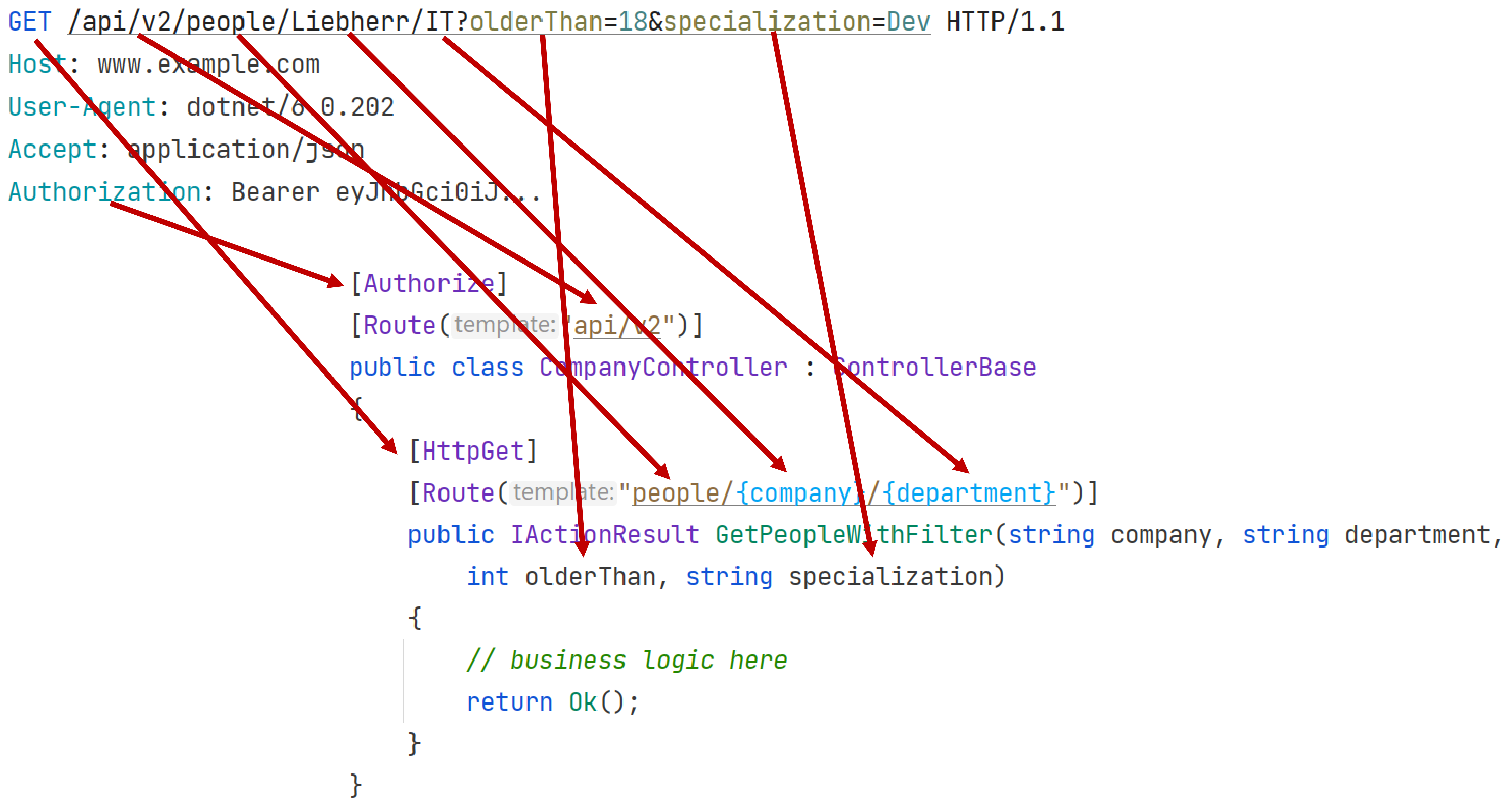


```
GET /api/v2/people/Liebherr/IT?olderThan=18&specialization=Dev HTTP/1.1
Host: www.example.com
User-Agent: dotnet/6.0.202
Accept: application/json
Authorization: Bearer eyJhbGciOiJ...
```

```
[Authorize]
[Route(template: "api/v2")]
public class CompanyController : ControllerBase
{
    [HttpGet]
    [Route(template: "people/{company}/{department}")]
    public IActionResult GetPeopleWithFilter(string company, string department,
        int olderThan, string specialization)
    {
        // business logic here
        return Ok();
    }
}
```

