

Real World REST with C# and ASP.NET

@dylanbeattie



Dylan Beattie

 @dylanbeattie

- Building websites since 1992
- Systems architect at www.spotlight.com
- Web site: www.dylanbeattie.net

Slides & code:

github.com/dylanbeattie/NDC2017/

Workshop Programme

Day 1: Resources and Representations

Introducing Representation State Transfer (REST)

- Constraints and architectural styles

Meet the Frameworks: ASP.NET WebAPI

Working with read-only resources

- HTTP GET
- Hypermedia formats
- Content negotiation
- Resource expansion
- Layered systems, caching and scalability

Meet the Frameworks: NancyFX

Creating and updating resources

- Hypermedia PUT and DELETE
- Hypermedia POST and PATCH
- Beyond CRUD: modelling more complex domains

Day 2: Architecture and Infrastructure

Meet the Frameworks: ServiceStack

Security, monitoring and scalability

- Securing REST APIs
- Testing APIs
- Hosting APIs
- Scaling APIs

Meet the Frameworks: OpenRASTA

Integration and Collaboration

- Managing APIS
- Integrating
- Alternatives to REST - and why you might want to use them
- REST on your roadmap: making the business case for hypermedia APIs

Module Structure

Talk & Slides

Demo

Workshop

Discussion

Introduction

Defining REST

Architectural Constraints



UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

2000

"When given **a name, a
coordinated set of
**architectural
constraints****

becomes an

****architectural style"****



**"REST is software design on
the scale of decades:
every detail is intended to promote
software longevity and
independent evolution.**



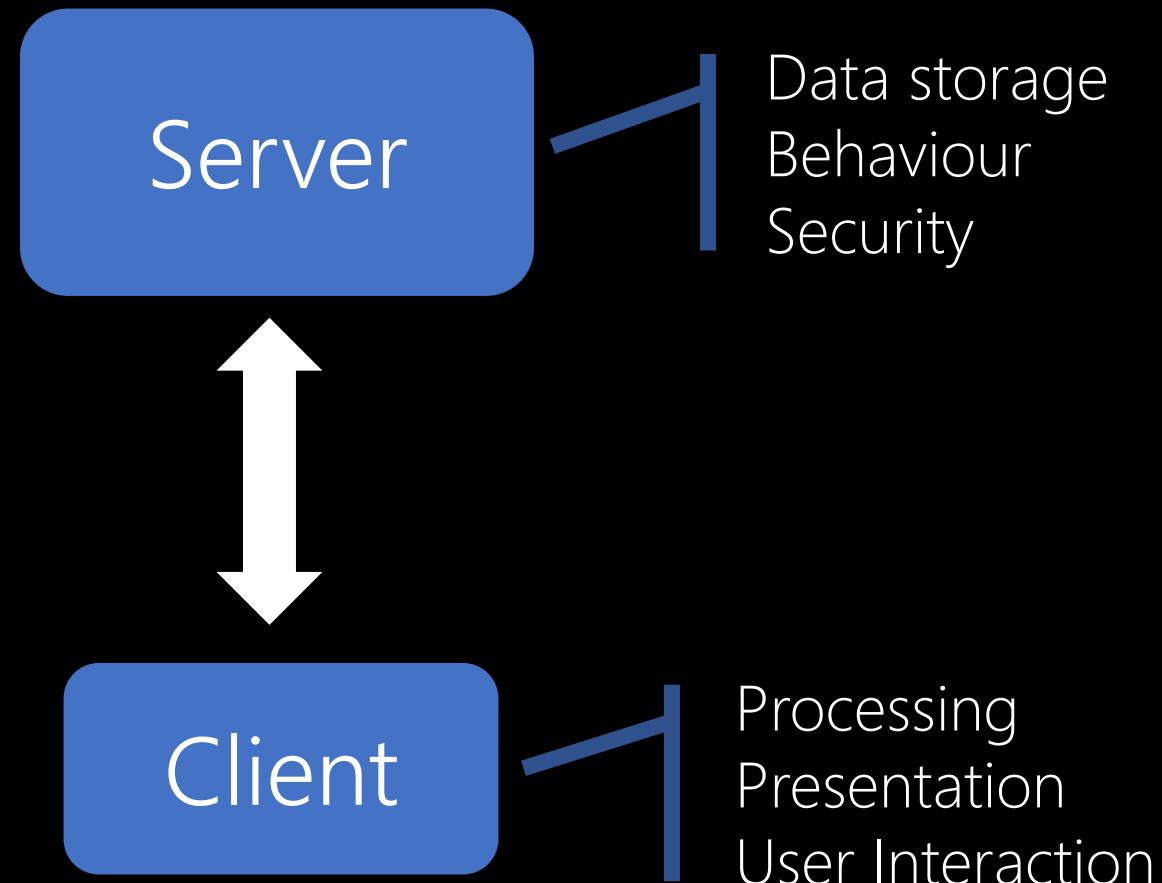
**Many of the constraints are
directly opposed to
short-term efficiency."**

