Getting Resources

Kevin Dockx
ARCHITECT

@KevinDockx https://www.kevindockx.com

Coming Up

Improving our architecture

IActionResult vs. ActionResult<T>

Working with AutoMapper

Parent/child relationships

Dealing with faults

Supporting HEAD

Outer Facing vs. Entity Model

The entity model represents database rows as objects

The outer facing model represents what's sent over the wire

Outer Facing Model vs. Entity Model

Outer facing model (Author)

```
Guid Id
string FirstName
string LastName
int Age
```

Entity model (Author)

```
Guid Id
string FirstName
string LastName
DateTimeOffset DateOfBirth
```

Outer Facing Model vs. Entity Model

Outer facing model (Author)

```
Guid Id

string Name

int Age
```

Entity model (Author)

```
Guid Id

string FirstName

string LastName

DateTimeOffset DateOfBirth
```

Outer Facing Model vs. Entity Model

Outer facing model (Author)

```
Guid Id

string Name

int Age

float Royalties
```

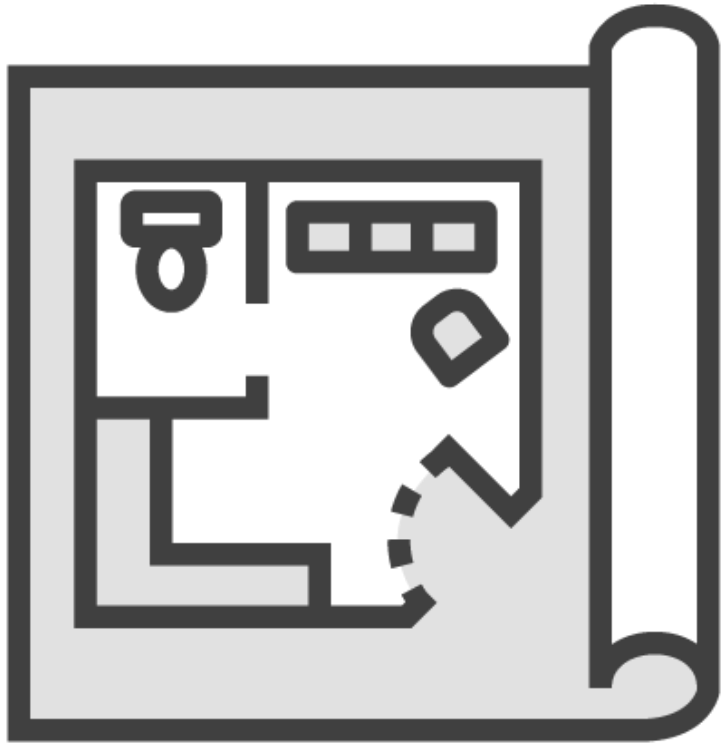
Entity model (Author)

```
Guid Id

string FirstName

string LastName

DateTimeOffset DateOfBirth
```



Separating outer and entity models leads to more robust, reliable and evolvable code

Demo

Separating entity model and outer facing model

Demo

Improving action return types with ActionResult<T>

Demo

Adding AutoMapper to our project

Demo

Using AutoMapper

Demo

Working with parent/child relationships

Demo

Returning a single child resource

Demo

Handling faults

Interacting with Resources through HTTP Methods

HTTP Method	Request Payload	Sample URI	Response Payload
GET	-	/api/authors /api/authors/{authorId}	author collection single author
POST	single author	/api/authors	single author
PUT	single author	/api/authors/{authorId}	single author or empty
PATCH	JsonPatchDocument on author	/api/authors/{authorId}	single author or empty
DELETE	-	/api/authors/{authorId}	-
HEAD	-	/api/authors /api/authors/{authorId}	-
OPTIONS	-	/api/...	-

Supporting HEAD

HEAD is identical to GET, with the notable difference that the API shouldn't return a response body

It can be used to obtain information on the resource

Demo

Supporting HEAD

Summary

The entity model represents database rows as objects

The outer facing model represents what's sent over the wire

Separating these robust, reliable and evolvable code

Summary

Use ActionResult<T> when possible so other pieces of code can infer the action's expected return type

Dealing with parent/child relationships often means additional checks to return the correct status code

- It's nevertheless important to execute those so the consumer knows what's going on

Summary

Avoid returning stack traces when faults happen by configuring the ExceptionHandler

Use HEAD to obtain information on a resource