Designing Versioning For Your API

Shawn Wildermuth
MICROSOFT MVP, INSTRUCTOR AND FILMMAKER

@shawnwildermuth https://wilderminds.com

Agenda

Versioning Your API

- Should you version your API?
- URI Versioning Strategies
- Content Versioning

Should You Version Your API?

- Once you publish, it's set in stone
- Users rely on the API not changing
- But requirements will change

Evolve the API without breaking clients

- API Version isn't Product Version
- Don't tie them together

API versioning is harder

- Needs to support both new & old
- Side-by-side deployment isn't feasible

Lots of Ways to Version an API

- Not all are recommended
- Find a mechanism that works for you
- You're serving your clients...
- ...not yourselves

```
// URI Path
https://foo.org/api/v2/Customers
```

Versioning in the URI Path

Pros:

- Very clear to clients where the version is handled

Cons:

- Every version needs to change URIs, can be brittle

```
// Query String
https://foo.org/api/Customers?v=2.0
```

Versioning with Query String

Pros:

- Versioning is optionally included (can use default version)

Cons:

- Too easy for clients to miss needing the version

```
GET /api/camps HTTP/1.1
Host: localhost:44388
Content-Type: application/json
X-Version: 2.0
```

Versioning with Headers

Pros:

- Separates versioning from the rest of the API

Cons:

- Requires more sophisticated developer to manipulate headers

```
GET /api/camps HTTP/1.1
Host: localhost:44388
Content-Type: application/json
Accept: application/json;version=2.0
```

Versioning with Accept Header

Pros:

- No need to create your own custom header

Cons:

- Even less discoverable than query strings

```
GET /api/camps HTTP/1.1
Host: localhost:44388
Content-Type: application/vnd.yourapp.camp.v1+json
Accept: application/vnd.yourapp.camp.v1+json
```

Versioning with Content Type

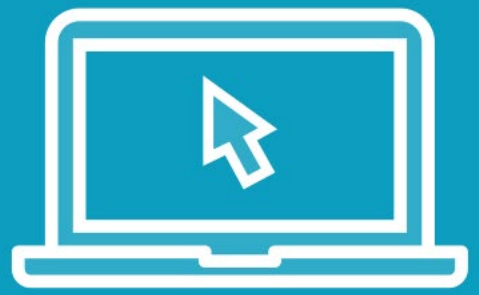
Pros:

- Can version the payload as well as the API call itself

Cons:

- Requires a lot more development maturity to create and maintain

Demo



Versioning an API

What We've Learned

Versioning is critical to a mature, long lasting API

Deciding on which versioning scheme means fitting requirements

Content versioning is complex, but could benefit long lived APIs

Coming Up:
Locking Down Your API