Module Four

REST Design Fundamentals

Agenda

This is basically the REST …

- Idempotency
- Content negotiation
- HATEOAS
- Exceptions

Indempotency

More than just a fancy word

Idempotency

Idem (same) *potency* (power)

- Certain operations must be idempotent
- Multiple calls affect the data the same way
  - Doesn't mean the result is the same as the state may change
  - Does mean there are no side effects on the server
- Example: delete, first time 200/204, subsequent 404, but effect on server same

Idempotency and REST

Method	Description	Idempotent?
GET	Returns value	YES
PUT	Replace object: always replaces with the same item	YES
DELETE	Delete an object: always results in object “not there”	YES
POST	New object: only new once!	NO
HEAD	Return metadata	YES
PATCH	Partial update, side effects	NO
OPTIONS	Information about request	YES

Content

Negotiate carefully

Content Negotiation

"This is what you want, this is what you get."

- Request sends "Accept." Not a requirement but a "hint."
- Server *should* return, if it can, but is not required to
- Default response should be application/json
- All services should support JSON at a minimum
- Alternative: encode request in URL
  - /resource/result.json
  - /resource/result.xml

Demo

Content Negotiation

XML

Demo

Content Negotiation

Extensions

Demo

Content Negotiation

Custom Content

HATEOAS

(Don’t hate!)

HATEOAS

Don’t hate the HATEOAS

- Hypertext As The Engine Of Application State
- Every response *explicitly* points to possible next steps
- Links w/ “rel” and “href”


```
{
    "value": {
        "id": 1,
        "name": "Item1",
        "isComplete": false
    },
    "links": [
        {
            "href": "http://localhost:5000/api/Todo/1",
            "rel": "self",
            "method": "GET"
        },
        {
            "href": "http://localhost:5000/api/Todo/1",
            "rel": "put-todo",
            "method": "GET"
        },
        {
            "href": "http://localhost:5000/api/Todo/1",
            "rel": "delete-todo",
            "method": "GET"
        }
    ]
}
```

Demo

Adding a little HATEOAS

Exceptions

"I take exception to that…"

Exception Guidelines

Except when …

- Should always use appropriate response codes
  - 4xx for application-aware errors (not found, invalid, etc.)
  - 5xx for faults (i.e. lost database connectivity, couldn't deserialize, etc.)
- Return a consistent error response object

Error Response

ErrorResponse : Object

Property	Type	Required	Description
`error`	Error	✔	The error object.

Error : Object

Property	Type	Required	Description
`code`	String (enumerated)	✔	One of a server-defined set of error codes.
`message`	String	✔	A human-readable representation of the error.
`target`	String		The target of the error.
`details`	Error[]		An array of details about specific errors that led to this reported error.
`innererror`	InnerError		An object containing more specific information than the current object about the error.

InnerError : Object

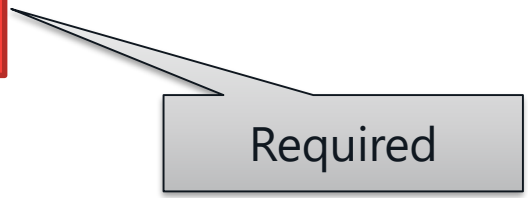
Property	Type	Required	Description
`code`	String		A more specific error code than was provided by the containing error.
`innererror`	InnerError		An object containing more specific information than the current object about the error.

Source: Microsoft API Guidelines
https://github.com/Microsoft/api-guidelines/blob/master/Guidelines.md

PASS

Error Response


```
{
  "error": {
    "code": "BadArgument",
    "message": "Previous passwords may not be reused",
    "innererror": {
      "code": "PasswordError",
      "innererror": {
        "code": "PasswordDoesNotMeetPolicy",
        "minLength": "6",
        "maxLength": "64",
        "characterTypes": ["lowerCase","upperCase","number","symbol"],
        "minDistinctCharacterTypes": "2",
        "innererror": {
          "code": "PasswordReuseNotAllowed"
        }
      }
    }
  }
}
```

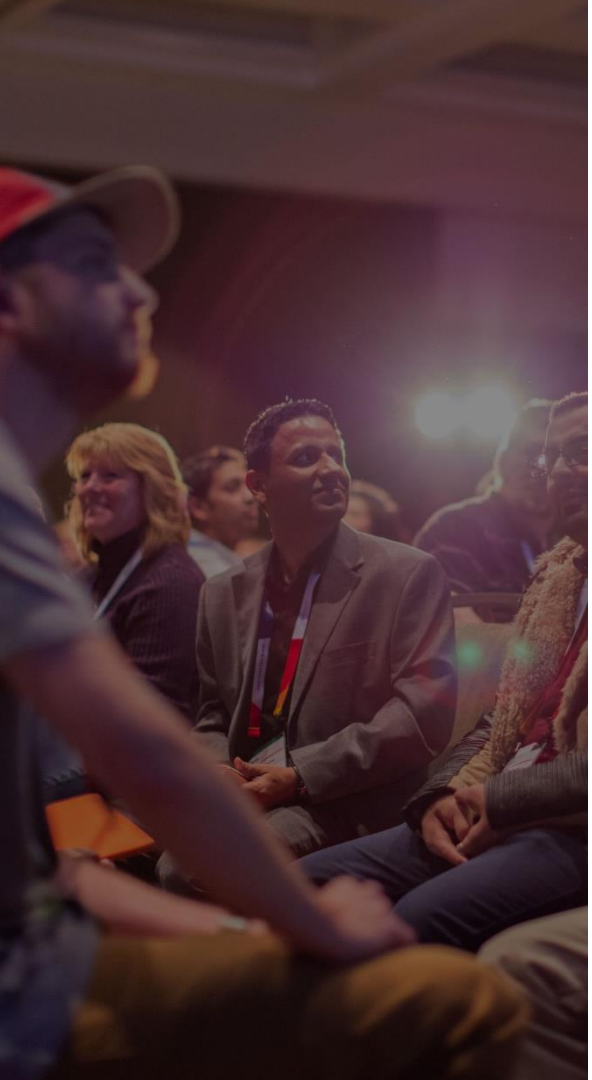
Demo

Exceptions in Practice

Recap

You learned …

- What *idempotency* is and why it should be followed
- How to properly negotiate content
- How to create custom content types
- What HATEOAS is, and how to implement it
- The appropriate way to deal with exceptions in web APIs



Thank You

Learn more from Jeremy Likness

@JeremyLikness

Jeremy.Likness@microsoft.com

PASS