Binding Domain Specific Value Types

To Endpoints in ASP.NET (Core) with three different approaches

Hi, I‘m David

- C# Developer @ Liebherr
- Slides and code at GitHub

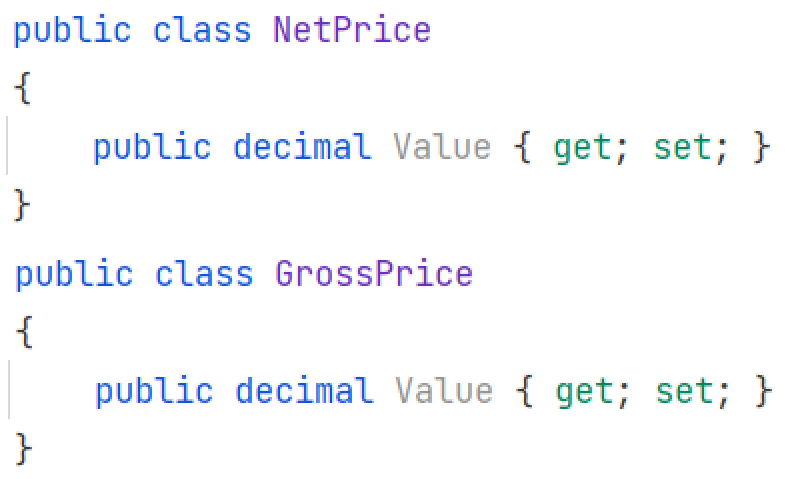
https://github.com/davidkroell/BindingDomainSpecificValueTypes

