apigee

Google Cloud

Web API Design: The Missing Link

Best Practices for Crafting Interfaces that Developers Love

Table of contents

Foreword 05

Introduction 07

Web APIs and REST 08

- The Job of the API Designer 08
- What is a web API? 08
- What is REST? 08

HTTP and REST: A Data-oriented Design Paradigm 10

- Why is a data-oriented approach useful? 10
- API Design Elements 12

Designing Representations 13

- Use JSON 13
- Keep your JSON simple 13
- Include Links 15
    - Why is this better? 17
    - Are URI templates still needed when you have links? 17
    - An analogy with the World Wide Web 18
    - Including links, step 2 20
    - A word of caution 21
    - How should I represent links in my resources? 22
    - Who uses links? 22
    - More Later 25

Designing URLs 26
- In URLs, nouns are good; verbs are bad 26
- Well-known URLs 26
- Designing entity URLs 26
  - Permalinks 27
  - The web is flat 28
  - Solutions to the renaming dilemma 29
  - Stability of types 32

Designing query URLs 32
  - Representing relationships in query URLs 34
  - Express relationships symmetrically in URLs and representations 35
  - A general model for query URLs 35
  - Path parameters, or matrix parameters 36
  - Filtering collections 36
  - What about responses that don't involve persistent resources? 37

More on Representation Design 40
- Include self-reference and kind properties 40
- Why are the self and the kind properties good ideas? 40
- How should I represent collections? 40
  - Paginated collections 43
- Custom resource types and using URLs for resource types 45
- Supporting multiple formats 46
- What about property names? 47
- Date and time formats 48

Chatty APIs 48

Pagination and partial response 48
  - Add optional fields in a comma-delimited list 49
  - Make it easy for application developers to paginate objects in a database 50

Handling Errors 51
- A message for people 53

Modeling Actions 53

Authentication 55

Complement with an SDK 55

Versioning 56
- Doing nothing for versioning 56
- Links and version identifiers in URLs make awkward bedfellows 57

Conclusion 60
- Resources 61

Appendix: Other Approaches to Representing Links 62

Foreword

The state of the art in web API design is constantly evolving as web APIs continue to become more important in business and in technology.

As a leader in API management, Apigee works with hundreds of customers to develop and manage a large number of APIs. By reflecting on our experiences and those of our customers and the industry at large, we have gained some insights into which API design innovations are bringing real benefits and becoming notable trends.

This book is our attempt to capture some of the significant trends in API design that we have seen emerge in the past couple of years. This book tries to be clear and simple, but it is not intended to be a beginner's guide to API design. If you are looking for more introductory material, you may wish to consult previous books from Apigee on the topic, like this one, or one of many other texts available.

Our earlier book used the example of a simple application for tracking dogs and their owners. In this book, we show how that example might be evolved to match more recent thinking about APIs. Here are two example resources from our earlier book:

```
https://dogtracker.com/dogs/12345678:
{       “id”: “12345678”,
        “kind”: “Dog”
        “name”: “Lassie”,
        “furColor”: “brown”,
        “owner”: “98765432”
}
```

and

```
https://dogtracker.com/persons/98765432:
{       “id”: “98765432”,
        “kind”: “Person”
        “name”: “Joe Carraclough”,
        “hairColor”: “brown”
}
```

Click here to download full PDF material