Working with Derived Types, Open Types and Batch Processing

KEVIN DOCKX
ARCHITECT

@KevinDockx https://www.kevindockx.com

Coming Up

Derived types

Open types

Batch processing

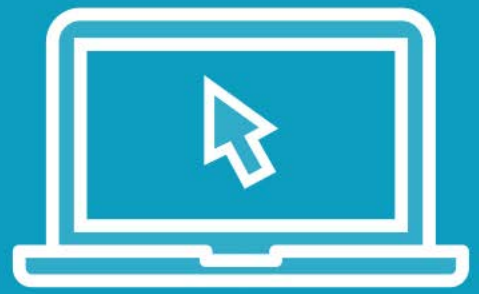
Working with Derived Types

Address derived type directly

Path segment: qualified name

Result set will only contain derived type instances

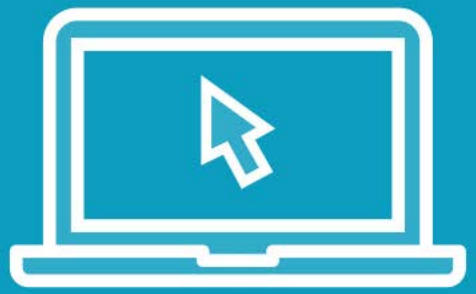
Demo



Working with Derived Types

- GET root/EntitySet/QualifiedNameOfDerivedType
- GET root/EntitySet(entityKey)/QualifiedNameOfDerivedType

Demo



Manipulating Derived Types

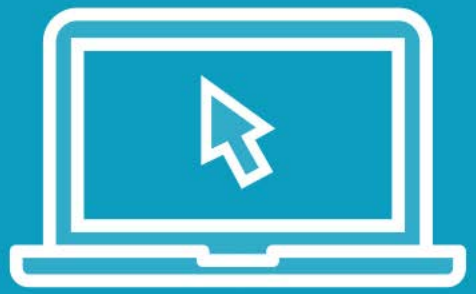
- POST root/EntitySet
- PATCH root/EntitySet(entityKey)/QualifiedNameOfDerivedType
- DELETE root/EntitySet(entityKey)/QualifiedNameOfDerivedType

Working with Open Types

Allows persisting undeclared properties

Model: Dictionary<string, object>

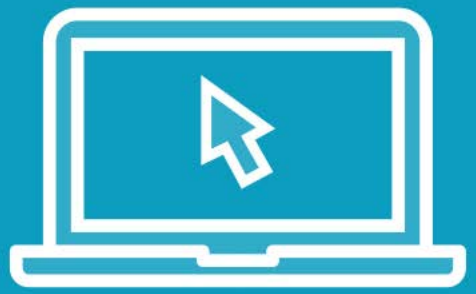
Demo



Working with Open Types

- Model: Dictionary<string,object>

Demo



Manipulating Open Types

- POST root/EntitySet
- PATCH root/EntitySet(entityKey)

Grouping Multiple Operations in a Single Request

Multiple operations in one payload

- Multipart MIME v1 message

POST to root/$batch

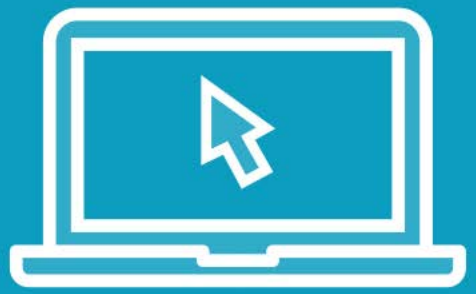
Grouping Multiple Operations in a Single Request

Content-Type: multipart/mixed + boundary specification

Individual requests

- HTTP method, URI + HTTP version
- Content-Type: application/http
- Content-Transfer-Encoding: binary
- *Request Body*

Demo



Grouping Multiple Operations in a Single Request

- POST root/$batch
- Multipart MIME v1 message

Summary

Derived types

- Qualified name

Open types

- Persist undeclared properties

Batch requests

- Multipart MIME v1 message