Scope

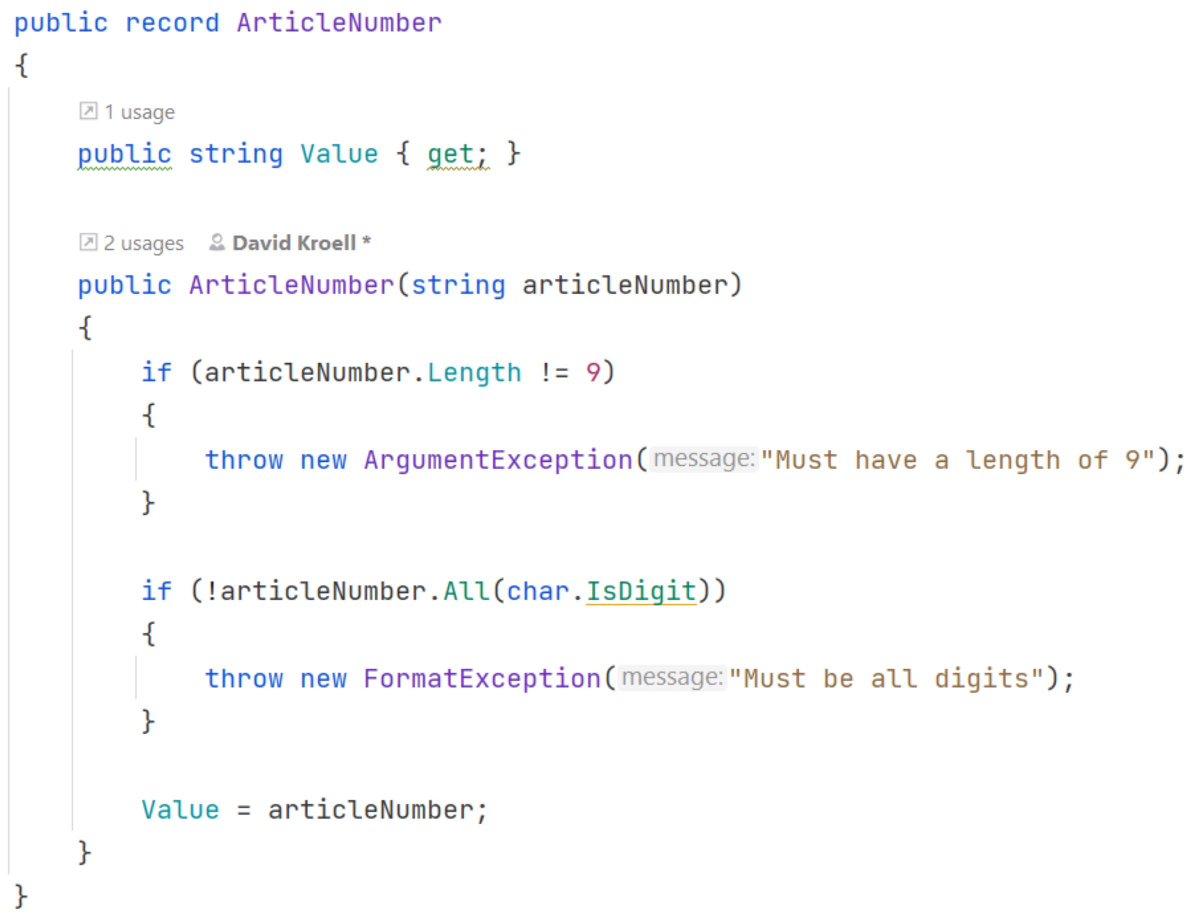
- Domain Specific Value Types
- Brief introduction to ASP.NET
- Binding a model with
  - TypeConverter
  - ModelBinder
  - ModelBinderProvider
- Architecture & Summary

Domain Specific Value Types

- a domain specific type
- with a single value



```
public class NetPrice
{
    public decimal Value { get; set; }
}
public class GrossPrice
{
    public decimal Value { get; set; }
}
```



```
public record ArticleNumber
{
    1 usage
    public string Value { get; }

    2 usages   David Kroell *
    public ArticleNumber(string articleNumber)
    {
        if (articleNumber.Length != 9)
        {
            throw new ArgumentException(message: "Must have a length of 9");
        }

        if (!articleNumber.All(char.IsDigit))
        {
            throw new FormatException(message: "Must be all digits");
        }

        Value = articleNumber;
    }
}
```

