Making life a bit easier for Mobile App developers through better REST API Design

CodeMash 2016

Priya Rajagopal

Twitter: @rajagp
www.priyaontech.com

Invicara (Director, Mobile Development)
http://invicara.com

dailyKARMA (Principal Tech Consultant)
http://dailykarma.com

REST- A Quick Primer

- A set of architectural constraints , guidelines and best practices on how web services can be consumed by a client
  - It is an “architectural style”
  - Roy T. Fielding’s Doctoral Dissertation - Father of REST
  - Not a protocol
  - Not a standard, but standards-based
  - Not tied to specific data transfer protocol (although HTTP used almost universally)
  - Not tied to a specific data representation

Architectural Principles of REST

Academic ….

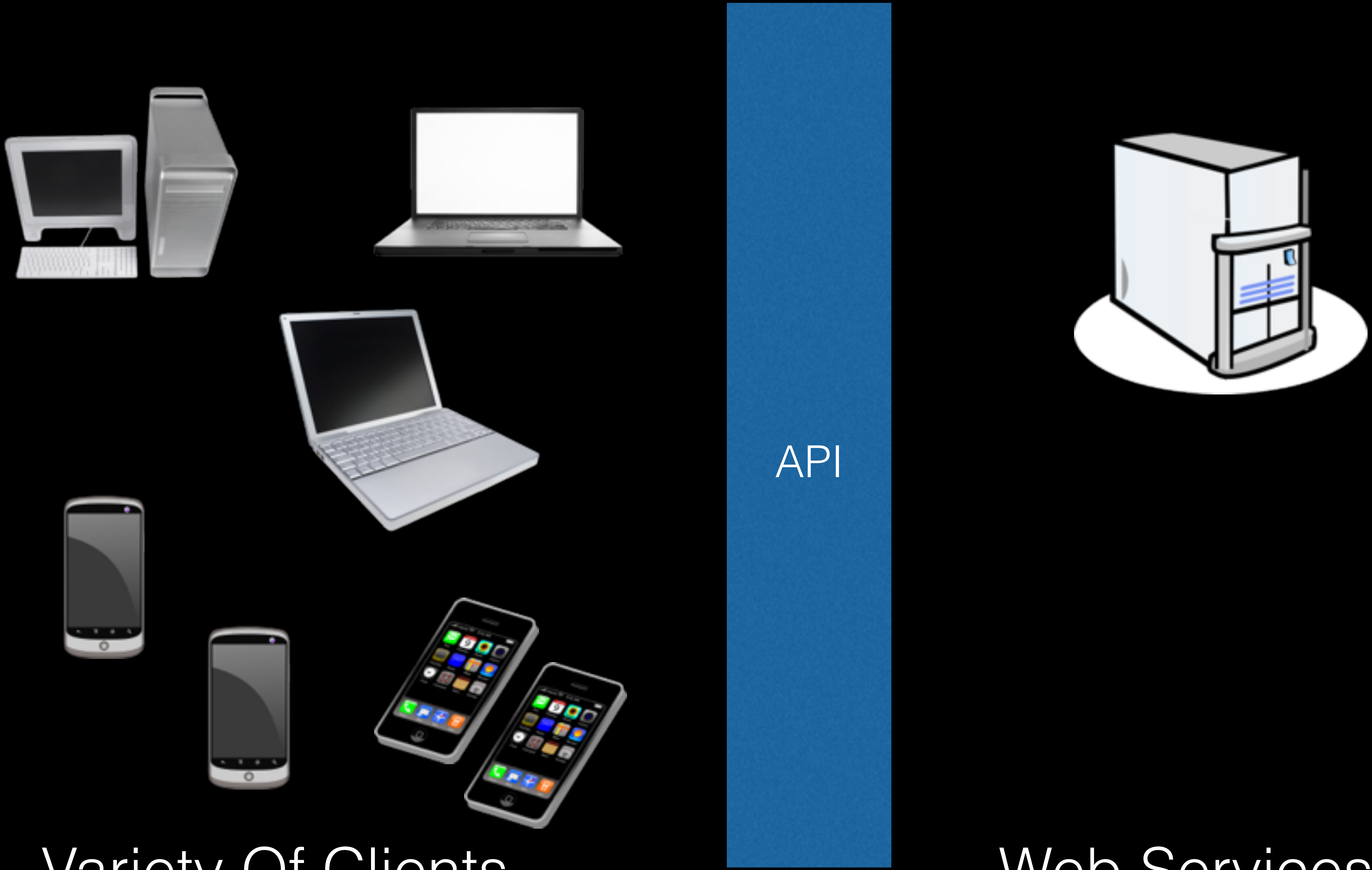
- Client-Server
- Uniquely Addressable Resources
- Stateless /self-contained
- Cacheable
- Layered/ Proxies

Architectual Principles of REST

Academic ….

• Client-Server	client-agnostic, separation of concerns
• Uniquely Addressable Resources	Loosely coupled clients, Independent evolution
• Stateless /self-contained	Scalability, Reliability
• Cacheable	Performance, Scalability
• Layered/ Proxies	Scalability

API Driven Development



API

Variety Of Clients

Web Services

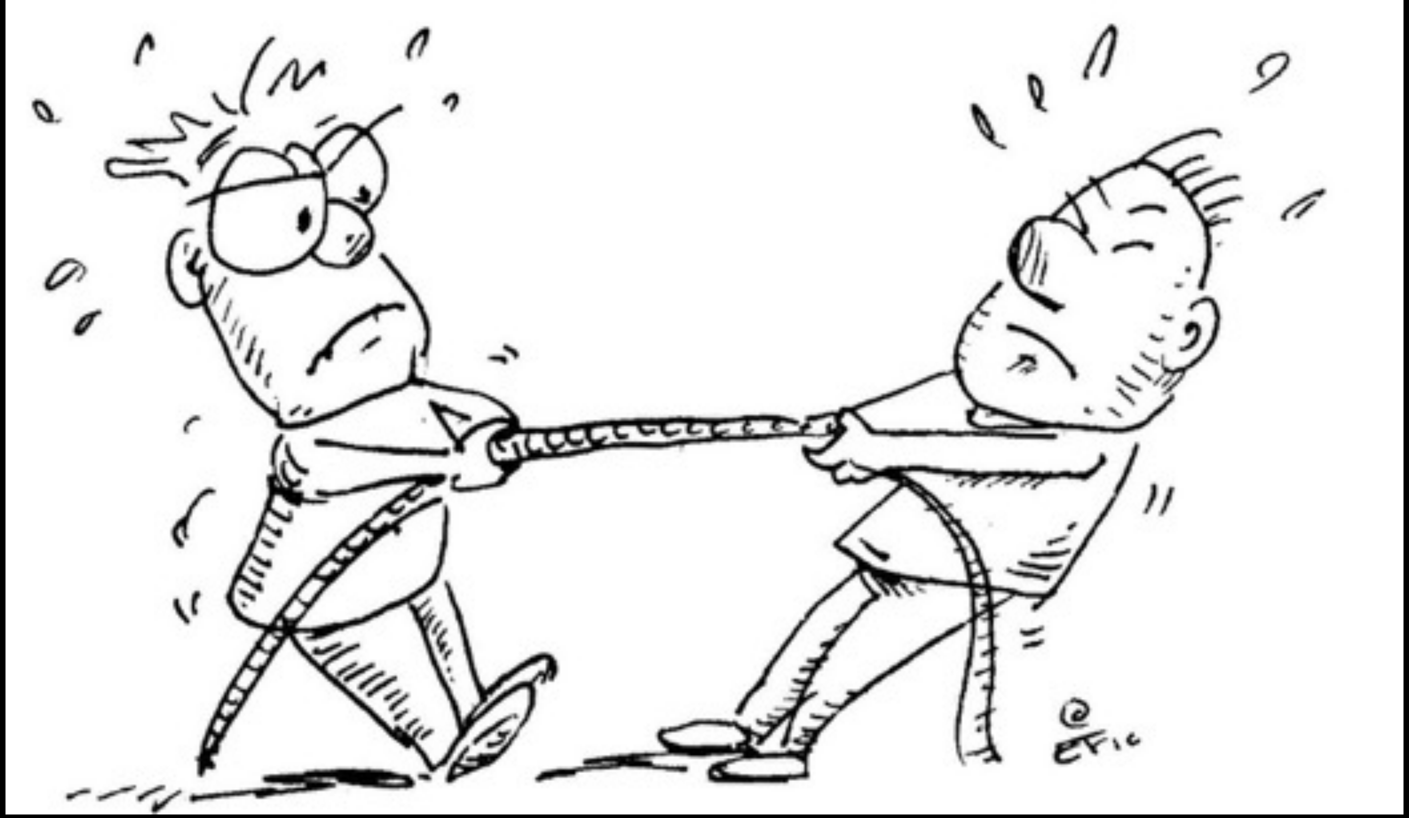
Icons:source:fastIcon.com.

Who Should Define the API?

"Architects"?

Client Teams?

Server Teams?

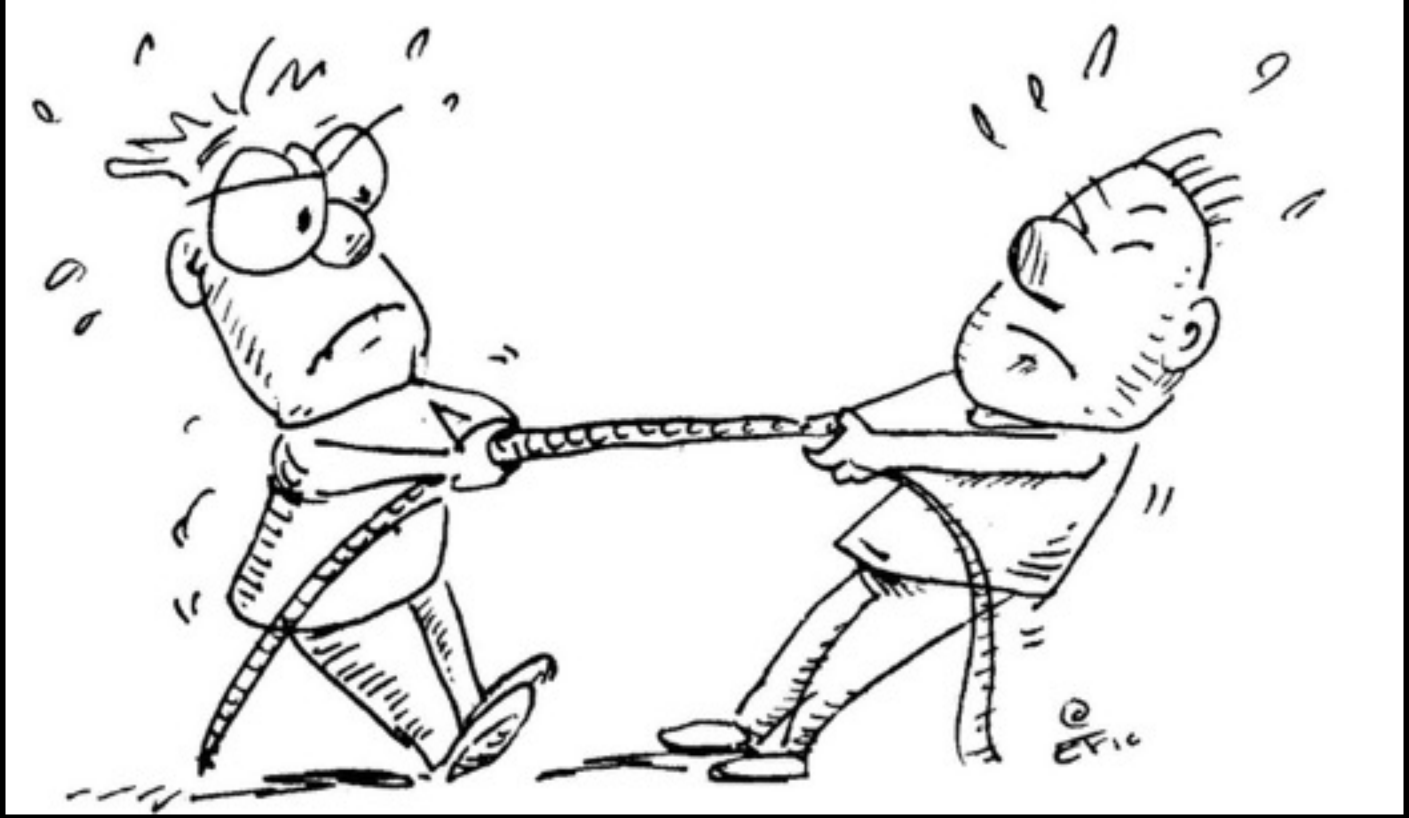


Who Should Define the API?

“Architects”?

Client Teams?

Server Teams?