Representational State Transfer (REST)

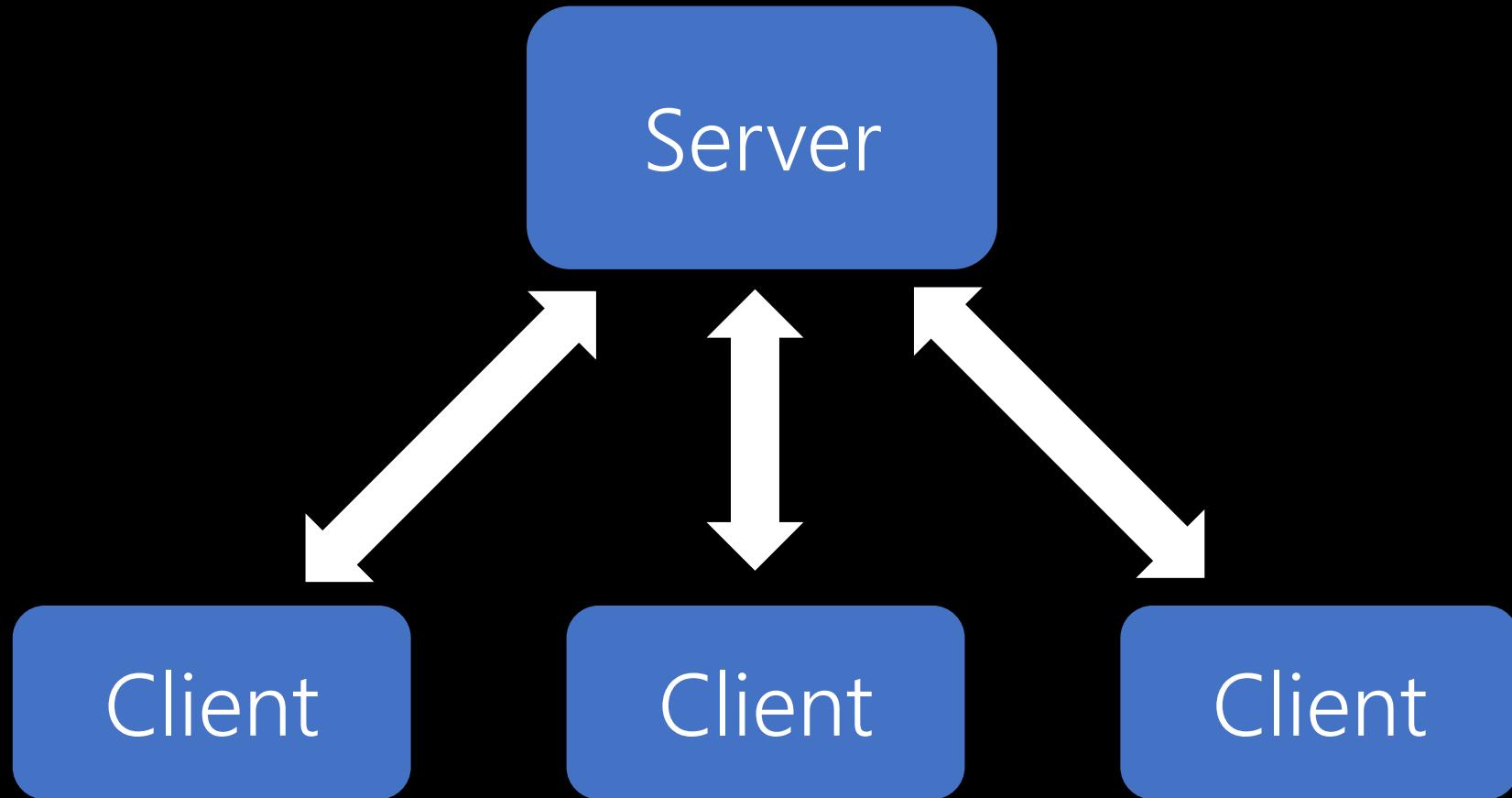
A coordinated set of architectural constraints...

