

Manipulating Resources and Validating Input



Kevin Dockx

Architect

@Kevindockx | www.kevindockx.com



Coming Up



Manipulating resources

- Creating resources
- Updating resources
- Deleting resources

Validating input

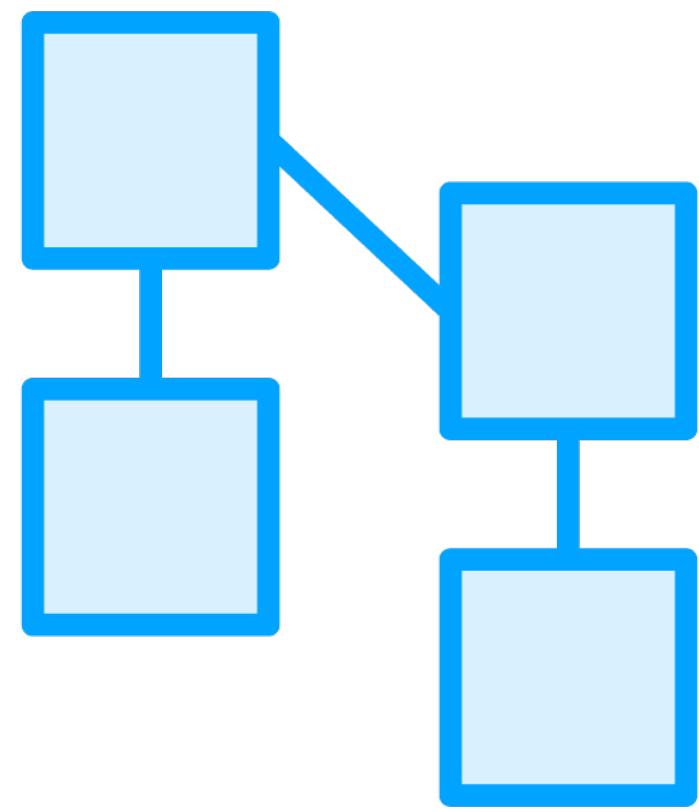


Passing Data to the API

**Data can be passed to an API by
various means**

**Binding source attributes tell the model
binding engine where to find the
binding source**





[FromBody]

- Request body

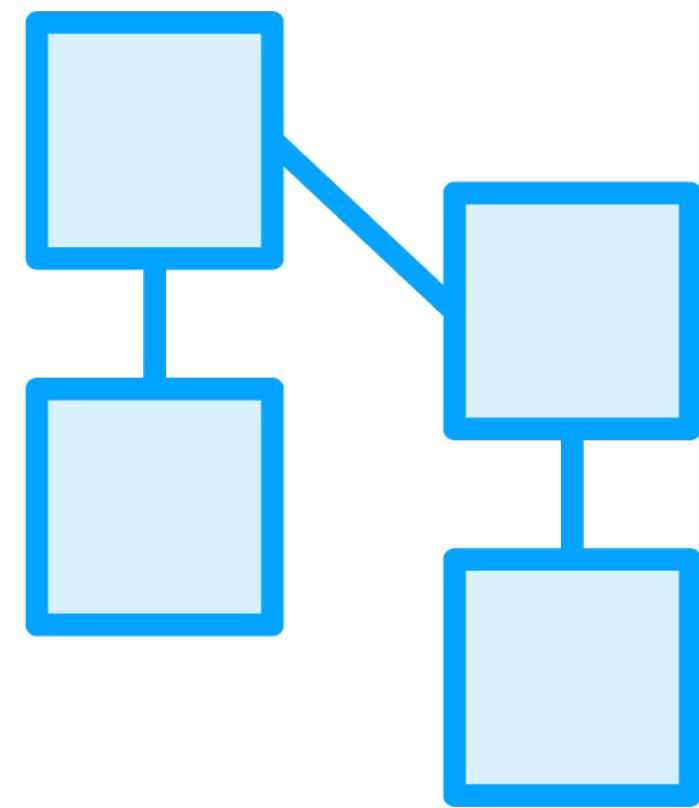
[FromForm]

- Form data in the request body

[FromHeader]

- Request header





[FromQuery]

- Query string parameters

[FromRoute]

- Route data from the current request

[FromServices]

- The service(s) injected as action parameter

[AsParameters]

- Method parameters



```
public ActionResult<CityDto> GetCity(  
    [FromRoute] int cityId)
```

Passing Data to the API

Use binding source attributes to explicitly state where the action parameter should be bound from

