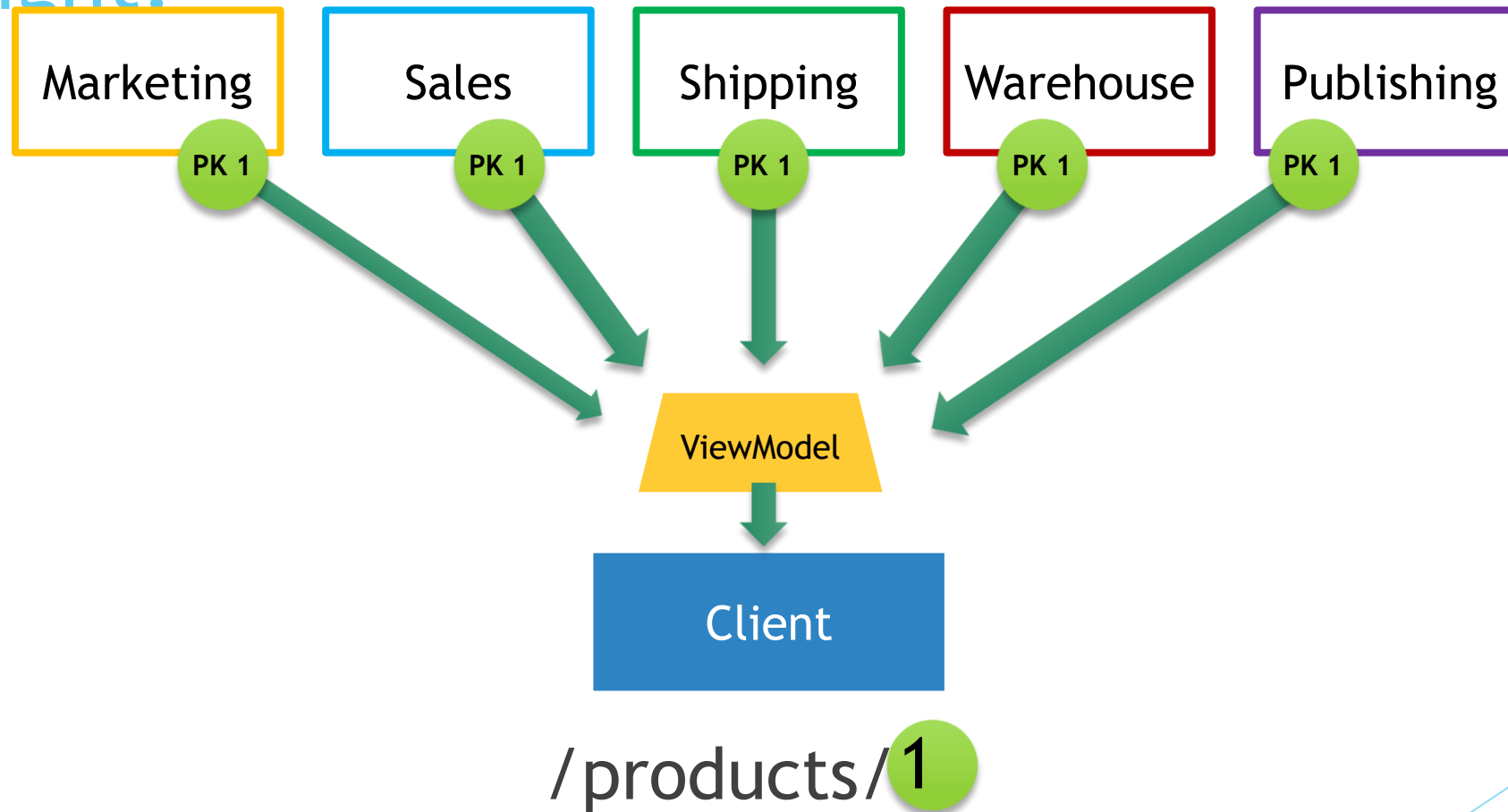
- ▶ Copyright Mauro Servienti => “SOA done right!”

Starting Microservices or “SOA done right!”

- ▶ (micro)services owning their own piece of information.

Single Responsibility Principle

Starting Microservices or “SOA done right!”



Starting Microservices or “SOA done right!”

▶ Much more next time ... 😊

Web API

► Flights - Demo

One more thing...

One more thing...

Rules of Optimization:

Rule 1: Don't do it.

Rule 2 (for experts only): Don't do it yet.

Bibliography

- ▶ Pluralsight
- ▶ Mauro Servienti - “SOA done right!”

Questions?



Thanks!

See you next time! 😊