...intended to promote software longevity and independent evolution

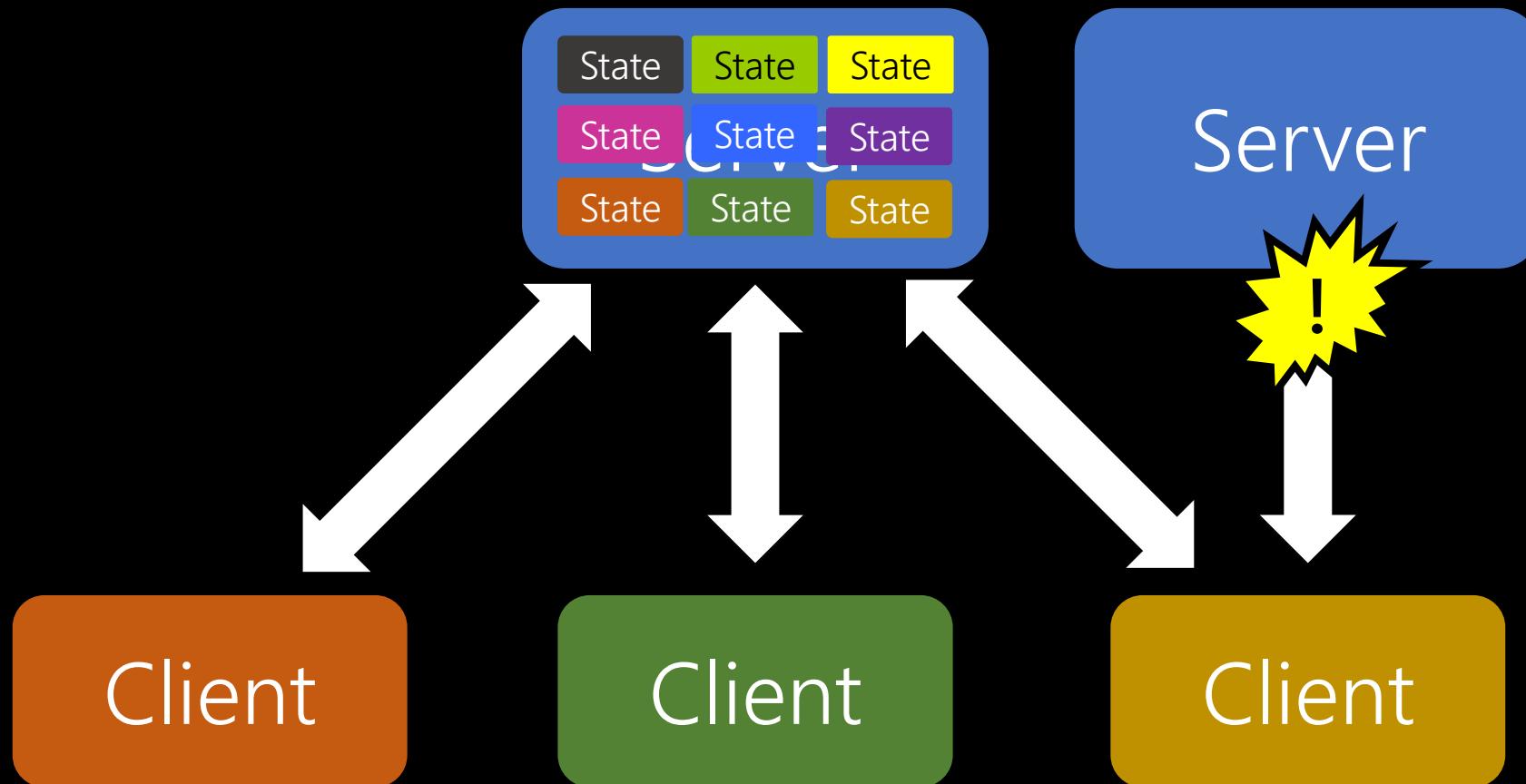
1. Client–Server



