Dealing with Variable Resource Presentations

Kevin Dockx
ARCHITECT

@KevinDockx www.kevindockx.com

Coming Up

Tightening the Contract Between Client and API

Using Vendor-specific Media Types

Media Type Strategies

Media Types and Versioning

Tightening the Contract Between Client and API

Resource Identifier (URI)

http://api/tours/{tourid}

HTTP Method

GET, POST, ...

Payload (Media Type)

Accept & Content-Type
application/json?

Tightening the Contract Between Client and API

GET api/tours/{tourid}

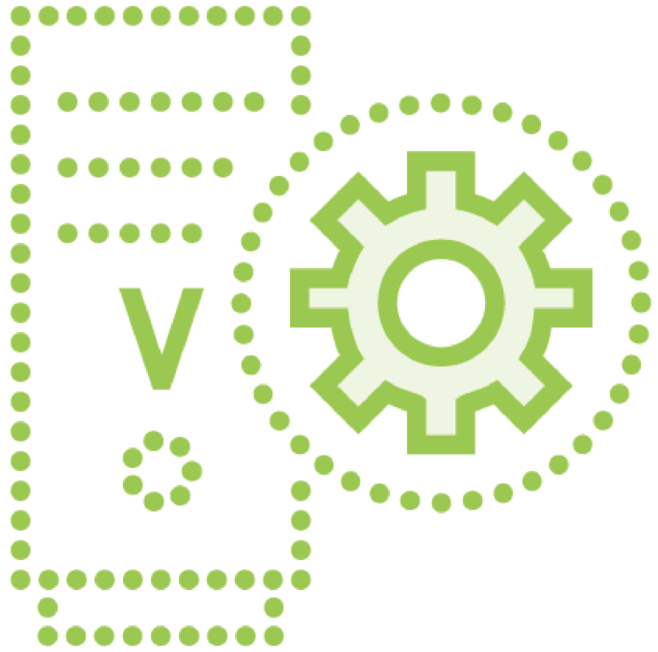
```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
  …
}
```

Accept: application/json?

These are different representations of the same resource

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
  …
 "estimatedProfits": 780906
}
```

Accept: application/json?



Tightening the Contract Between Client and API

POST api/tours

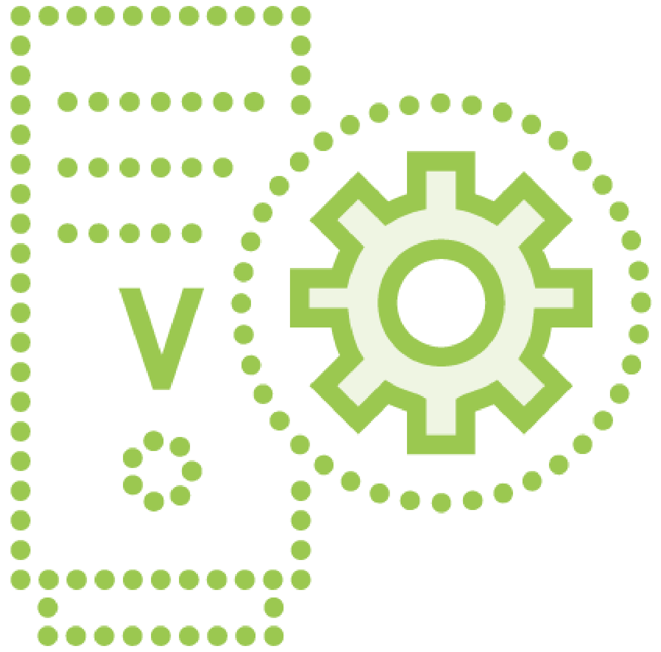
```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
 "tourManagerId": <id>
}
```

Content-Type: application/json?

These are different representations of the same resource

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
}
```

Content-Type: application/json?



Using Vendor-specific Media Types

“application/json” tells us something about the format, but not about the type.

Vendor-specific media types allow us to be explicit about the type, next to the format.