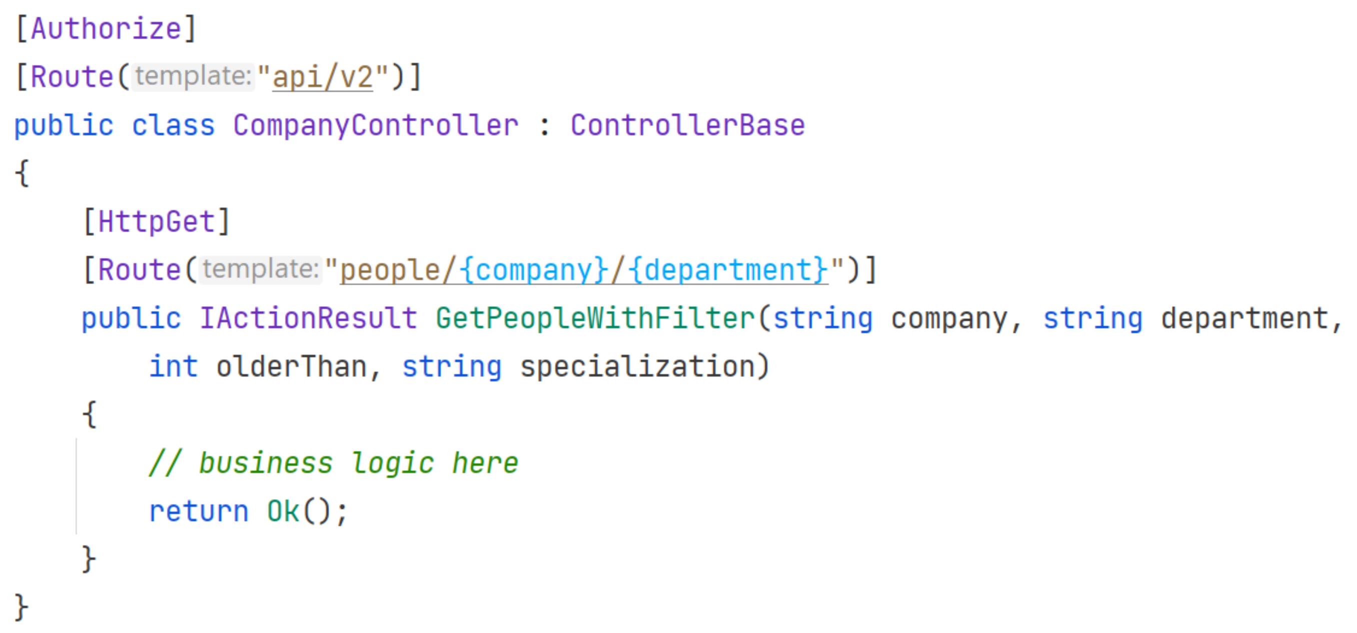
ASP.NET

- Web-Framework for C#, from Microsoft
  - Webpages, APIs

- Rewrite of the .NET Framework ASP.NET
- Open Source

ASP.NET – The Problem

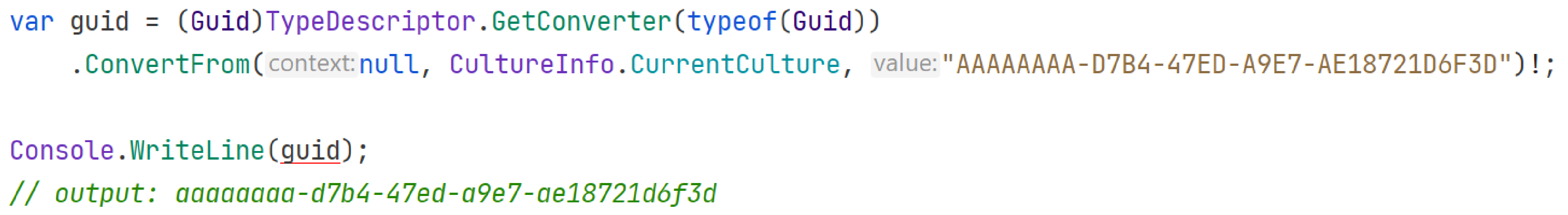
- Only primitive Types in GET's
- A simple solution:
  - Bind primitives, create object in route
- May get messy



```
[Authorize]
[Route(template:"api/v2")]
public class CompanyController : ControllerBase
{
    [HttpGet]
    [Route(template:"people/{company}/{department}")]
    public IActionResult GetPeopleWithFilter(string company, string department,
        int olderThan, string specialization)
    {
        // business logic here
        return Ok();
    }
}
```