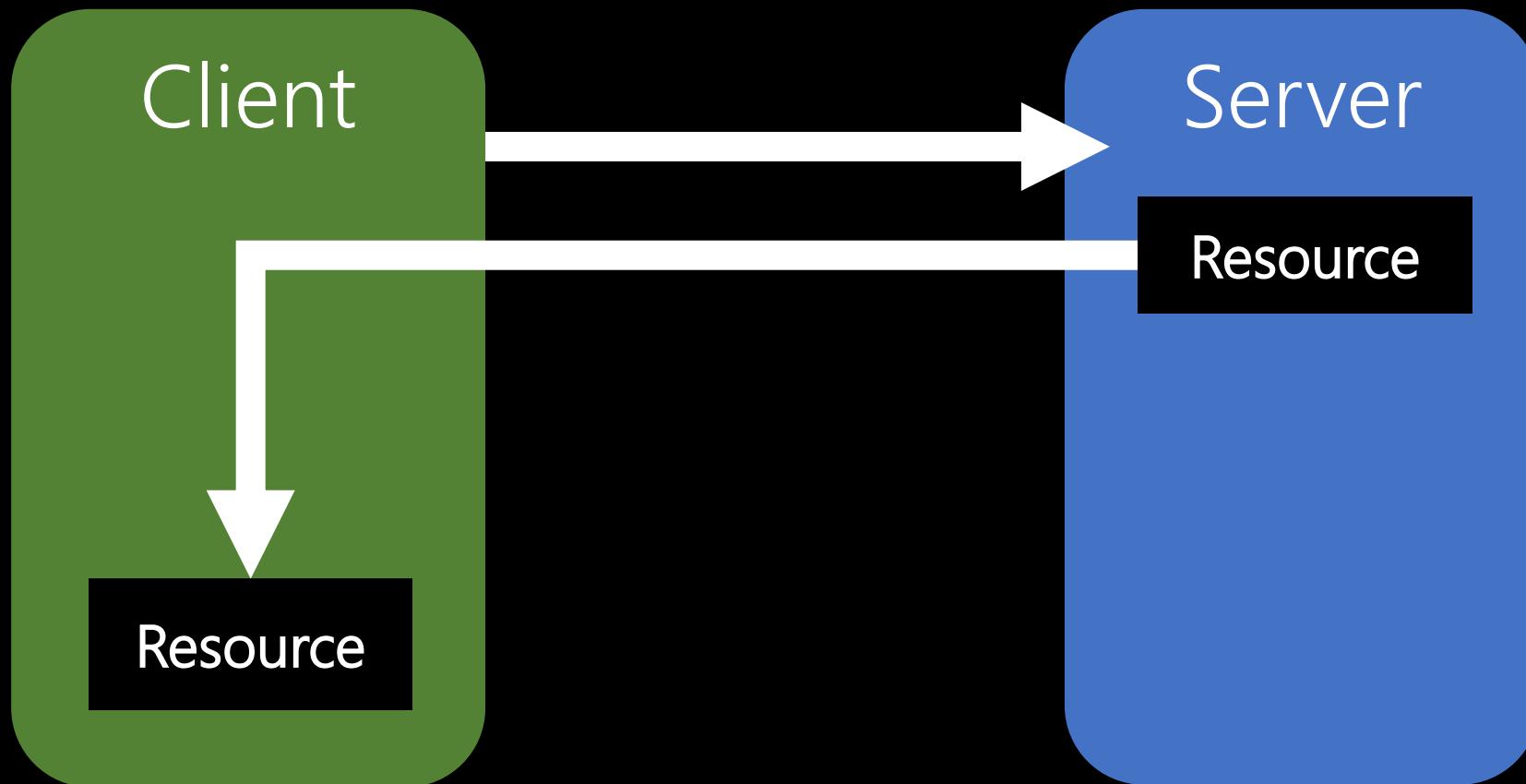
1. Client–Server



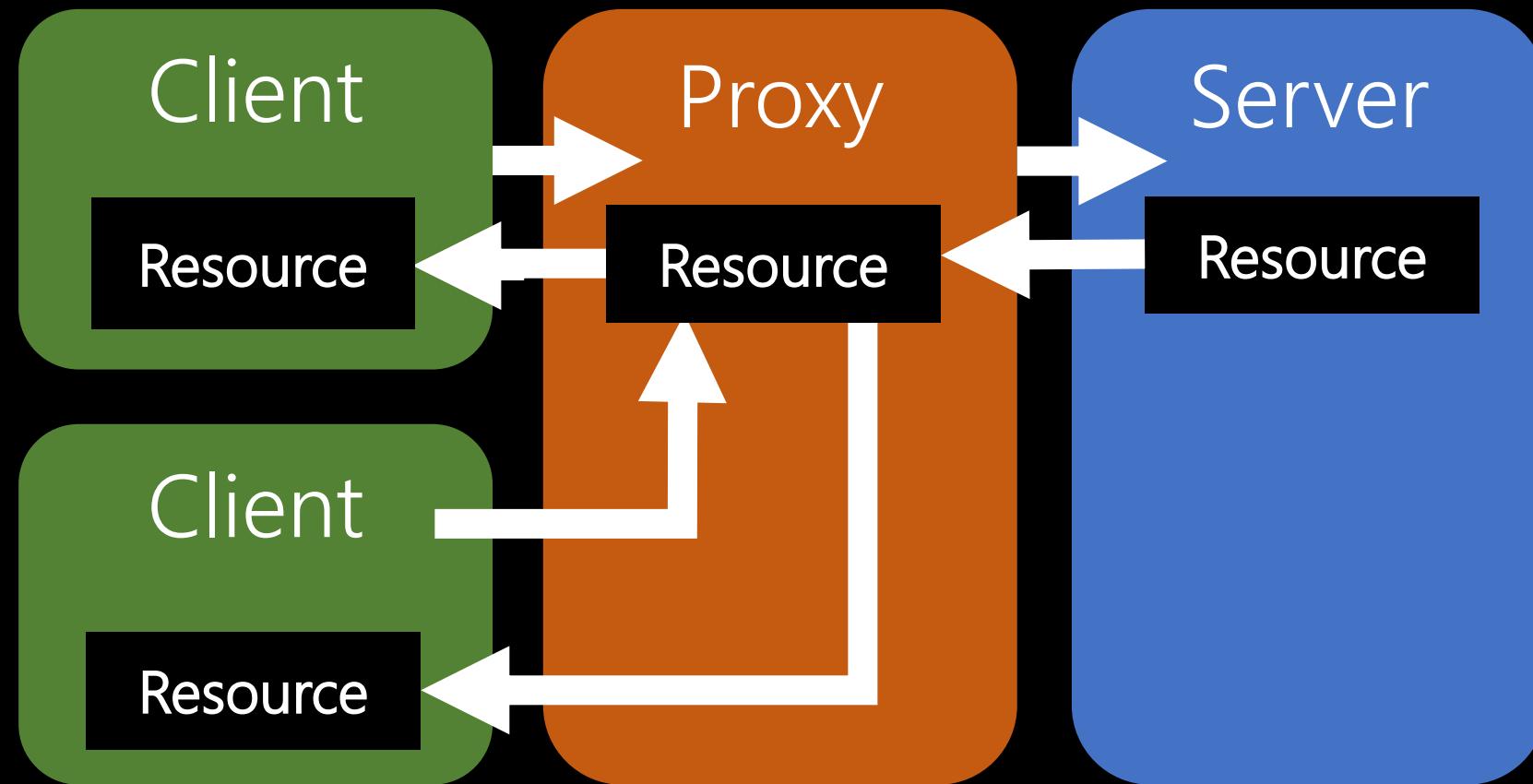