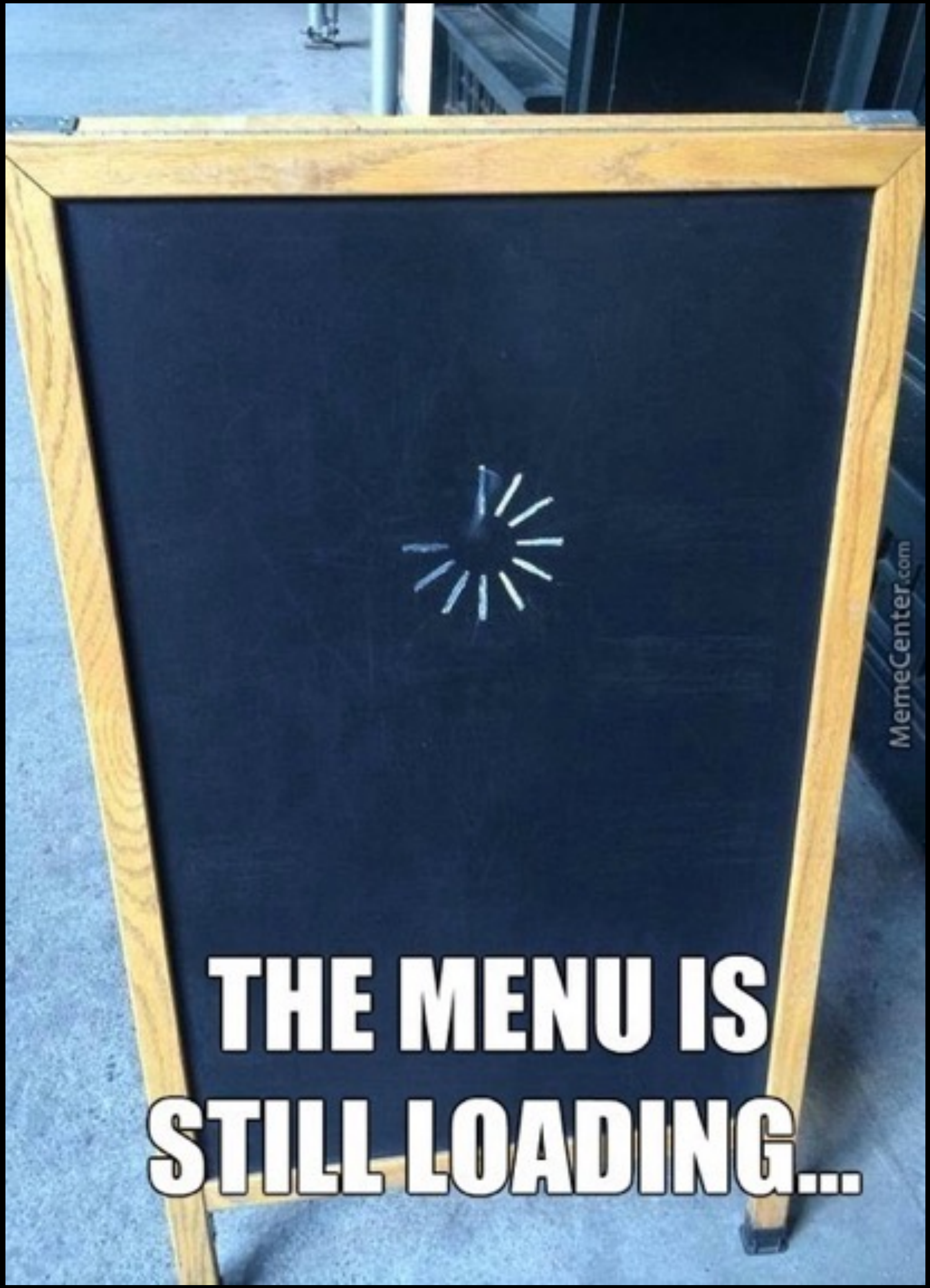


User Stories

Mobile Clients

- Unreliable Networks
- Less Secure Devices & Networks
- Limited Device & Network Resources
- Limited Control Over Adoption Of Mobile App Upgrades
- Leverage Mobile Client Technologies