TypeDescriptor & TypeConverter

- Define a Converter for a Type
- Retrieve the Converter
- Convert



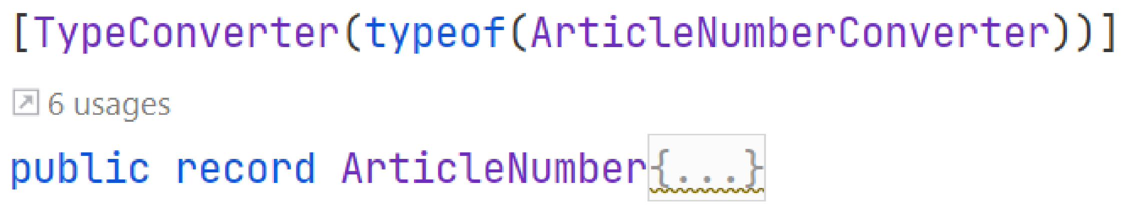
```
var guid = (Guid)TypeDescriptor.GetConverter(typeof(Guid))
    .ConvertFrom(context: null, CultureInfo.CurrentCulture, value: "AAAAAAAA-D7B4-47ED-A9E7-AE18721D6F3D")!;

Console.WriteLine(guid);
// output: aaaaaaaa-d7b4-47ed-a9e7-ae18721d6f3d
```

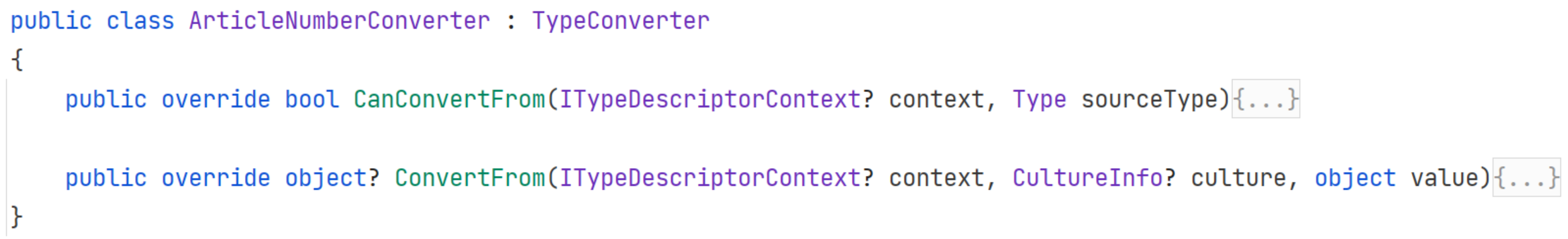
SimpleTypeModelBinder

– Utilized to bind Guids

– Can be extended



```
[TypeConverter(typeof(ArticleNumberConverter))]
6 usages
public record ArticleNumber{...}
```

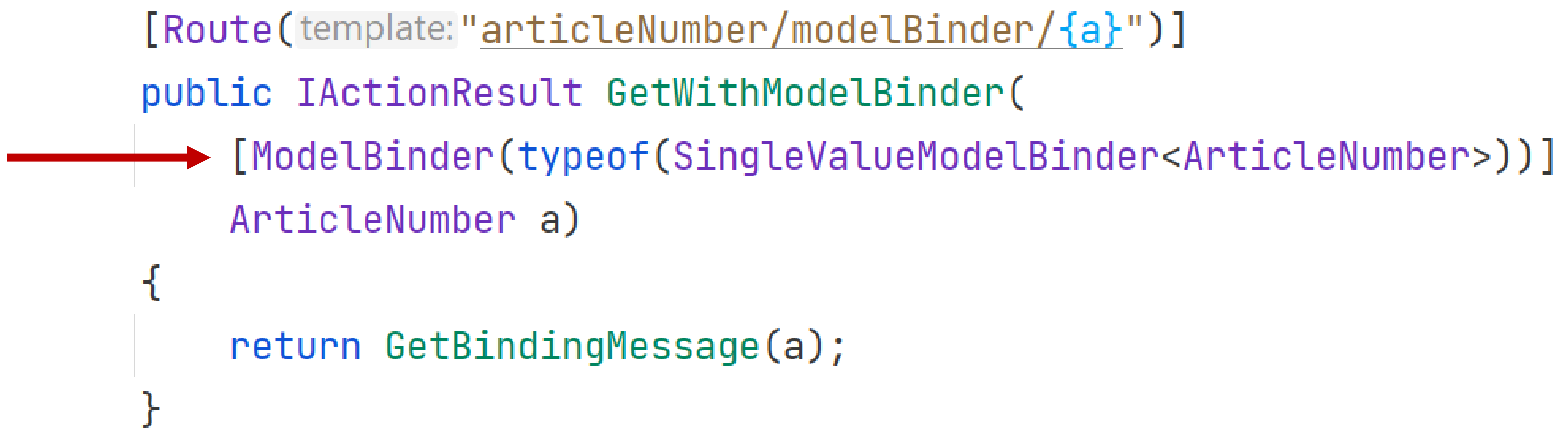


```
public class ArticleNumberConverter : TypeConverter
{
    public override bool CanConvertFrom(ITypeDescriptorContext? context, Type sourceType){...}

    public override object? ConvertFrom(ITypeDescriptorContext? context, CultureInfo? culture, object value){...}
}
```