2. Stateless



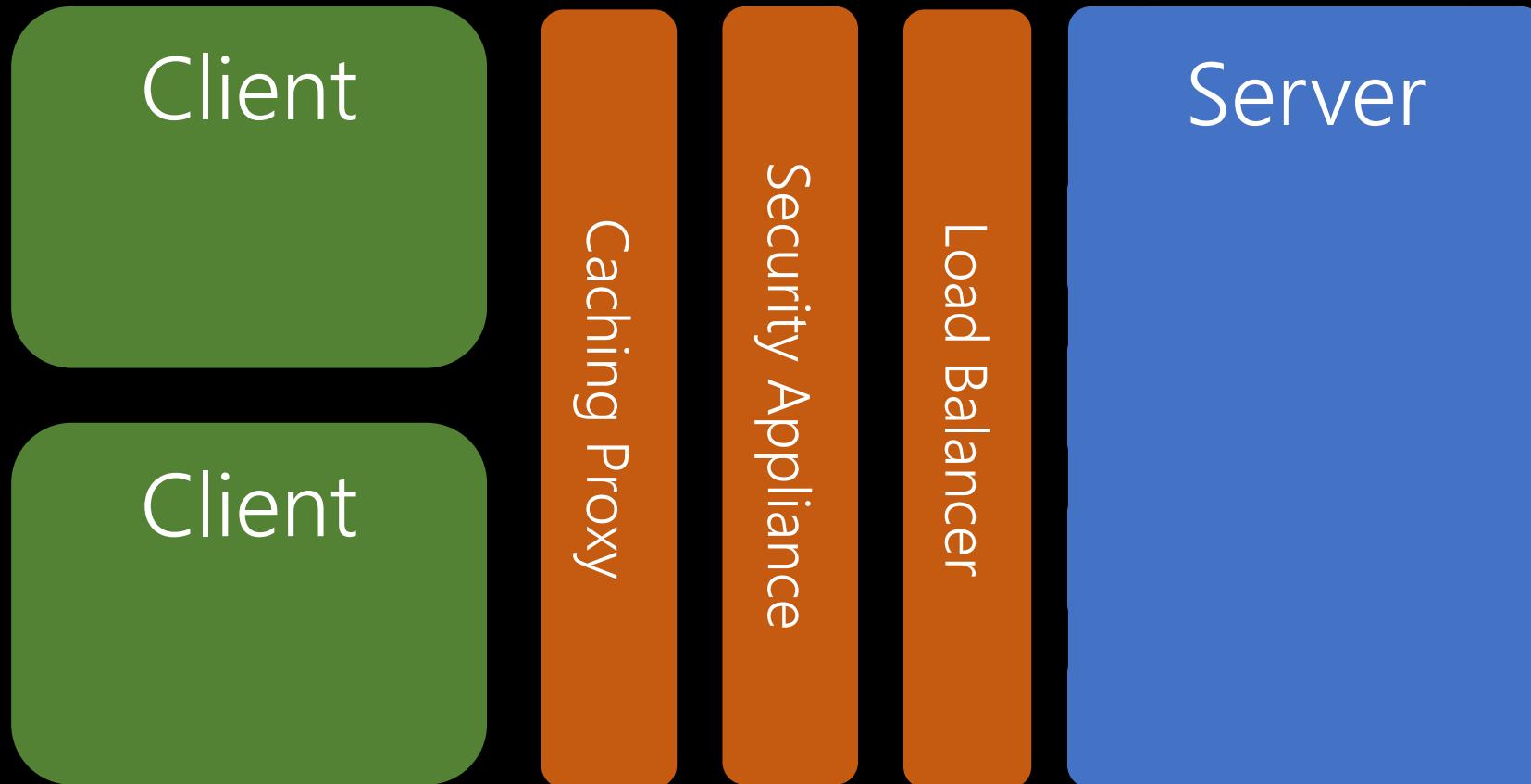