Unreliable Networks

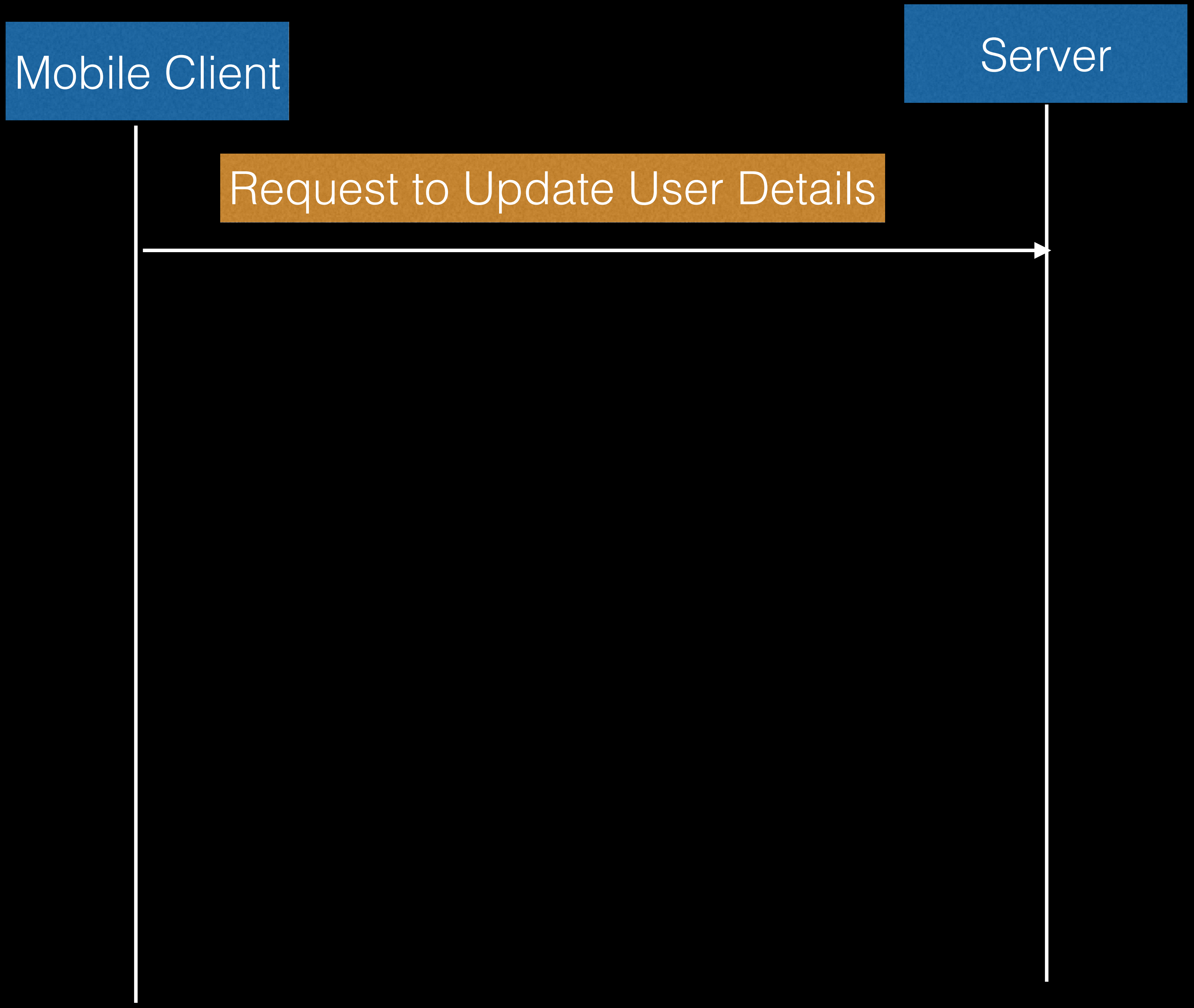


Network Unreliability

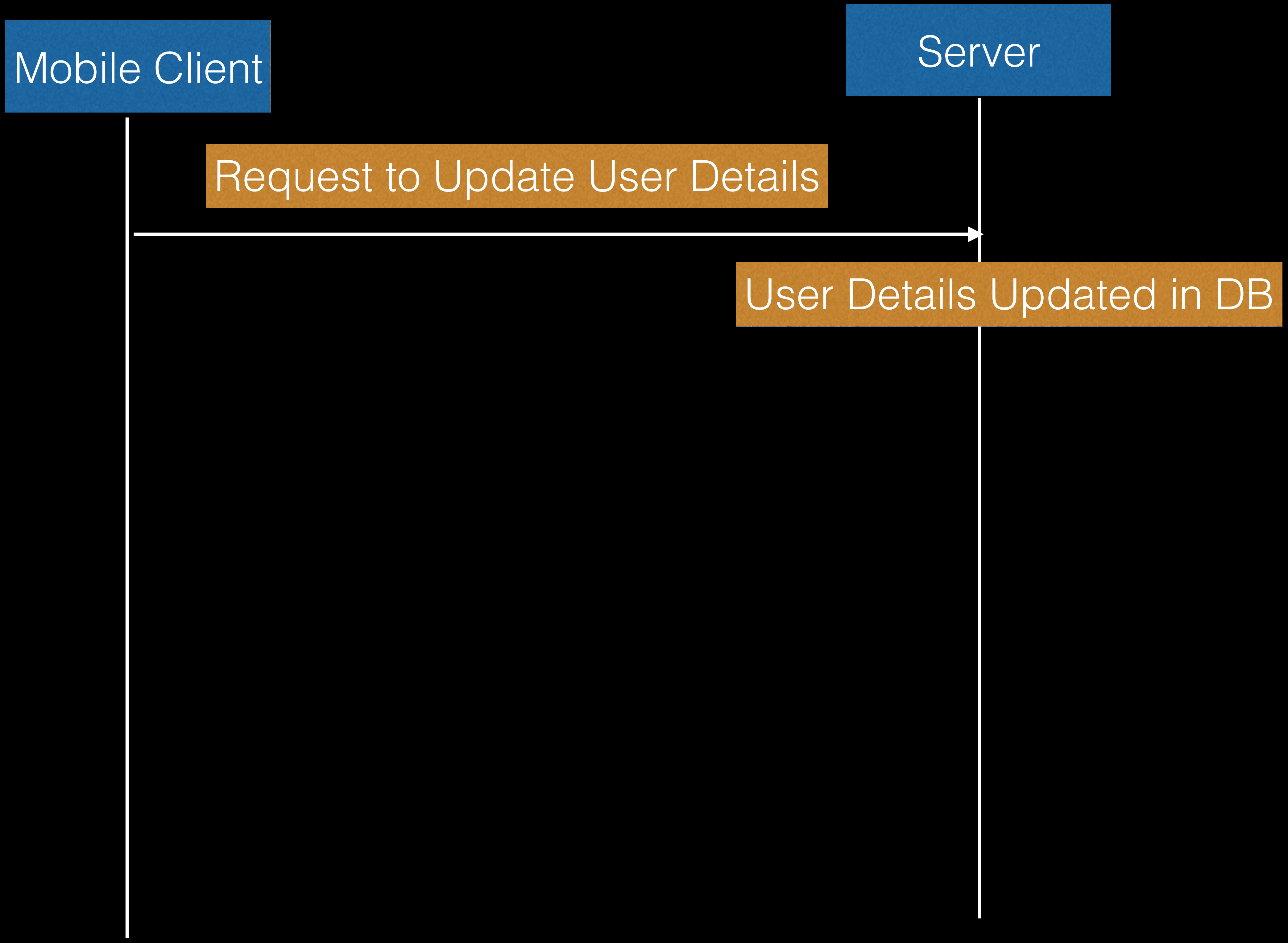
Mobile Client

Server

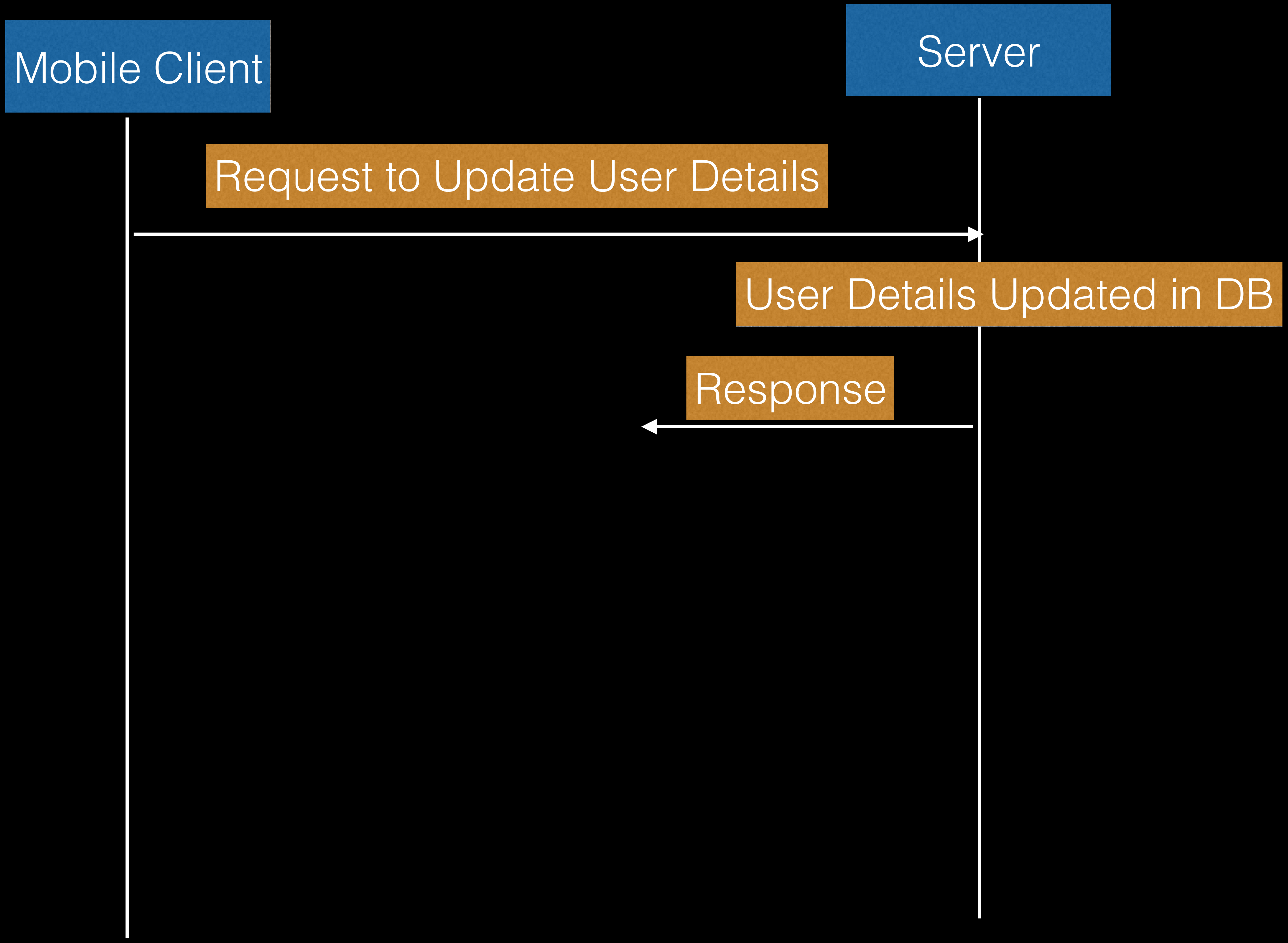
Network Unreliability


Mobile Client
Server
Request to Update User Details

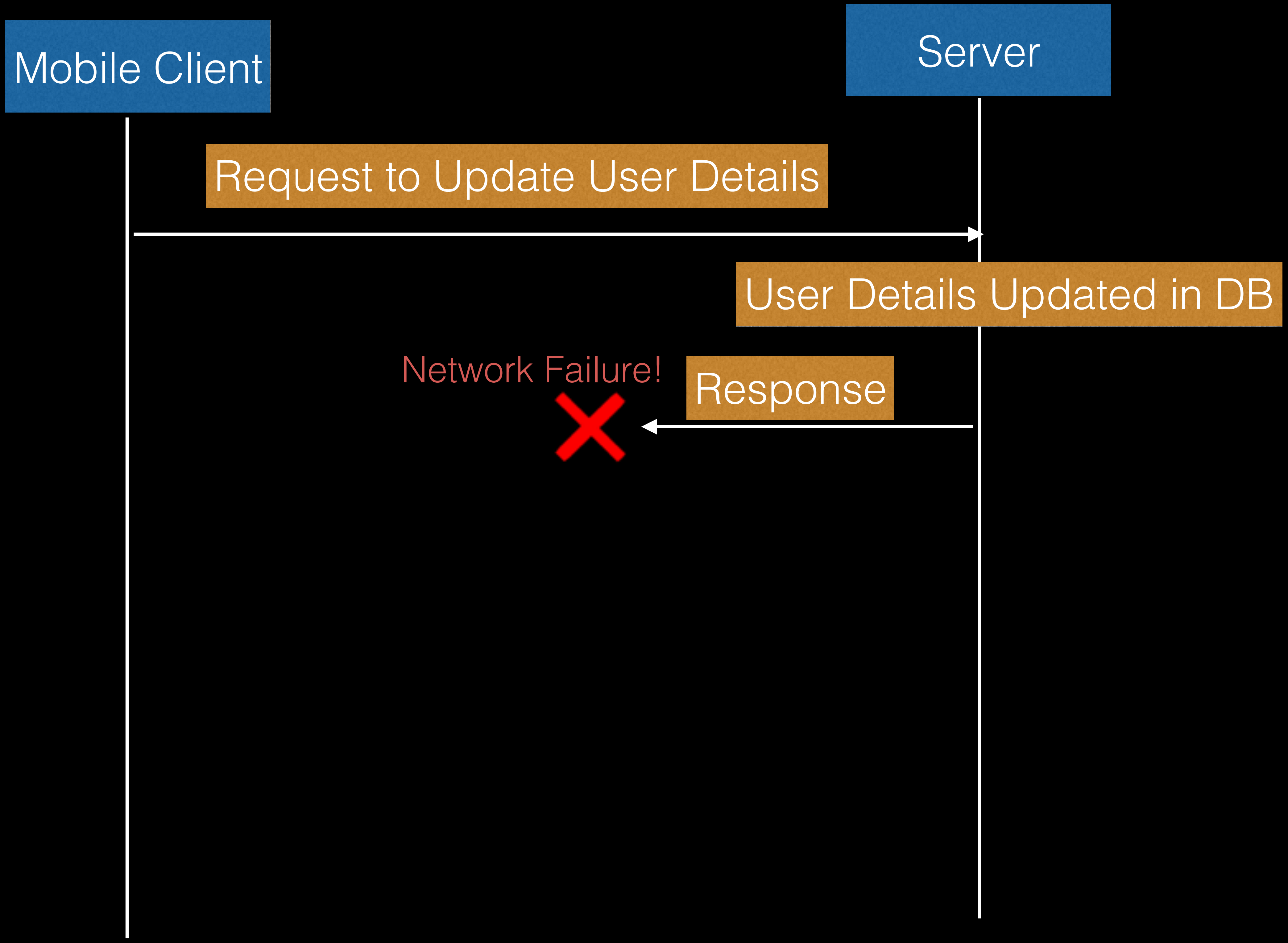
Network Unreliability



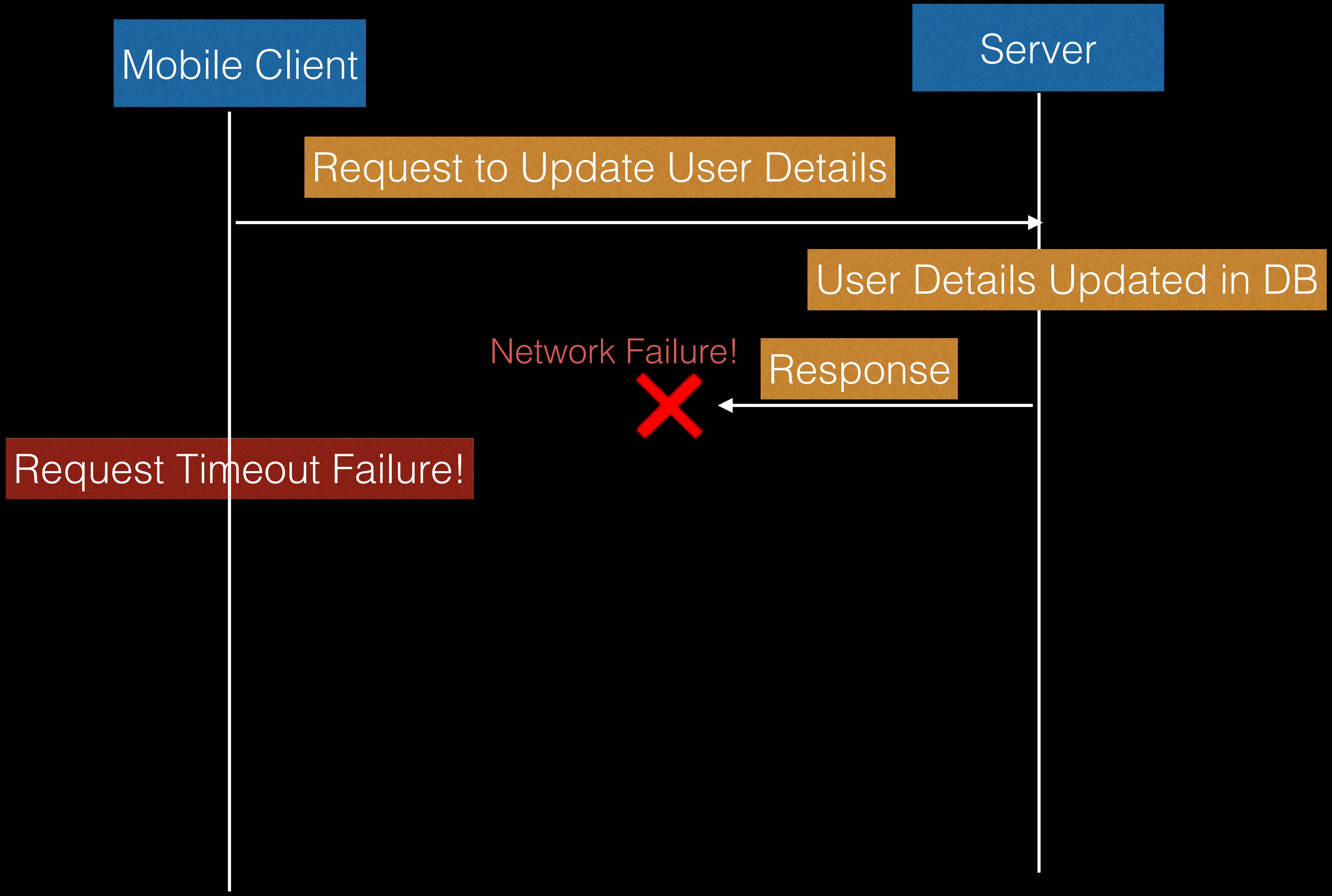
Network Unreliability



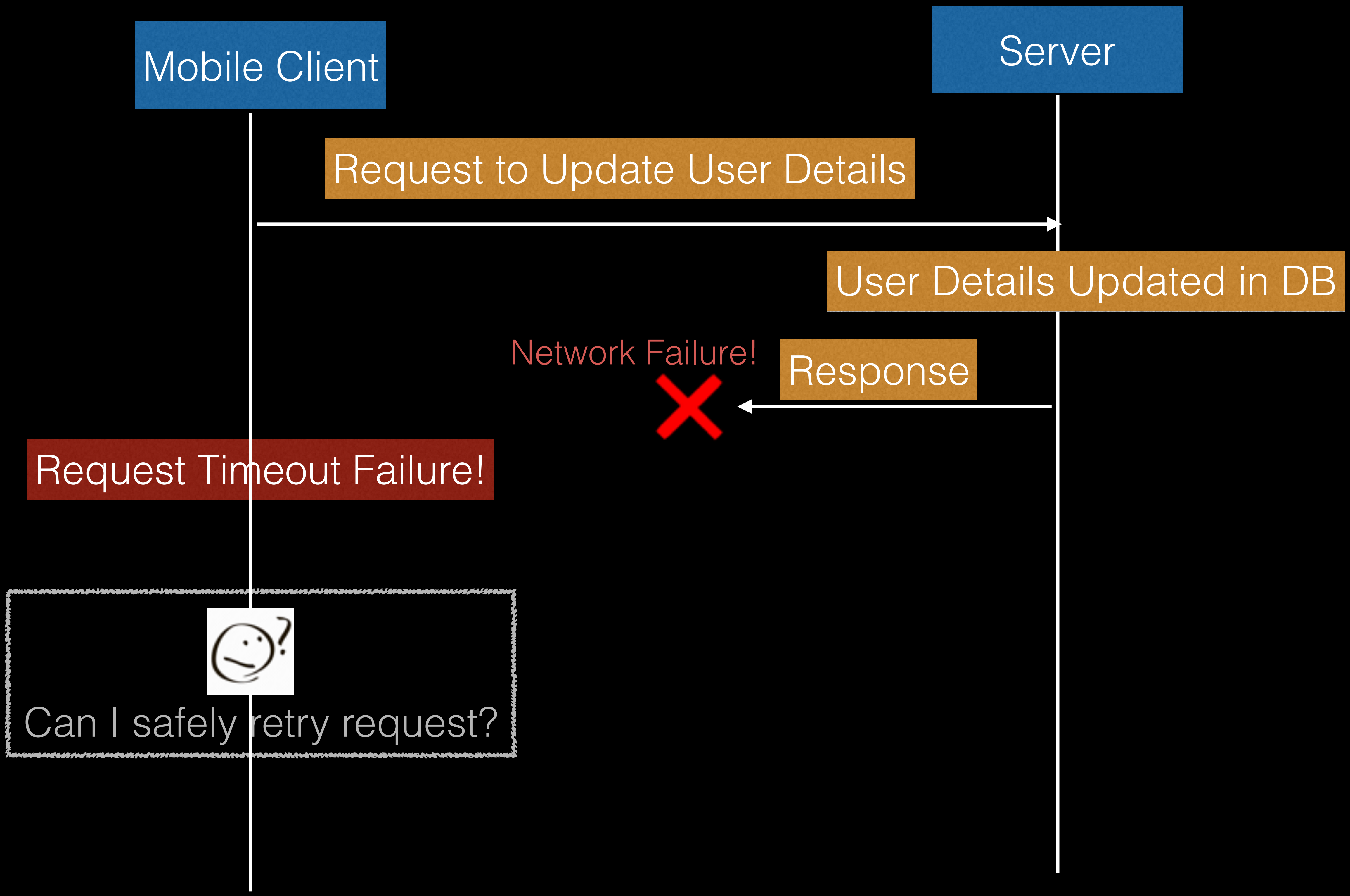
Network Unreliability



Network Unreliability



Network Unreliability



Mobile Client

Server

Request to Update User Details

User Details Updated in DB

Network Failure!

Response

Request Timeout Failure!

Can I safely retry request?