ModelBinder

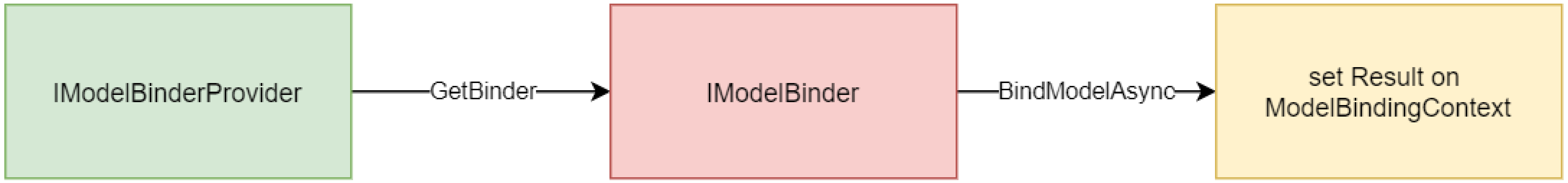
- Specify ModelBinder explicit
  - Based on ModelBindingContext



```
[Route(template: "articleNumber/modelBinder/{a}")]
public IActionResult GetWithModelBinder(
    [ModelBinder(typeof(SingleValueModelBinder<ArticleNumber>))]
    ArticleNumber a)
{
    return GetBindingMessage(a);
}
```