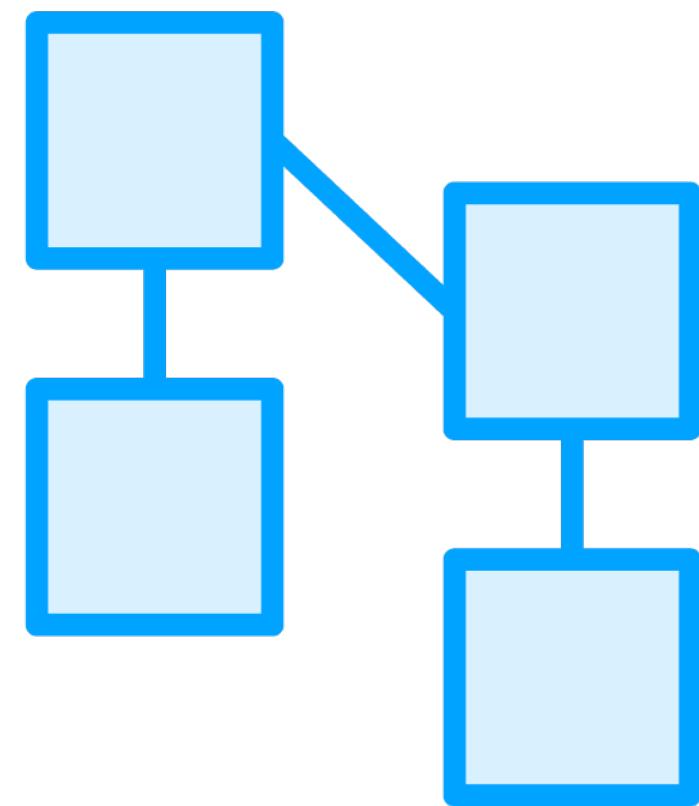


Passing Data to the API

By default, ASP.NET Core attempts to use the complex object model binder

The `[ApiController]` attribute changes the rules to better fit APIs





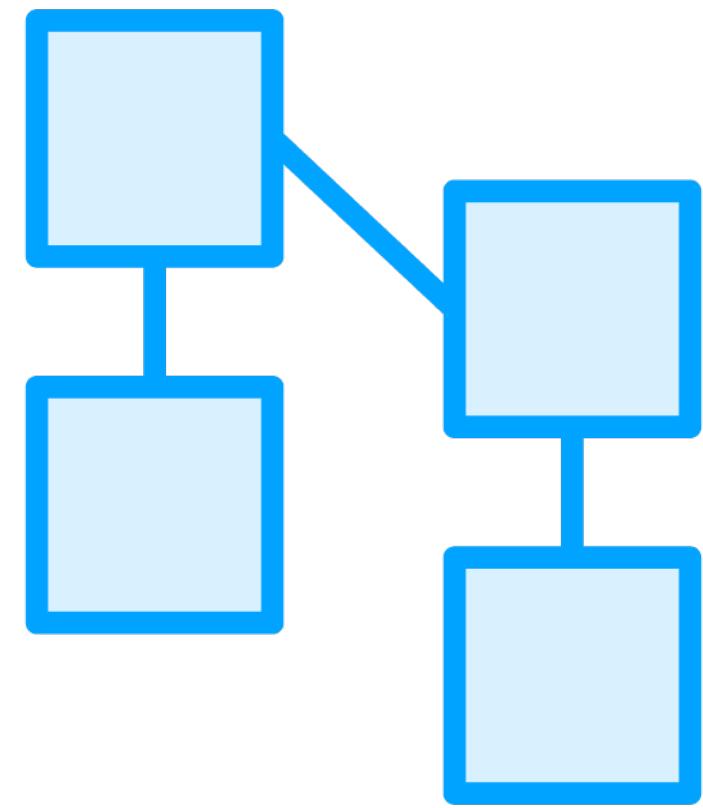
[FromBody]

- Inferred for complex types

[FromForm]

- Inferred for action parameters of type `IFormFile` and `IFormFileCollection`





[FromRoute]

- Inferred for any action parameter name matching a parameter in the route template

[FromQuery]

- Inferred for any other action parameters



Demo



Creating a resource



Demo



Validating input



A Validation Alternative

The built-in approach is ok for simple use cases

For complex rules, consider FluentValidation

- <https://fluentvalidation.net/>



Demo



Updating a resource



Partially Updating a Resource

Json Patch (RFC 6902)

- <https://tools.ietf.org/html/rfc6902>

**Describes a document structure for
expressing a sequence of operations to apply
to a Json document**



```
[  
  {  
    "op": "replace",  
    "path": "/name",  
    "value": "new name"  
  },  
  {  
    "op": "replace",  
    "path": "/description",  
    "value": "new description"  
  }  
]
```

◀ **array of operations**

◀ **“replace” operation**

◀ **“name” property gets value “new name”**

◀ **“replace” operation**

◀ **“description” property gets value “new description”**



Demo



Adding support for `JsonPatchDocument`



Demo



Partially updating a resource



Demo



Deleting a resource



Demo



Creating a file



Summary



Use POST for creating a resource

- 201 Created
- Header: Content-Type

Validation

- Data annotations
- ModelState



Summary



Use PUT for full updates, PATCH for partial updates

- JsonPatch standard
- 204 NoContent or 200 Ok

DELETE is for deleting resources

- 204 NoContent



Up Next:

Working with Services and Dependency Injection