Avoid Ambiguity

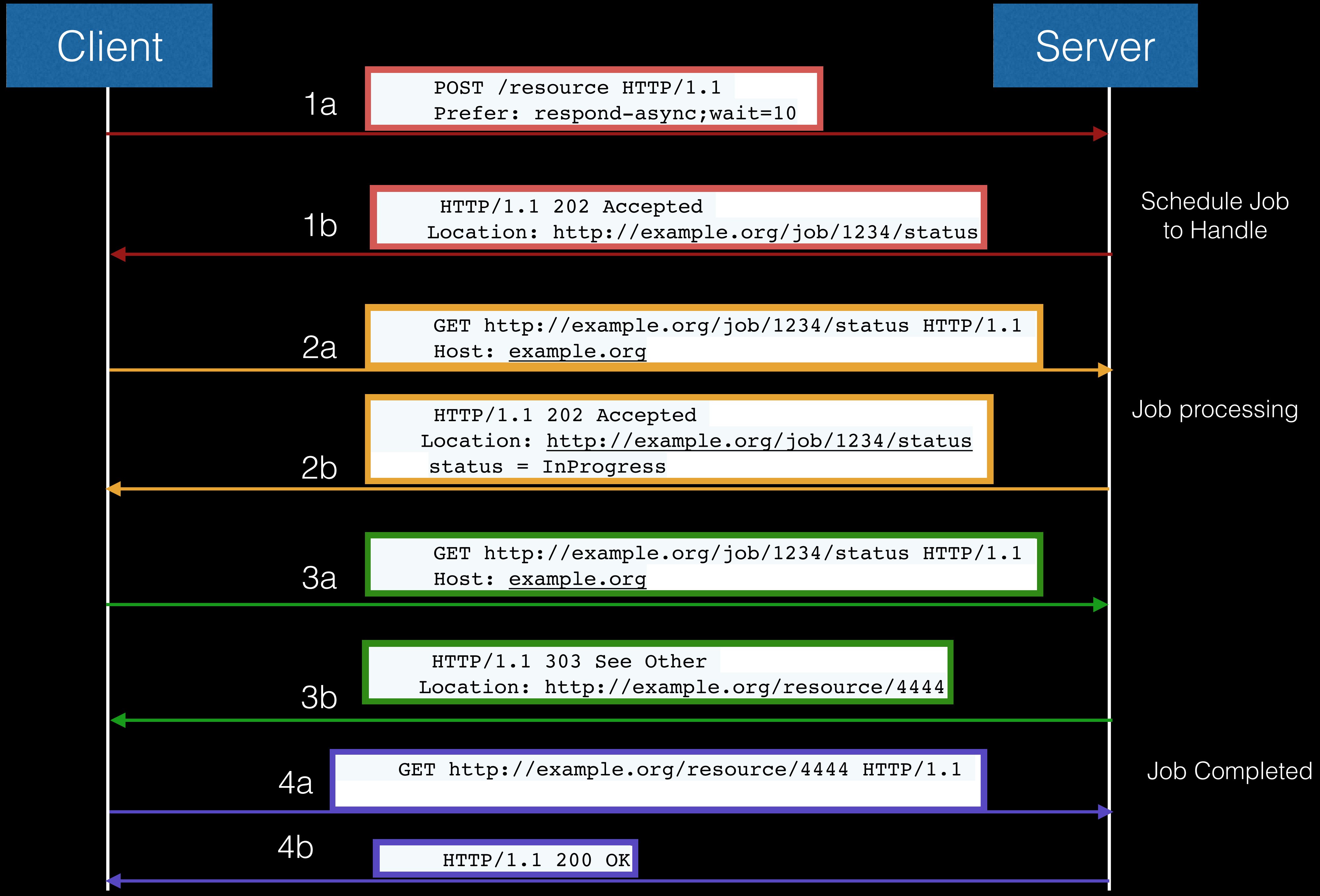
- Respect Idempotency Semantics of HTTP Methods

	Idempotent	Safe	Can Safely Retry?
POST	NO	NO	NO
PUT	YES	NO	YES
DELETE	YES	NO	YES
GET	YES	YES	YES
PATCH	NO	NO	NO

Avoid Blocking APIs

- Long Transactions Handled Asynchronously
- Async Handling Implementation Specific

Async Request Processing - The Standard Way



Client

Server

1a

```
POST /resource HTTP/1.1
Prefer: respond-async;wait=10
```

1b

```
HTTP/1.1 202 Accepted
Location: http://example.org/job/1234/status
```

Schedule Job to Handle

2a

```
GET http://example.org/job/1234/status HTTP/1.1
Host: example.org
```

Job processing

2b

```
HTTP/1.1 202 Accepted
Location: http://example.org/job/1234/status
status = InProgress
```

3a

```
GET http://example.org/job/1234/status HTTP/1.1
Host: example.org
```

3b

```
HTTP/1.1 303 See Other
Location: http://example.org/resource/4444
```

4a

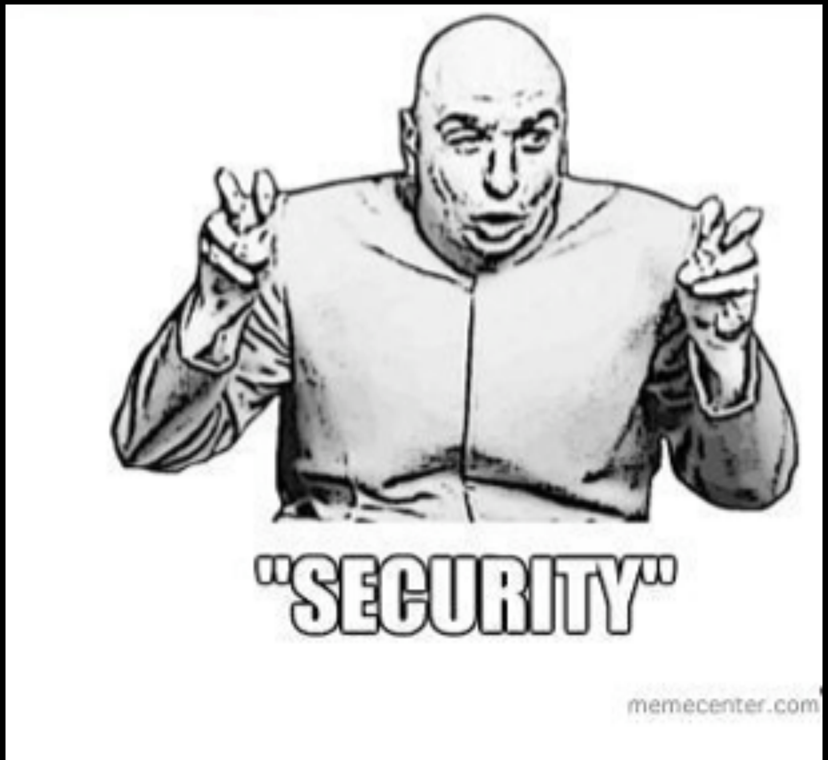
```
GET http://example.org/resource/4444 HTTP/1.1
```

Job Completed

4b

```
HTTP/1.1 200 OK
```

Less Secure Devices & Network



Network Security

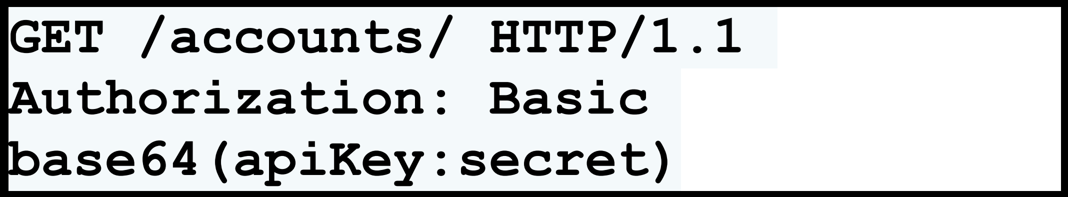
- Support HTTPS.
    - iOS9 - App Transport Security (ATS) Enabled by Default
        - Transport Layer Security (TLS) protocol version 1.2 (RFC 5246).
- Impl. Note on Client Side
    - Cert Pinning to Avoid MITM Attacks

Device Security

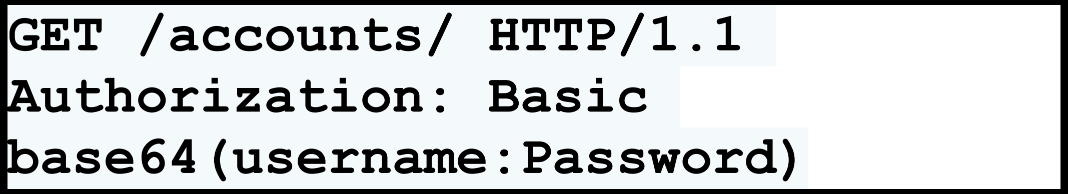
- Secure Storage on Mobile Devices Not Fully TrustWorthy… Getting better
  - iOS Devices - Keychain More Tamper Resistant
  - Android - KeyStore
- Impl. Note: Mobile Clients Adopt OWASP Recommendation

Basic Authentication … Meh

- Simple. Ubiquitous.
- API Key/Secret Needs to be securely stored on device
- Credentials Sent With Every Request - Increased Vulnerability
- Must use HTTPs



```
GET /accounts/ HTTP/1.1
Authorization: Basic
base64(apiKey:secret)
```



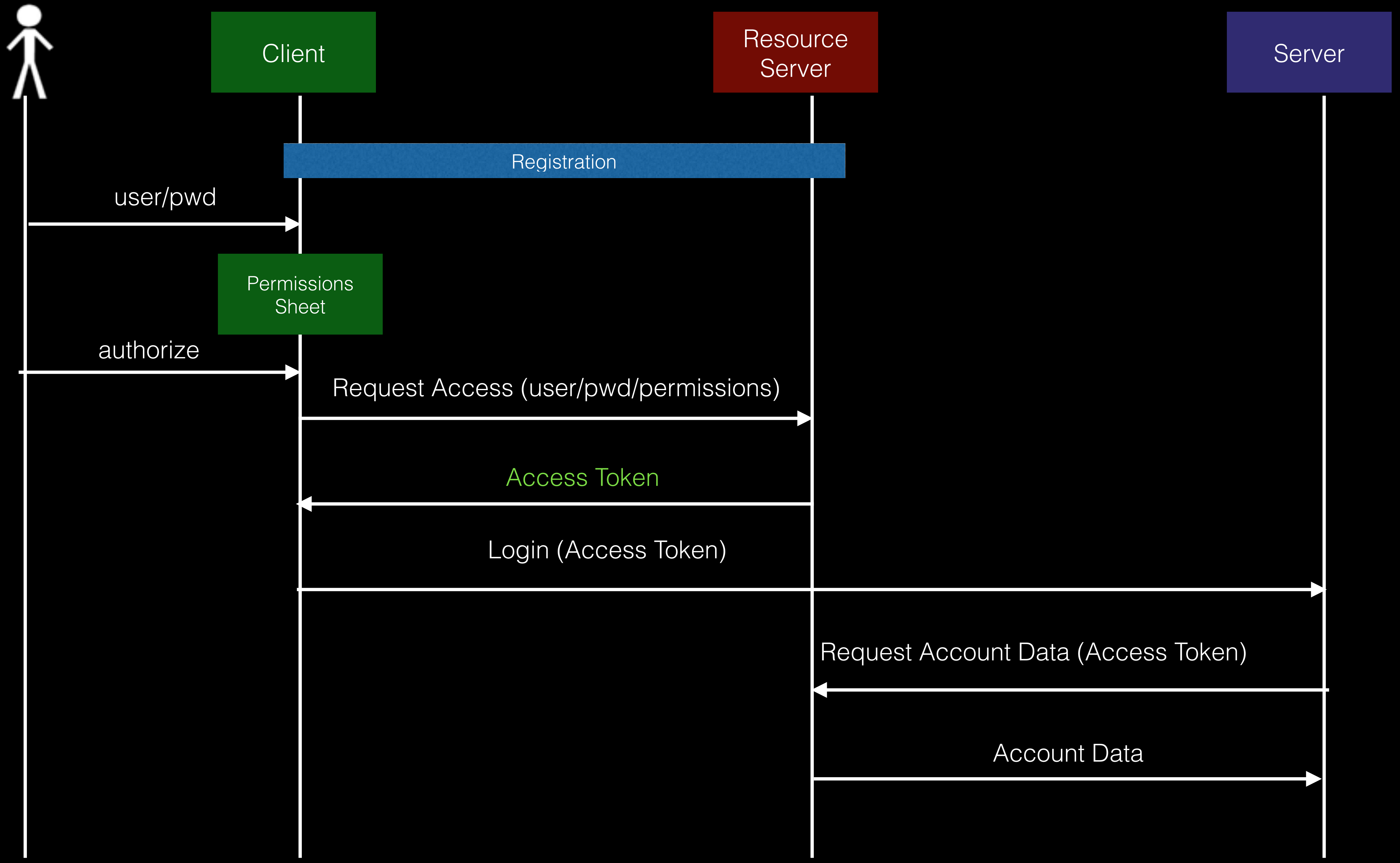
```
GET /accounts/ HTTP/1.1
Authorization: Basic
base64(username:Password)
```

Support Token Based Authentication

- Every Request is independently authenticated and authorized
- Limited Time Access (Time Bound)
- Limited Resource Access (Scoped)
    - Restrict resource visibility to mobile clients if needed
- Not Susceptible to Device Secure Storage Vulnerabilities
    - Stored Token is temporary