ModelBinderProvider

- Provides an IModelBinder
  - Based on ModelBinderProviderContext



SingleValueModelBinderProvider

- Provides a SingleValueModelBinder
  - When type is supported

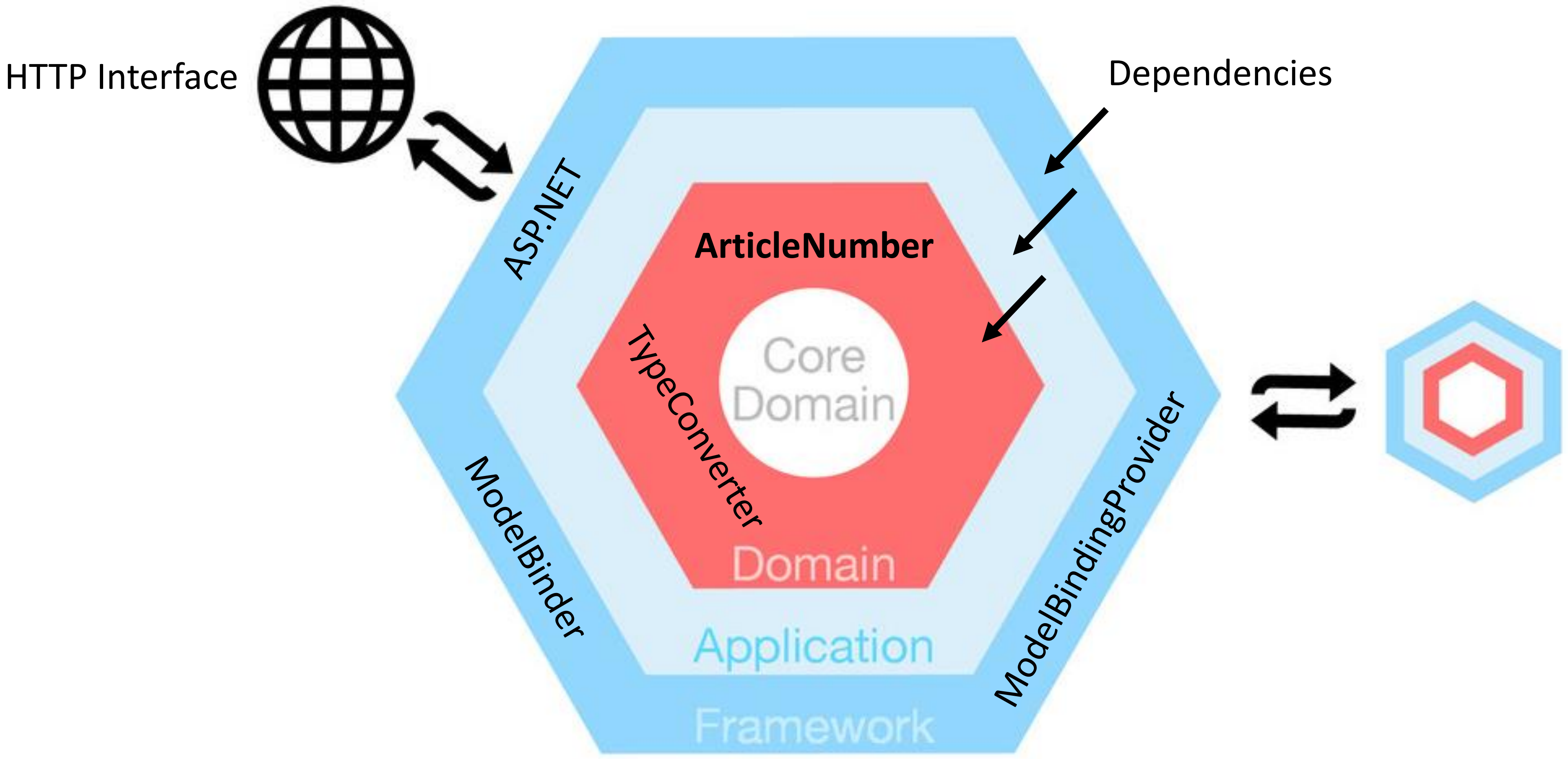
- Supported types configured in setup

ReflectionModelBinderProvider

- Provides a SingleValueModelBinder
  - When type has single ctor with single argument

- Supported types evaluated at runtime

Architecture



Summary

– Bind primitives and convert inside route
– TypeConverter & TypeDescriptor
– Explicit via ModelBinder attribute
– ModelBinderProvider

– Be aware of the introduced complexity

Sources

- ASP.NET full source code: https://github.com/dotnet/aspnetcore
- SimpleTypeModelBinder: https://github.com/dotnet/aspnetcore/blob/main/src/Mvc/Mvc.Core/src/ModelBinding/Binders/SimpleTypeModelBinder.cs
- Implementing ModelBinderProvider: https://docs.microsoft.com/en-us/aspnet/core/mvc/advanced/custom-model-binding?view=aspnetcore-6.0#implementing-a-modelbinderprovider
- TypeConverter: https://docs.microsoft.com/en-us/dotnet/api/system.componentmodel.typeconverter
- TypeDescriptor: https://docs.microsoft.com/en-us/dotnet/api/system.componentmodel.typedescriptor
- GuidConverter implementation from the dotnet runtime: https://github.com/dotnet/runtime/blob/main/src/libraries/System.ComponentModel.TypeConverter/src/System/ComponentModel/GuidConverter.cs
- Domain specific value types: https://freecontent.manning.com/domain-primitives-what-they-are-and-how-you-can-use-them-to-make-more-secure-software/
- Hexagonal Architecture: https://fideloper.com/hexagonal-architecture