Using Vendor-specific Media Types

application/vnd.marvin.tourwithestimatedprofits+json

Using Vendor-specific Media Types

Top-level type

application/vnd.marvin.tourwithestimatedprofits+json

Using Vendor-specific Media Types

Top-level type

application/vnd.marvin.tourwithestimatedprofits+json

Vendor prefix

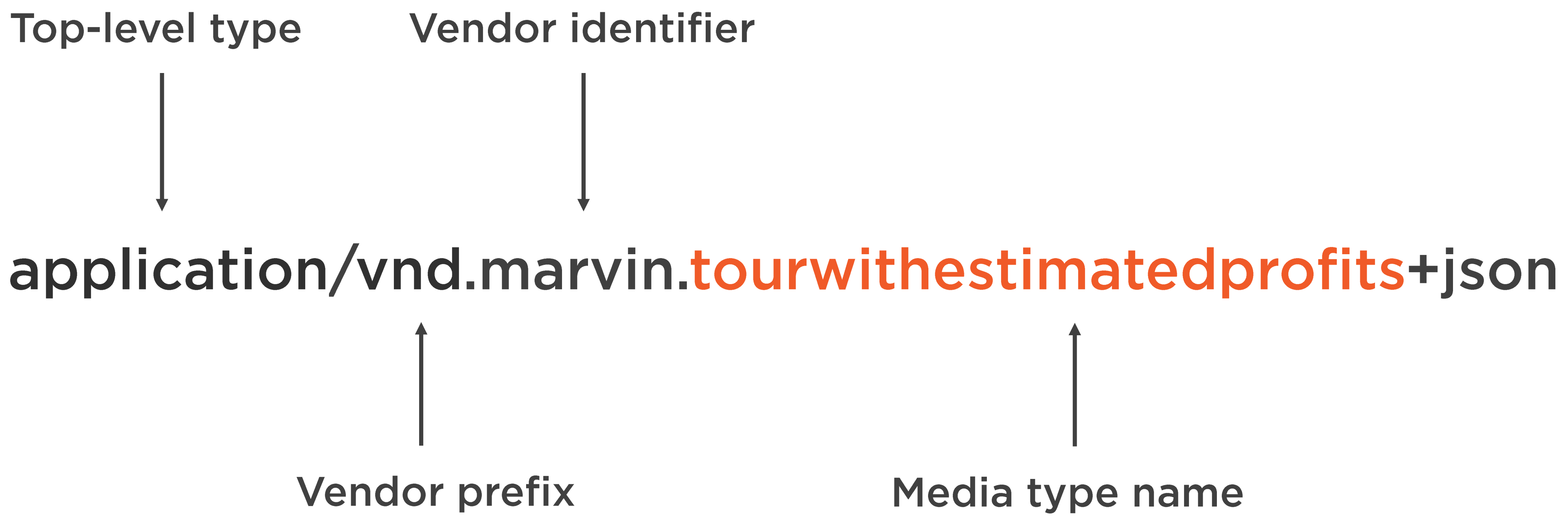
Using Vendor-specific Media Types

Top-level type Vendor identifier

application/vnd.marvin.tourwithestimatedprofits+json

Vendor prefix

Using Vendor-specific Media Types



Using Vendor-specific Media Types

Top-level type **Vendor identifier** **Suffix**

application/vnd.marvin.tourwithestimatedprofits+json

Vendor prefix **Media type name**

Using Vendor-specific Media Types

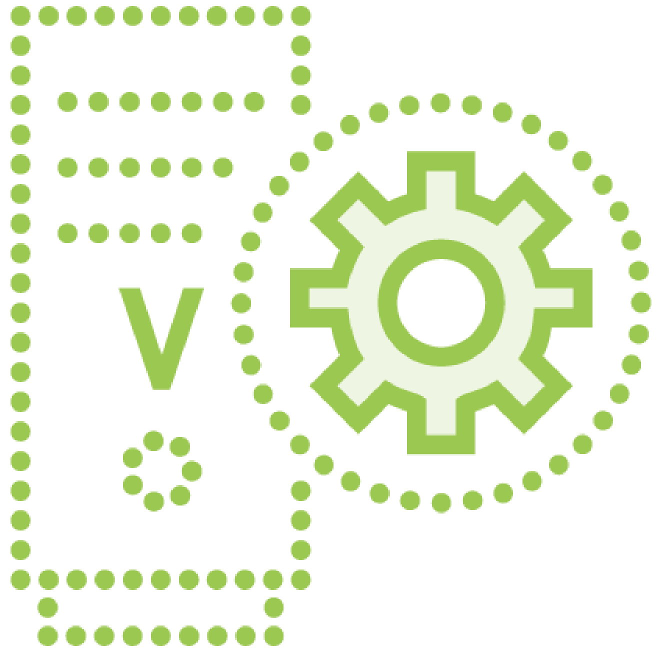
GET api/tours/{tourid}

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
  …
}
```

application/vnd.marvin.
tour+json

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
  …
 "estimatedProfits": 780906
}
```