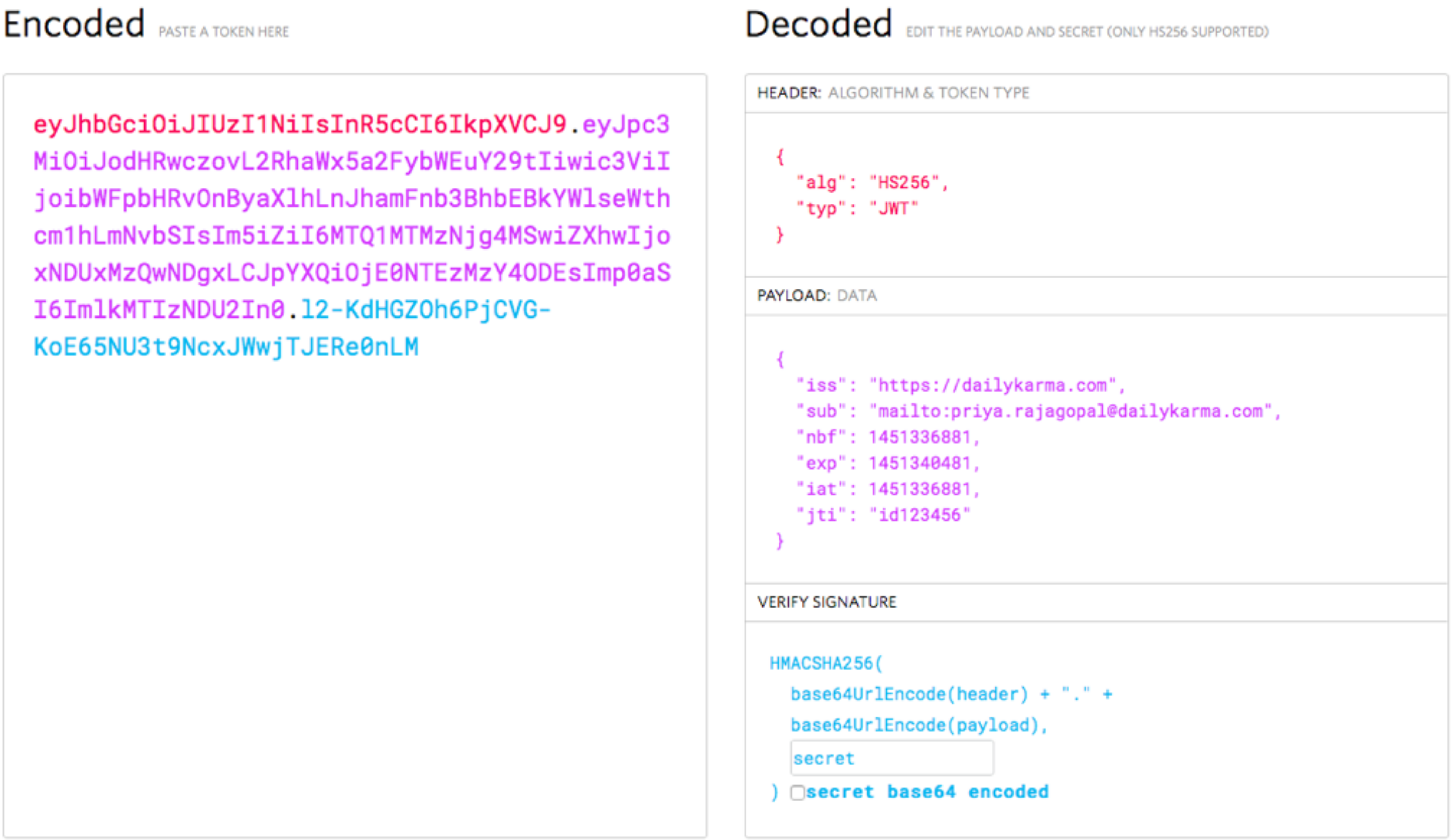
OAuth 2

Temporary Access to subset of resources



JSON Web Token (RFC 7519)

(jwt.io for libraries)



Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3
MiOiJodHRwczovL2RhaWx5a2FybWEuY29tIiwic3ViI
joibWFpbHRvOnByaXlhLnJhamFnb3BhbEBkYWlseWth
cm1hLmNvbSIsIm5iZiI6MTQ1MTMzNjg4MSwiZXhwIjo
xNDUxMzQwNDgxLCJpYXQiOjE0NTEzMzY4ODEsImp0aS
I6ImlkMTIzNDU2In0.l2-KdHGZOh6PjCVG-
KoE65NU3t9NcxJWwjTJERe0nLM
```

Decoded

EDIT THE PAYLOAD AND SECRET (ONLY HS256 SUPPORTED)

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "iss": "https://dailykarma.com",
  "sub": "mailto:priya.rajagopal@dailykarma.com",
  "nbf": 1451336881,
  "exp": 1451340481,
  "iat": 1451336881,
  "jti": "id123456"
}
```

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  secret
) □secret base64 encoded
```

Limited Device & Network Resources

Number of Network Requests Versus Response Size

- Need the right balance
- One Size Does Not Fit All
  - Flexibility in API
  - Let Clients Be In Control
- Risk of tighter coupling of clients w/ server data model

Support Ability To Control Message Size

- Pagination
- Filtering
- Sparse FieldSets
- HTTP Prefer Header
- gzip compression
  - Accept-Encoding: gzip

Pagination

```
GET https://myexample.com/api/resturants?page_number=0&page_size=1
Accept: application/vnd.myexample.restaurants+json;
```

```
HTTP/1.1 200 OK
Content-Type: application/vnd.myexample.restaurants+json; charset=utf-8
{
  "uri": “/restaurants",
  “total_pages”:10000,
  “page_number”:0,
  “page_size”:1
  "next": “/restaurants?page_number=1",
  "find": "/restaurants{?id}",
  "restaurants": [
    {
      "uri": "/restaurants/123",
      "name": “Bombay Express”,
      “cuisine”:”Indian”,
      “is_closed”:true
      “phone”:”555-333-2222”,
     “rating”:3.5,
     “website”:”http://be.com”,
     “location”:{
        "address": “Main Street",
        "city": “Ann Arbor",
        "country": “USA"
      “lat”:”43.28”
      “lon”:”83.74” }
    }
  ]
}
```

om

Filtering

```
GET https://myexample.com/api/resturants?page_number=0&page_size=1&query=
{“location.city”:”Canton”,”is_closed”:false}
content-type:application/json
Accept: application/vnd.myexample.restaurants+json;
```

```
HTTP/1.1 200 OK
Content-Type: application/vnd.myexample.restaurants+json; charset=utf-8
{
  "uri": “/restaurants",
  “total_pages”:78,
  “page_number”:0,
  “page_size”:1
  "next": “/restaurants?page_number=2",
  "find": "/restaurants{?id}",
  "restaurants": [
    {
      "uri": "/restaurants/560",
      "name": “Chinese Garden”
      “is_closed”:false
      “phone”:”555-222-2222”,
      “rating”:3.0,
      “website”:”http://cg.com”,
      “location”:{
        "address": “Canton Center",
        "city": “Canton",
        "country": “USA"
       “lat”:”43.28”
       “lon”:”83.74” }
    }
  ]
}
```

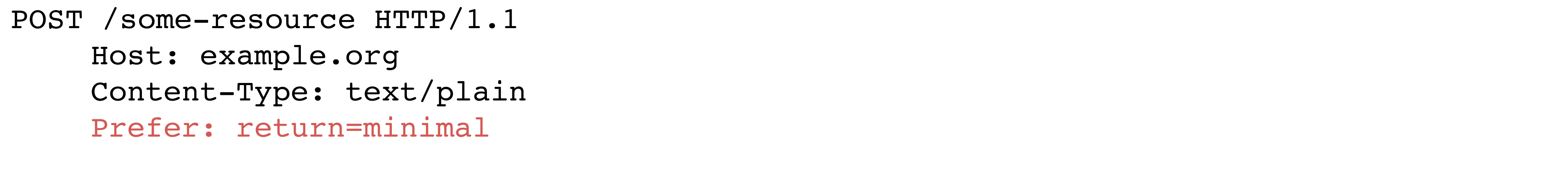
Sparse FieldSets

```
GET https://myexample.com/api/resturants?page_number=0&page_size=1& {“include”:
{“resource”:”reviews”,“fields”:”uri”},”fields”:[“name”,”rating”]}
content-type:application/json
Accept: application/vnd.myexample.restaurants+json;
```

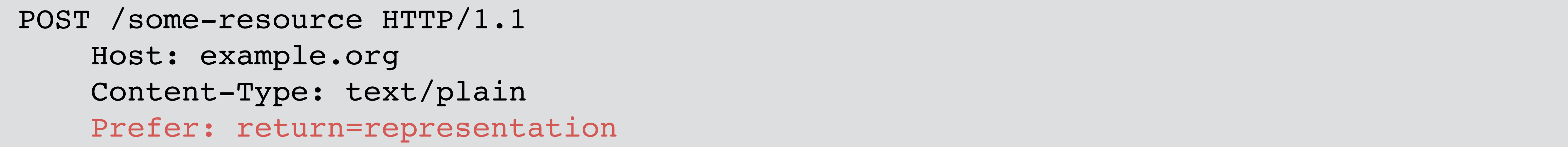
```
HTTP/1.1 200 OK
Content-Type: application/vnd.myexample.restaurants+json; charset=utf-8
{
  "uri": “/restaurants",
  “total_pages”:10000,
  “page_number”:0,
  “page_size”:1
  "next": “/restaurants?page_number=2",
  "find": "/restaurants{?id}",
  "restaurants": [
    {
      "uri": "/restaurants/123",
      "name": “Bombay Express”,
      “rating”:3.5,
      “reviews : [
           {
              “uri”:”/reviews/3322”
            },
            {
              “uri”:”/reviews/3322”
            },
            {
            “uri”:”/reviews/2211”
            }
            ],
      }
    }
  ]
}
```

HTTP Prefer Header

- IETF Standard . https://tools.ietf.org/html/rfc7240
- End-To-End
- Standard Way For Client to State Response Preferences
    - Interpretation is Application Specific



```
POST /some-resource HTTP/1.1
     Host: example.org
     Content-Type: text/plain
     Prefer: return=minimal
```



