Designing the Outer Facing Contract

Kevin Dockx
Architect

@KevinDockx https://www.kevindockx.com

Coming Up

- Structuring the outer facing contract
- URI design guidelines
- Routing
- The importance of status codes
- Content negotiation
- Outer facing model vs. entity model

Designing the Outer Facing Contract

Resource identifier
http://host/api/authors

HTTP method
https://datatracker.ietf.org/doc/html/rfc9110

Payload (representation: media types)

Designing the Outer Facing Contract

Resource identifier
http://host/api/authors

HTTP method
https://datatracker.ietf
.org/doc/html/rfc9110

Payload
(representation:
media types)

Resource Naming Guidelines

Nouns: things, not actions

- ~~`api/getauthors`~~
- GET `api/authors`
- GET `api/authors/{authorId}`

Convey meaning when choosing nouns

Resource Naming Guidelines

Follow through on this principle for predictability

- ~~`api/something/somethingelse/employees`~~
- `api/employees`
- ~~`api/id/employees`~~
- `api/employees/{employeeId}`

Resource Naming Guidelines

Represent hierarchy when naming resources

- `api/authors/{authorId}/courses`
- `api/authors/{authorId}/courses/{courseId}`

Resource Naming Guidelines

Filters, sorting orders, … aren't resources

- ~~`api/authors/orderby/name`~~
- `api/authors?orderby=name`

Resource Naming Guidelines

Sometimes, RPC-style calls don't easily map to pluralized resource names

- ~~`api/authors/{authorId}/pagetotals`~~
- ~~`api/authorpagetotals/{id}`~~
- `api/authors/{authorId}/totalamountofpages`

Routing

Routing matches a request URI to an action on a controller

```
// set up the request pipeline

...

app.MapControllers();

...
```

Endpoint Routing

Add endpoints for controller actions to the `IEndpointRouteBuilder` without specifying any routes
Specify routes via route attributes

Designing the Outer Facing Contract

Resource identifier
http://host/api/authors

HTTP method
https://datatracker.ietf.org/doc/html/rfc9110

Payload
(representation:
media types)

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/...	-

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/...	-

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/...	-

Demo

Adhering to URI guidelines

Learning why Status Codes are Important

Status codes tell the consumer of the API

- Whether or not the request worked out as expected
- What is responsible for a failed request

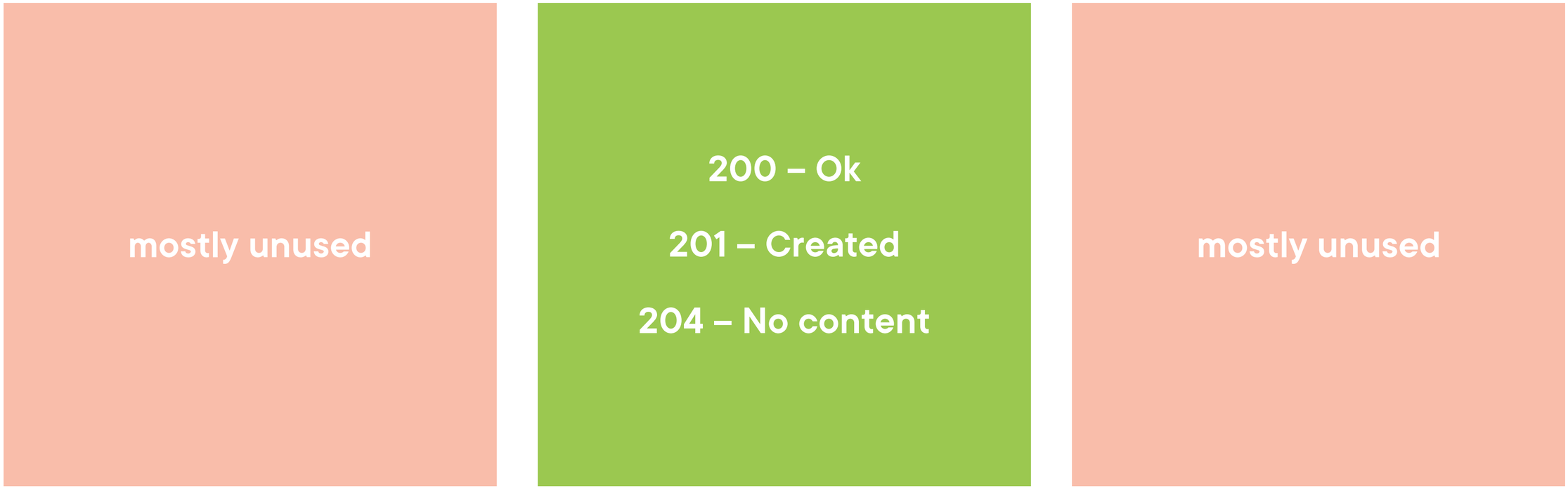
Learning why Status Codes are Important

Be as specific as possible

- API consumers are typically non-human

Be especially specific in regards to reporting who/what is responsible for a mistake