application/vnd.marvin.
tourwithestimatedprofits
+json



Tightening the Contract Between Client and API

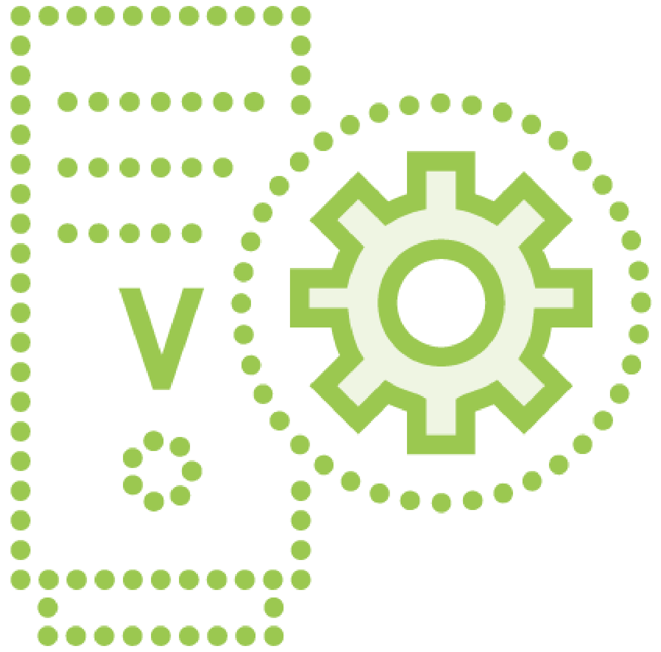
POST api/tours

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
 "tourManagerId": <id>
}
```

application/
vnd.marvin.
tourwithtourmanager+json

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
}
```

application/
vnd.marvin.tour+json



Demo

Creating an Action Constraint

Demo

Supporting Additional Media Types for Output

Demo

Getting Different Representations of the Same Resource (Server)

Demo

Getting Different Representations of the Same Resource (Client)

Differentiating Between Input and Output

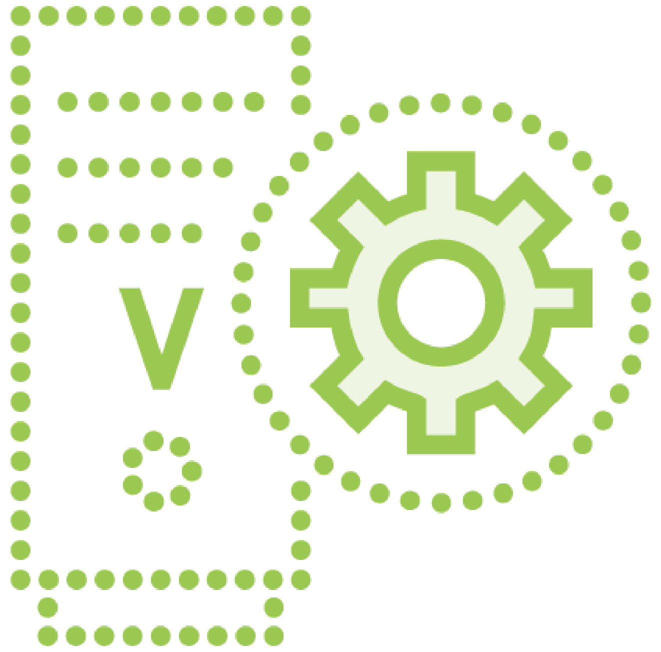
POST api/tours

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
 "tourManagerId": <id>
}
```

application/
vnd.marvin.
tourwithtourmanager+json

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
}
```