```
POST /some-resource HTTP/1.1
     Host: example.org
     Content-Type: text/plain
     Prefer: return=representation
```

Support Partial Updates

Full Updates	Partial Updates
PUT /users/101 HTTP/1.1 if-none-match:<etag> { “firstname”:”priya”, “lastname”:”rajagopal” }	PATCH /users/101 HTTP/1.1 if-match:<etag> [{“op”:”replace”,”path”:“lastname”,”value”:”rajagopal”}]
	POST w/ clear agreement on Missing Params POST /users/101 HTTP/1.1 if-match:<etag> { “lastname”:”rajagopal” }
	Avoid Tunneling Other Requests through POST
	Do Not use PUT

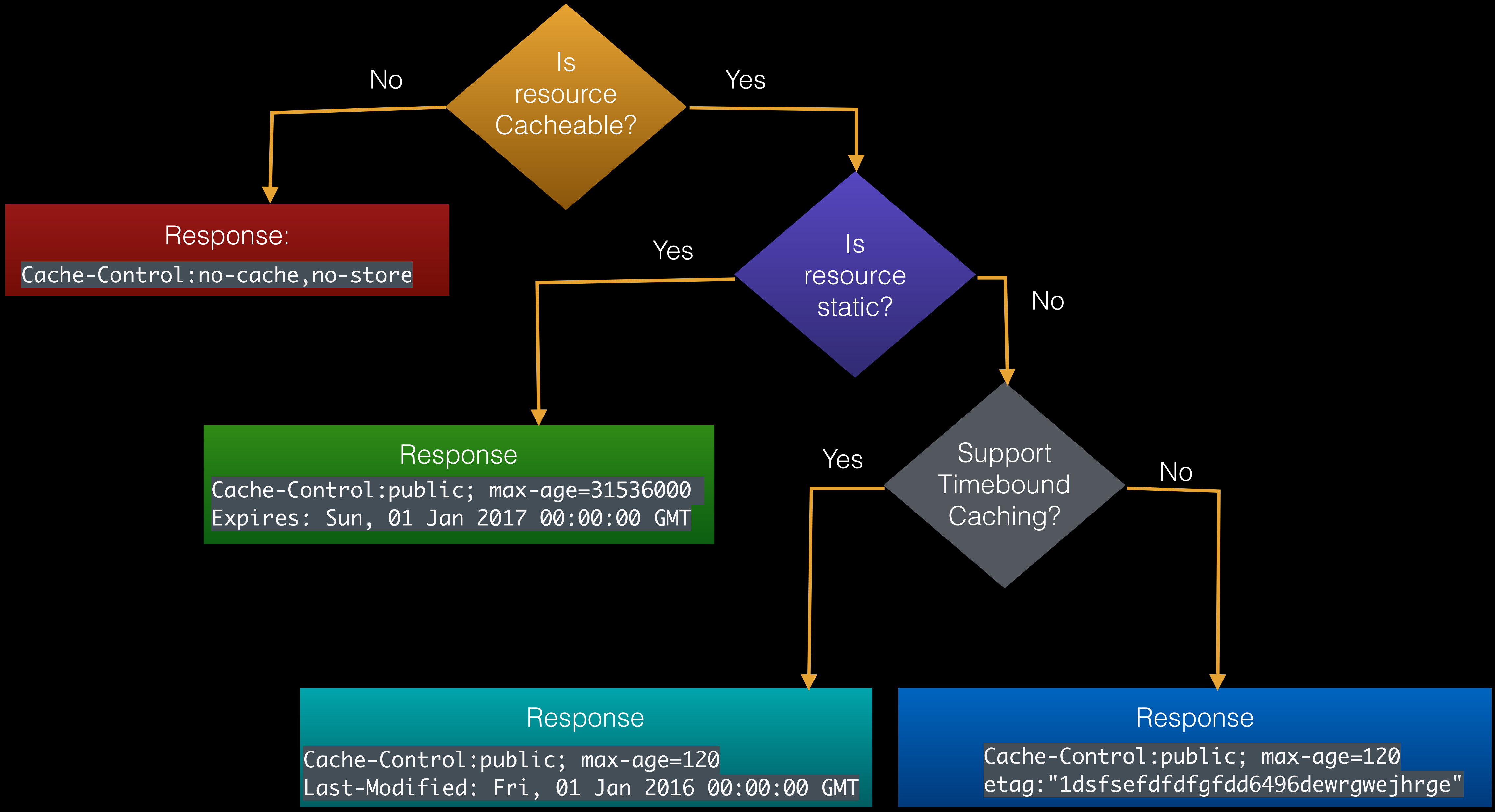
Sorting

- Server Side Or Client Side ?
  - Paging Considerations
  - Performance Implications
  - Need to Implement Logic in multiple clients
- Specifying Sort Criteria
  - Same Options as Filter

Support Resource Caching

- Don’t underestimate its importance!
- Mobile Clients can leverage local storage
- Improved User Perceived Performance
- Reduced Network Bandwidth Usage
- Leverage HTTP 1.1 Protocol caching
  - Avoid Custom Caching mechanisms
- Limited Memory => Smaller Caches on Devices
- Server Side Scalability Benefits

For Every Resource …



No

Is resource Cacheable?

Yes

Response:

Cache-Control:no-cache,no-store

Yes

Is resource static?

No

Response

Cache-Control:public; max-age=31536000
Expires: Sun, 01 Jan 2017 00:00:00 GMT

Yes

Support Timebound Caching?

No

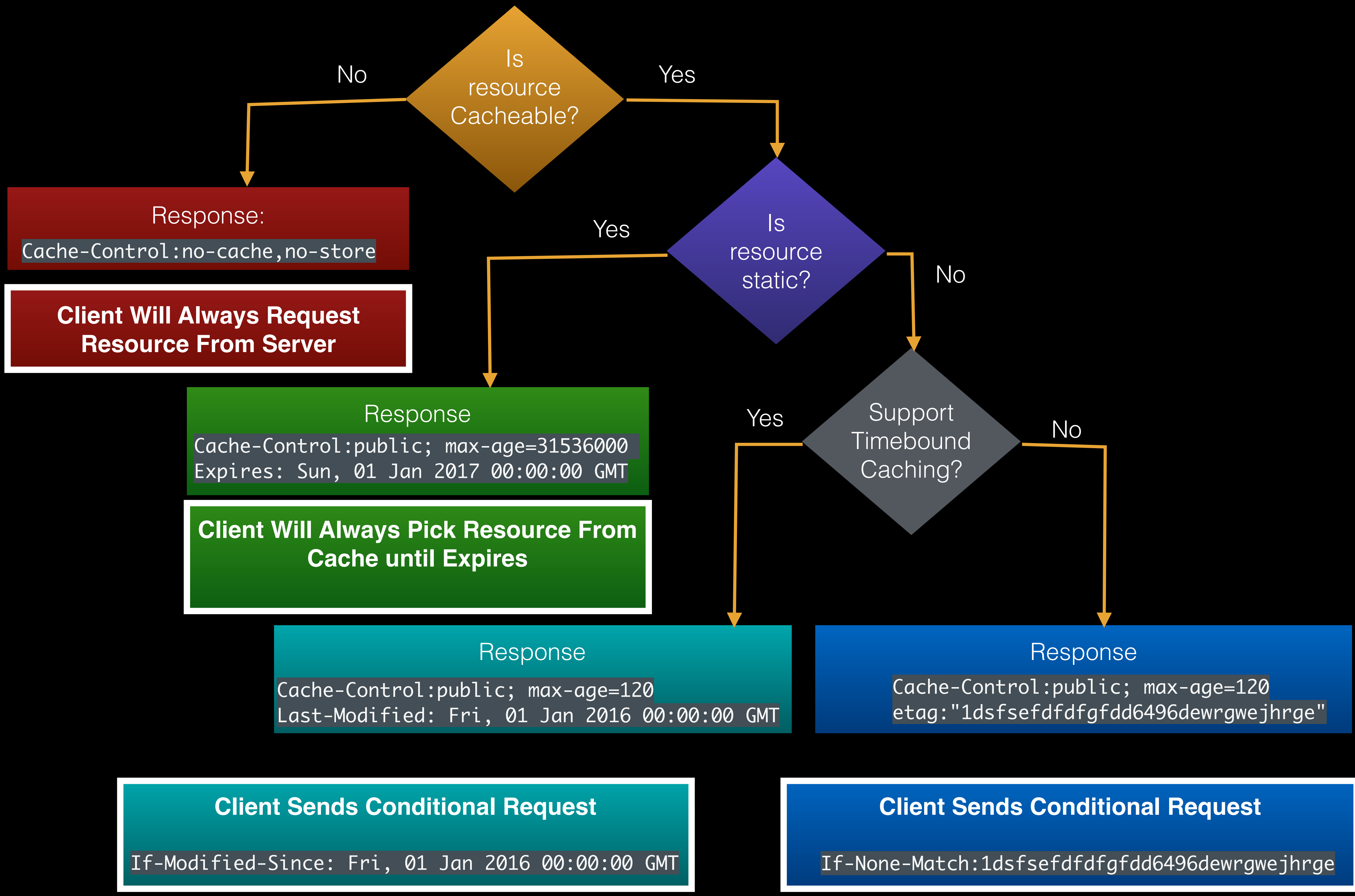
Response

Cache-Control:public; max-age=120
Last-Modified: Fri, 01 Jan 2016 00:00:00 GMT

Response

Cache-Control:public; max-age=120
etag:"1dsfsefdfdfgfdd6496dewrgwejhrge"

For Every Resource …



Is resource Cacheable?

No

Yes

Response:

`Cache-Control:no-cache,no-store`

Client Will Always Request Resource From Server

Is resource static?

Yes

No

Response

```
Cache-Control:public; max-age=31536000
Expires: Sun, 01 Jan 2017 00:00:00 GMT
```

Client Will Always Pick Resource From Cache until Expires

Support Timebound Caching?

Yes

No

Response

```
Cache-Control:public; max-age=120
Last-Modified: Fri, 01 Jan 2016 00:00:00 GMT
```

Response

```
Cache-Control:public; max-age=120
etag:"1dsfsefdfdfgfdd6496dewrgwejhrge"
```

Client Sends Conditional Request

`If-Modified-Since: Fri, 01 Jan 2016 00:00:00 GMT`

Client Sends Conditional Request

`If-None-Match:1dsfsefdfdfgfdd6496dewrgwejhrge`

Be Smart About Serving Images

- Different Image Sizes Depending on client type
- Use Separate URL For Image Resource Over Inline Images
    - Caching
    - Response Size
    - Multipart message handling
- Enable Appropriate Cache-control Header on images

Limited Control Over Mobile App Upgrade Adoption

Mobile App Updates

- Users “encouraged” to upgrade but cannot enforce
- Mobile App Upgrade Cycle Longer
  - App Store Reviews
  - Clients Less Vulnerable to URI changes
- API Versioning

Make Mobile Clients Less vulnerable to Resource Changes

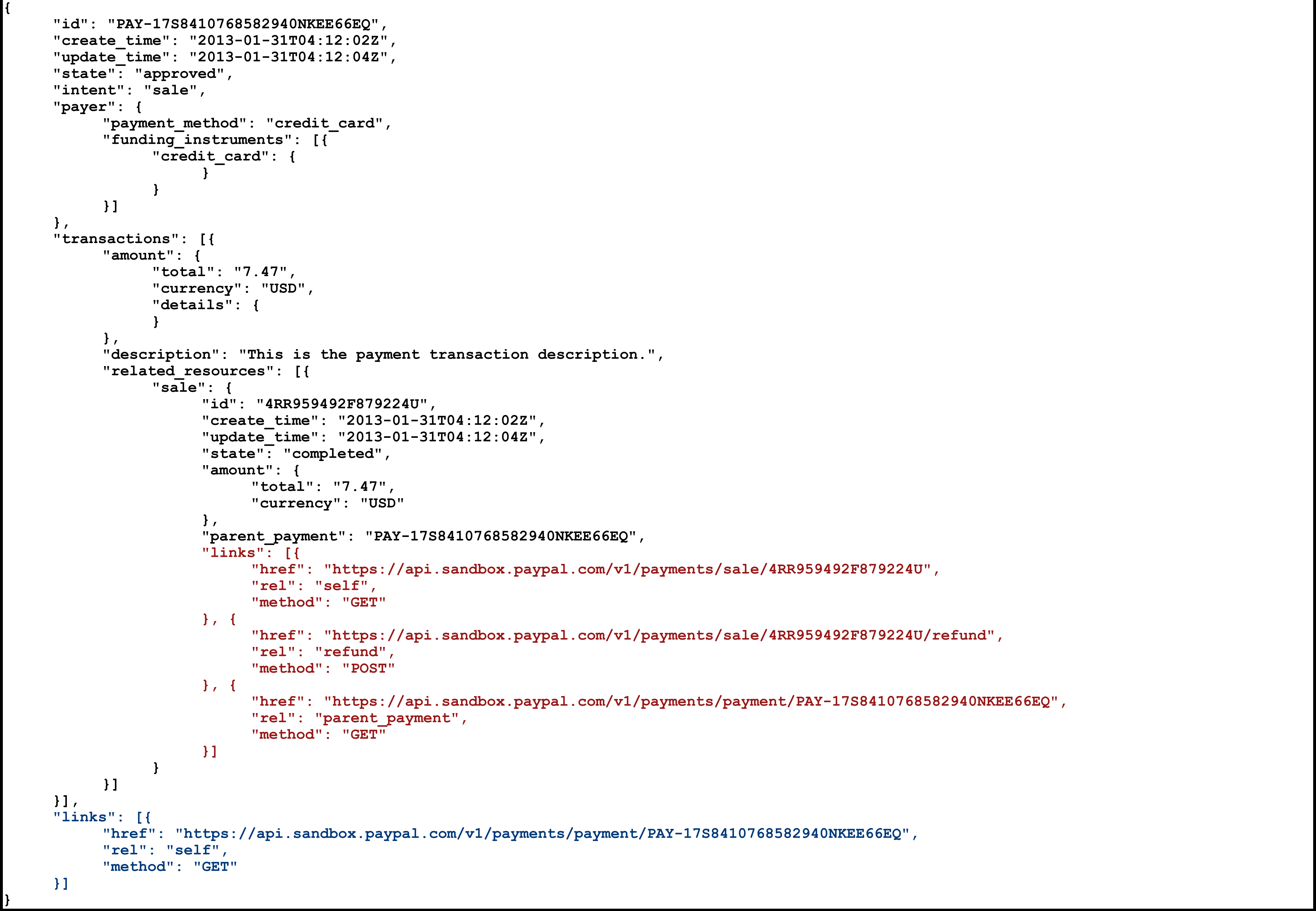
- Client- Server Decoupling Even More Important
- Hypertext As The Engine Of Application State (HATEOAS)
    - REST Constraint

HATEOAS

- Application State Driven Through HyperLinks
- Runtime Discoverability Of …
  - Resource URIs & Relationships
- Resources “Surfed” via Hyperlinks
- Clients More Decoupled From Server
- “Generic” Clients

PayPal Example

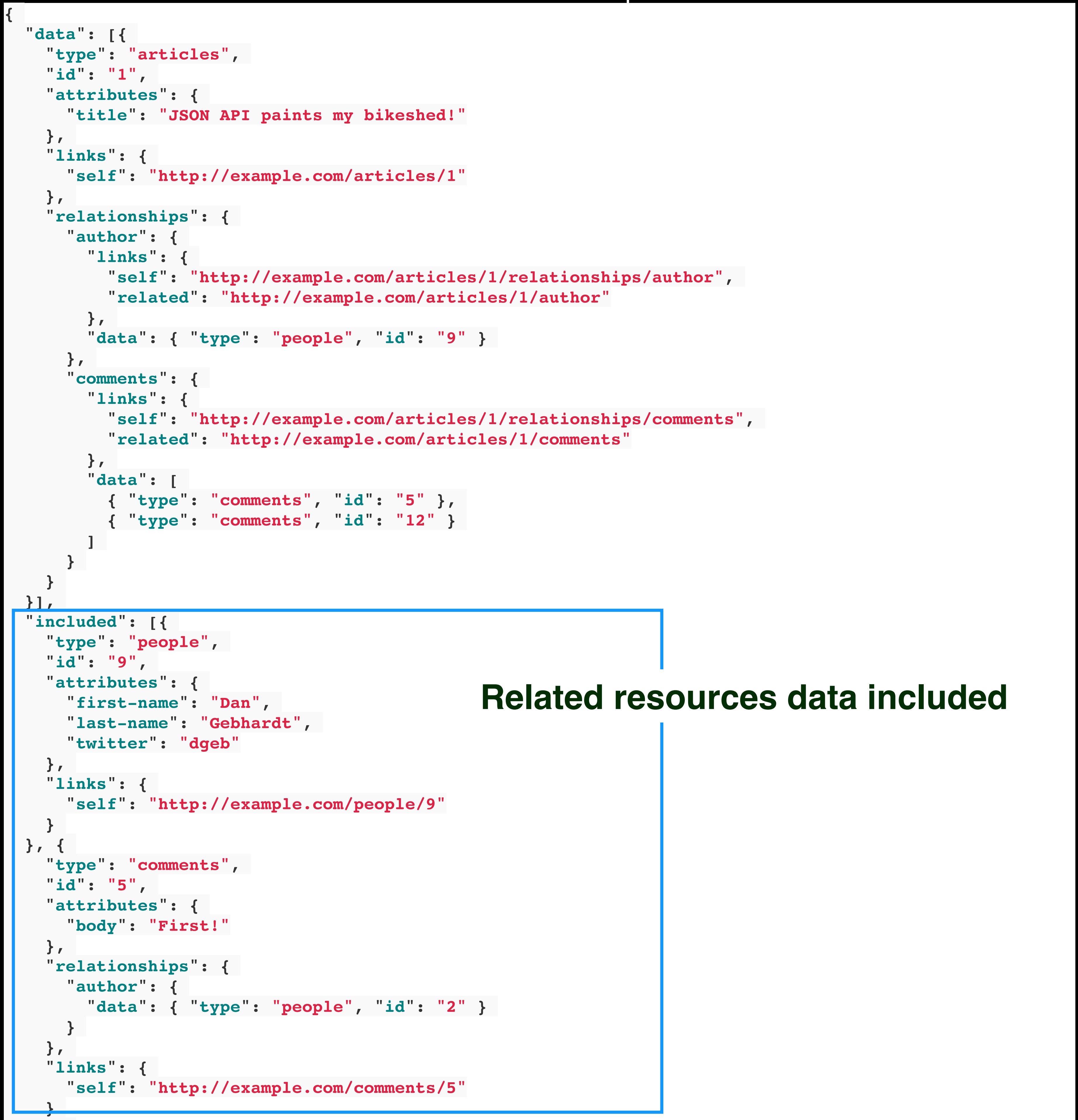
`POST /v1/payments/payment`