Learning why Status Codes are Important



Level 100
Informational

Level 200
Success

Level 300
Redirection

Learning why Status Codes are Important

400 – Bad request

401 – Unauthorized

403 – Forbidden

404 – Not found

405 – Method not allowed

406 – Not acceptable

409 - Conflict

415 – Unsupported media type

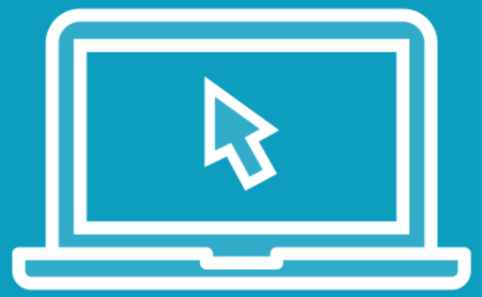
422 – Unprocessable entity

500 – Internal server error

Level 400
Client mistakes

Level 500
Server mistakes

Demo



Returning correct status codes

Errors, Faults and API Availability

Errors	Faults
Consumer passes invalid data to the API, and the API correctly rejects this	API fails to return a response to a valid request
Level 400 status codes	Level 500 status codes
Do not contribute to API availability	Do contribute to API availability

Demo

Handling faults and avoiding exposing implementation details

Designing the Outer Facing Contract

Resource identifier
http://host/api/authors

HTTP method
https://datatracker.ietf
.org/doc/html/rfc9110

Payload (representation: media types)

Content negotiation

The process of selecting the best representation for a given response when there are multiple representations available

Working with Content Negotiation and Formatters

Media type is passed via the Accept header of the request

- application/json
- application/xml
- ...

Working with Content Negotiation and Formatters

Returning a representation in a default format when no Accept header is included is acceptable

Returning a representation in a default format when the requested media type isn't available isn't acceptable

Return "406 – Not acceptable"

Working with Content Negotiation and Formatters

Output formatter

Deals with output

Media type: Accept header

Input formatter

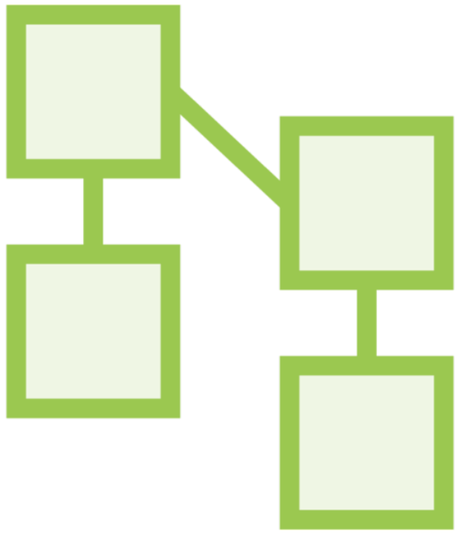
Deals with input

Media type: Content-type header

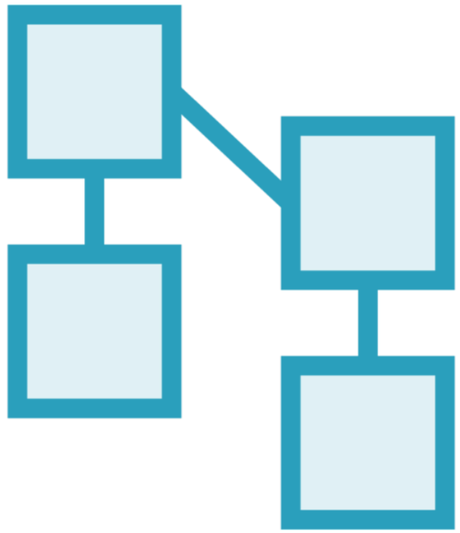
Demo

Supporting XML

Outer Facing Model 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 (AuthorDto)

```
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 (AuthorDto)

```
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 (AuthorDto)

```
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

Important statement

Supporting HEAD

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

Outer facing contract

- Resource identifiers
- HTTP methods
- Optional payload (media type)

Summary

Resource identifiers

- Use pluralized nouns that convey meaning
- Represent model hierarchy
- Be consistent

Summary

HTTP methods

- GET
- POST
- PUT/PATCH
- DELETE
- HEAD
- OPTIONS

Summary

Routing matches a request URI to an action on a controller

Summary

Content negotiation is the process of selecting the best representation for a given response when there are multiple representations available

Summary

Status codes

- Level 200: success
- Level 400: errors (client)
- Level 500: faults (server)

Summary

Don’t return stack traces to the consumers of the API – they have no use for them

Summary

The entity model represents database rows as objects, the outer facing model represents what's sent over the wire

- Separating these leads to more robust, reliable and evolvable code

Summary

Use HEAD to obtain information on a resource

Up Next:
Manipulating Resources