application/
vnd.marvin.tour+json



Differentiating Between Input and Output

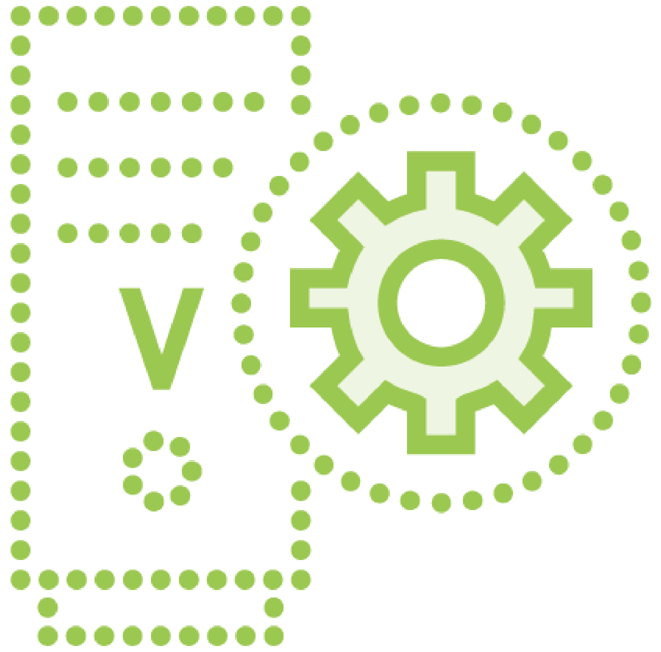
POST api/tours

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
 "tourManagerId": <id>
}
```

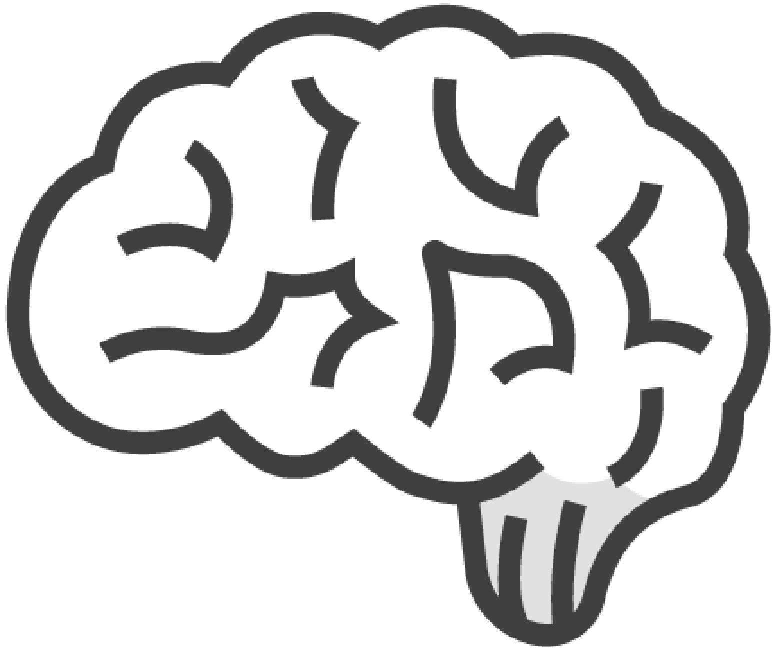
application/ vnd.marvin. tourwithtourmanager+json

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
}
```

application/ vnd.marvin.tour+json



Differentiating Between Input and Output



Media type for creating a tour

- Shouldn't have a band name
- Should have a band id
- Shouldn't have a tour id

Media type for creating a tour with manager

- Should have a manager id
- Shouldn't have a tour id

Both of these are used for creation – consistency in naming is important

Differentiating Between Input and Output

POST api/tours

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
 "tourManagerId": <id>
}
```

application/
vnd.marvin.
tourwithmanager
forcreation+json

```
{
 "title": "Villains World Tour",
 "band": "Queens of the Stone Age",
   …
}
```

application/
vnd.marvin.tour
forcreation+json

Improve Evolvability and Stability

The more precise the contract between client and API is, the less likely it becomes that, over time, they stop working together

Demo

Inputting Different Representations of the Same Resource (Server)

```
TourForCreation
Title
Description
StartDate
EndDate
BandId
```

```
TourWithManagerForCreation
Title
Description
StartDate
EndDate
BandId
ManagerId
```

```
Tour
TourId
Title
Description
StartDate
EndDate
Band
```

```
TourWithEstimatedProfits
: Tour
EstimatedProfits
```

```
TourAbstractBase
Title
Description
StartDate
EndDate
```

```
TourForCreation :
TourAbstractBase
BandId
```

```
Tour :
TourAbstractBase
TourId
```

```
TourWithManagerForCreation :
TourForCreation
ManagerId
```

```
TourWithEstimatedProfits :
Tour
EstimatedProfits
```

Demo

Inputting Different Representations of the Same Resource (Client)

Media Type Strategies (Part 1)

Always have a default (that works on application/json)

...but also describe the default as a custom media type

For other representations, use vendor-specific media types

Using Vendor-specific Media Types

GET api/tours/{tourid}

application/json
application/vnd.marvin.tour+json

```
{ tour }
```

application/
vnd.marvin.tourwithestimatedprofits+json

```
{ tour with
  estimated profits
}
```

POST api/tours

```
{ tour }
```

application/json
application/vnd.marvin.tourforcreation+json

POST api/tours

```
{ tour with tour manager }
```

application/vnd.marvin.tourwithtourmanagerforcreation+json

Demo

Supporting Defaults When Working with Vendor Specific Media Types

Media Type Strategies (Part 2)

Improve reliability and evolvability by sending an Accept header by default

Demo

Adding a Default Accept Header with an Http Interceptor

Media Types and Versioning

Don’t version your resource URIs

- api/v1/tours
- api/v2/tours

Use versioned media types instead

- application/vnd.marvin.tour.v1
- application/vnd.marvin.tour.v2

Summary

Contract

- Resource identifier
- Http method
- Payload / media type

The more precise the contract between client and API is, the less likely it becomes that, over time, they stop working together

- Improves stability and evolvability

Summary

“application/json” isn’t specific enough

- Tells us something about the data format, but not about the type

To tighten this contract, we can use vendor-specific media types

- These allow us to request different representations of the same resource

Summary

Media type strategies

- Always have a default (application/json)
- Also describe that default as a custom media type
- For other representations, use vendor-specific media types

The client must let the API know about the representation it wants to accept

- Improves reliability

Version media types, not URIs