```
{
      "id": "PAY-17S8410768582940NKEE66EQ",
      "create_time": "2013-01-31T04:12:02Z",
      "update_time": "2013-01-31T04:12:04Z",
      "state": "approved",
      "intent": "sale",
      "payer": {
            "payment_method": "credit_card",
            "funding_instruments": [{
                  "credit_card": {
                        }
                  }
            }]
      },
      "transactions": [{
            "amount": {
                  "total": "7.47",
                  "currency": "USD",
                  "details": {
                  }
            },
            "description": "This is the payment transaction description.",
            "related_resources": [{
                  "sale": {
                        "id": "4RR959492F879224U",
                        "create_time": "2013-01-31T04:12:02Z",
                        "update_time": "2013-01-31T04:12:04Z",
                        "state": "completed",
                        "amount": {
                              "total": "7.47",
                              "currency": "USD"
                        },
                        "parent_payment": "PAY-17S8410768582940NKEE66EQ",
                        "links": [{
                              "href": "https://api.sandbox.paypal.com/v1/payments/sale/4RR959492F879224U",
                              "rel": "self",
                              "method": "GET"
                        }, {
                              "href": "https://api.sandbox.paypal.com/v1/payments/sale/4RR959492F879224U/refund",
                              "rel": "refund",
                              "method": "POST"
                        }, {
                              "href": "https://api.sandbox.paypal.com/v1/payments/payment/PAY-17S8410768582940NKEE66EQ",
                              "rel": "parent_payment",
                              "method": "GET"
                        }]
                  }
            }]
      }],
      "links": [{
            "href": "https://api.sandbox.paypal.com/v1/payments/payment/PAY-17S8410768582940NKEE66EQ",
            "rel": "self",
            "method": "GET"
      }]
}
```

https://developer.paypal.com/docs/api/#hateoas-links

JSON API Example



```
{
  "data": [{
    "type": "articles",
    "id": "1",
    "attributes": {
      "title": "JSON API paints my bikeshed!"
    },
    "links": {
      "self": "http://example.com/articles/1"
    },
    "relationships": {
      "author": {
        "links": {
          "self": "http://example.com/articles/1/relationships/author",
          "related": "http://example.com/articles/1/author"
        },
        "data": { "type": "people", "id": "9" }
      },
      "comments": {
        "links": {
          "self": "http://example.com/articles/1/relationships/comments",
          "related": "http://example.com/articles/1/comments"
        },
        "data": [
          { "type": "comments", "id": "5" },
          { "type": "comments", "id": "12" }
        ]
      }
    }
  }],
  "included": [{
    "type": "people",
    "id": "9",
    "attributes": {
      "first-name": "Dan",
      "last-name": "Gebhardt",
      "twitter": "dgeb"
    },
    "links": {
      "self": "http://example.com/people/9"
    }
  }, {
    "type": "comments",
    "id": "5",
    "attributes": {
      "body": "First!"
    },
    "relationships": {
      "author": {
        "data": { "type": "people", "id": "2" }
      }
    },
    "links": {
      "self": "http://example.com/comments/5"
    }
```

Related resources data included

Support Resource Representations Suited For Mobile Native Clients

- Think Non- Browser Clients
- JSON Popular Media Format
- JSON's Lack of Native Hypermedia Support
- Response To Have Right Balance of Hyperlinks & Inline Content
    - Reduce chatter

JSON and Hypermedia Support

Lots of Ongoing Work in Standards

Media Type	Reference	MIME
JSON-LD - JSON For Linked Document	http://www.w3.org/TR/json-ld-api/	application/ld+json
JSON-HAL- JSON Hypertext Application Language	https://tools.ietf.org/html/draft-kelly-json-hal-06	application/hal+json
Collection+JSON	http://amundsen.com/media-types/collection/format/	application/vnd.collection+json
JSONAPI	http://jsonapi.org	application/vnd.api+json
Vendor Specific	http://thenextbigthinginhypermedia.com	application/vnd.priyahyperapi+json

JSON and Hypermedia Support

Client Support

Media Type	Client Side Support Libraries
JSON-LD - JSON For Linked Document	• Java- https://github.com/jsonld-java/ • iOS: Nothing actively maintained
JSON-HAL- JSON Hypertext Application Language	• https://github.com/mikekelly/hal_specification/wiki/Libraries
Collection+JSON	• Java: https://github.com/hamnis/json-collection • iOS: Nothing actively maintained for iOS
JSONAPI	• http://jsonapi.org/implementations/

API Versioning

- "*You don't even need versioning if you HATEOAS*" …. Well.. Maybe
- Include version in the URI

  https://mywebservice/api/V1/resource1

- Media Type Specifies Version

  Accept: application/vnd.company.mywebservice.resource1+json;version=2.0

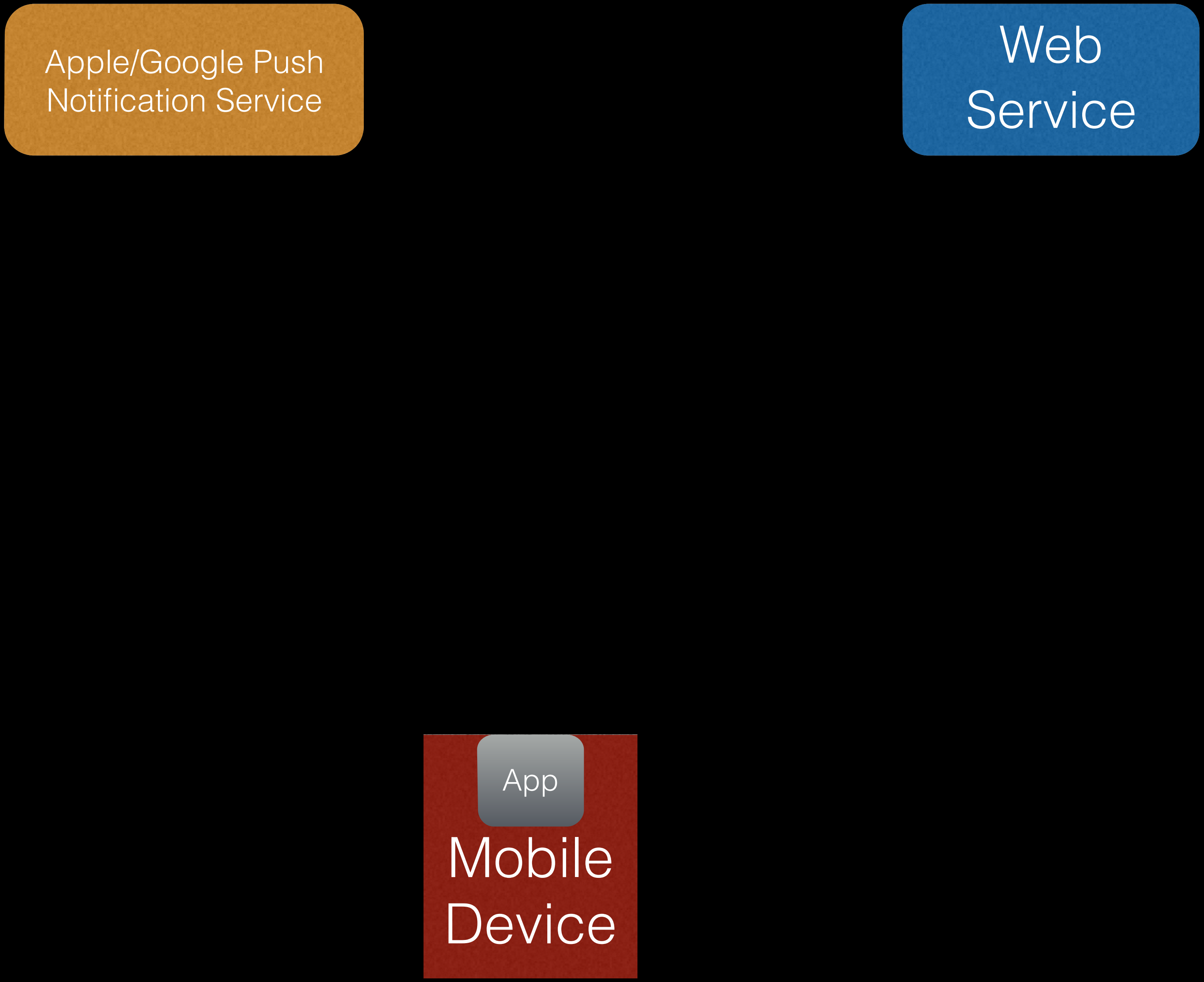
  Cntent-Type: application/vnd.company.mywebservice.resource1+json;version=2.0

- While Migrating To a New Version
  - Maintain penultimate version for period of time
  - Sufficient transition time for clients to migrate

Leverage Native App Technologies

Push Notifications

- Avoid Polling
- Apple Push Notification Service
- Google Cloud Messaging



Apple/Google Push Notification Service

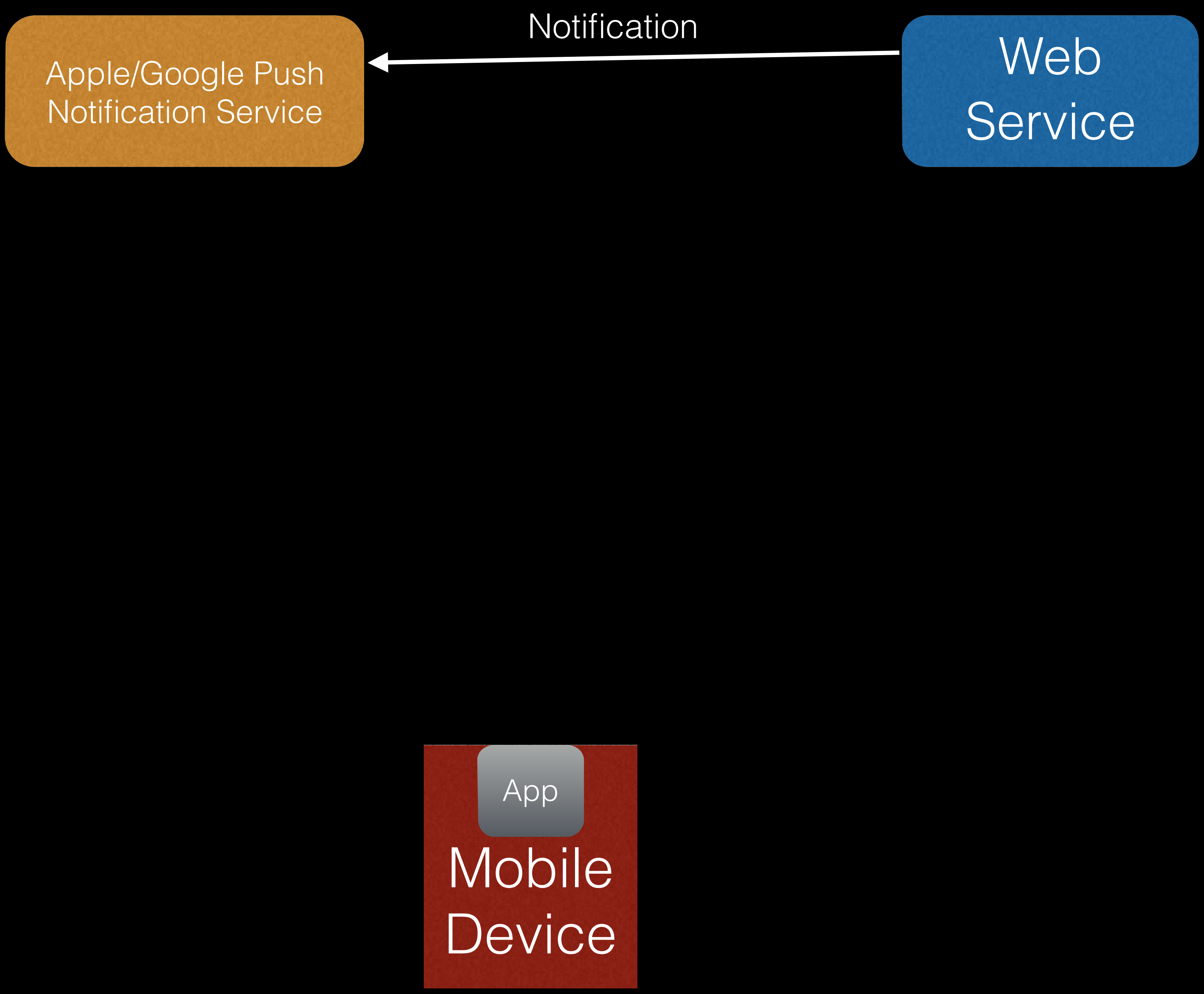
Web Service

App

Mobile Device

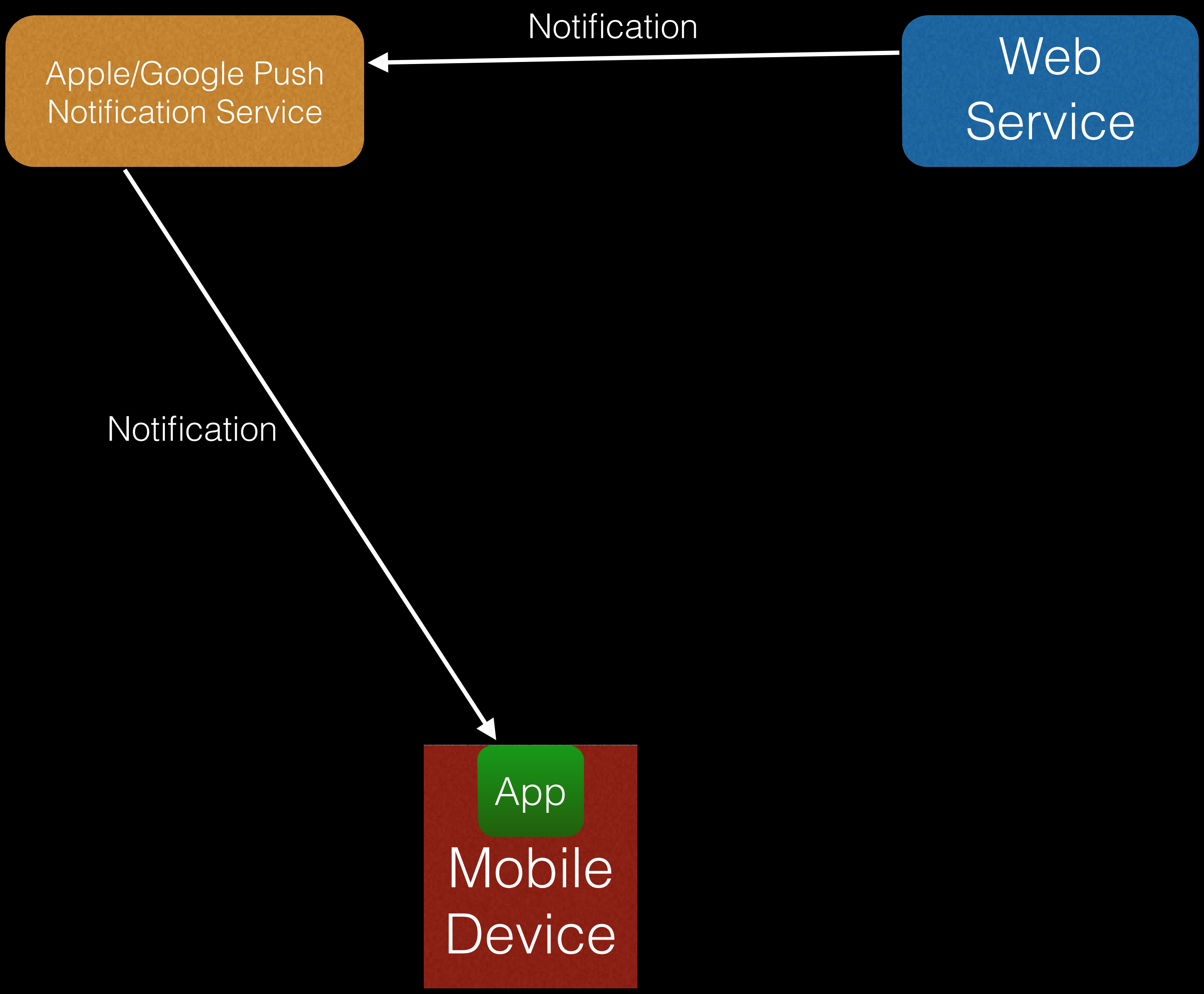
Push Notifications

- Avoid Polling
- Apple Push Notification Service
- Google Cloud Messaging



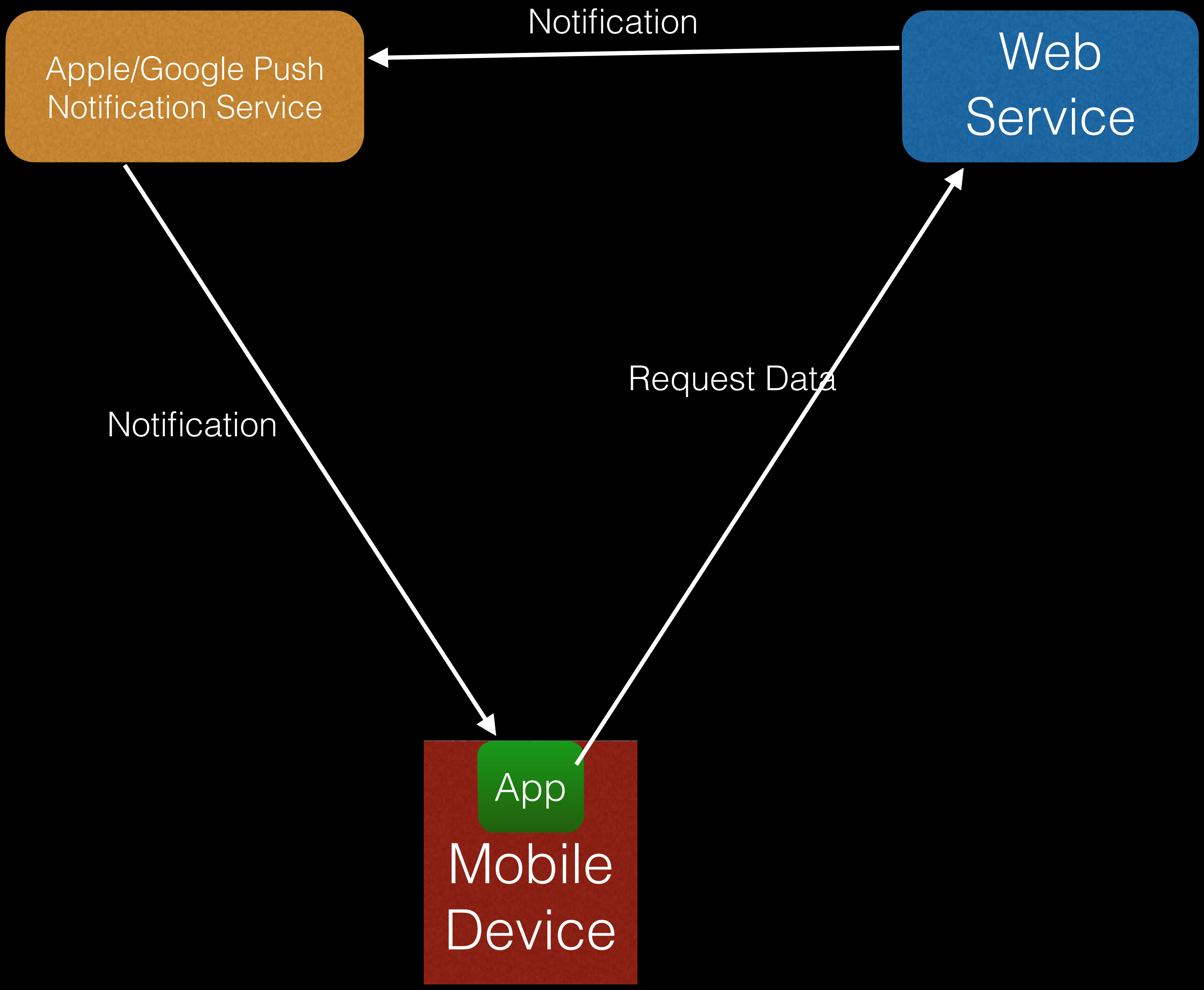
Push Notifications

- Avoid Polling
- Apple Push Notification Service
- Google Cloud Messaging



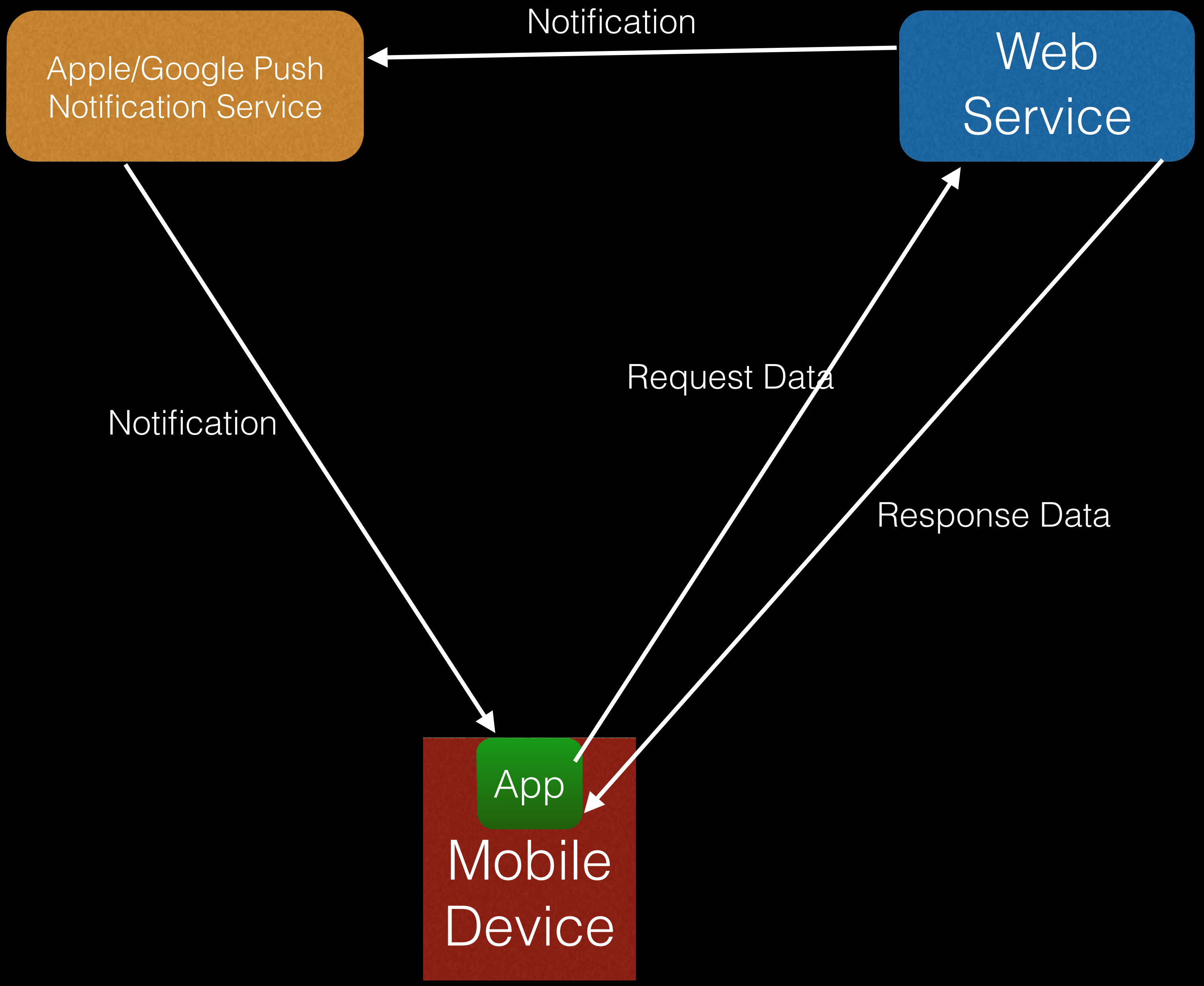
Push Notifications

- Avoid Polling
- Apple Push Notification Service
- Google Cloud Messaging



Push Notifications

- Avoid Polling
- Apple Push Notification Service
- Google Cloud Messaging



So To Recap…

1. Respect Idempotency Semantics of HTTP Methods
2. Avoid Blocking APIs
3. Do HTTPs
4. Support Token Based Authentication
5. Support Ability to Control Response Size
6. Support Partial Updates
7. Support Resource Caching
8. Be Smart About Serving Images
9. HATEOAS for Decoupling Clients
10. Resource Versioning Done Right

If You haven’t had enough REST…

- Testing RESTful Web Services, Mark Winteringham,(Indigo Bay,January 6, 2016 1:00 PM)
- Get Some REST- On Practical RESTful API Design, Priya Rajagopal (Orange,January 7, 2016 9:15 AM)
- Consuming REST APIs, for all interpretations of REST, Darrel Miller, (Indigo Bay, January 7, 2016 10:30 AM)
- Making life a bit easier for mobile app developers through better REST API Design, Priya Rajagopal (Mangrove, January 7, 2016 11:45 AM)
- Hypermedia APIs: The rest of REST, Chris Marinos (Salon A, January 7, 2016 1:00 PM)
- Ember Data. The key to good relationships is communication (to your REST server), Brian Gantzler (Portia, Wisteria, January 7, 2016 3:30 PM)

Thank You !

Priya Rajagopal
Twitter: @rajagp