3. Cacheable



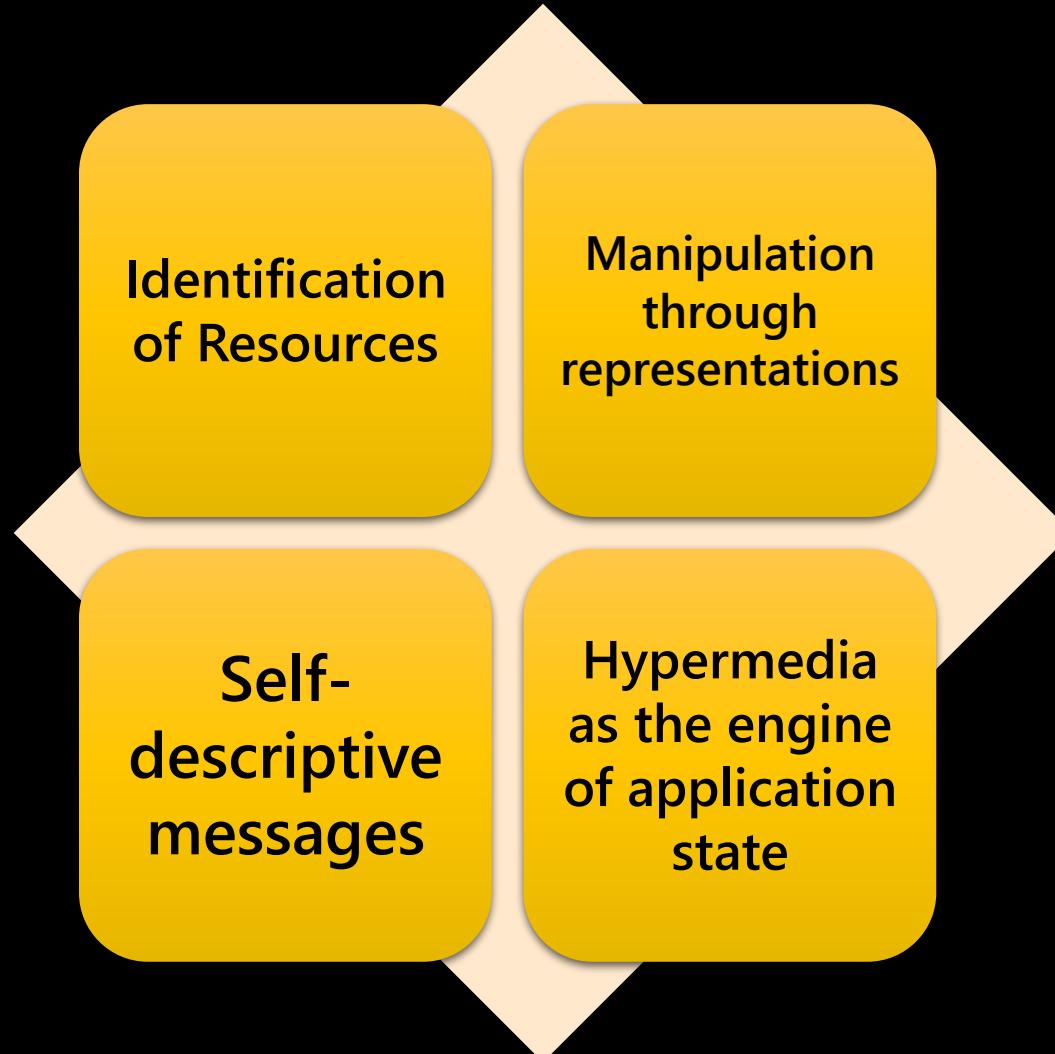
4. Layered System



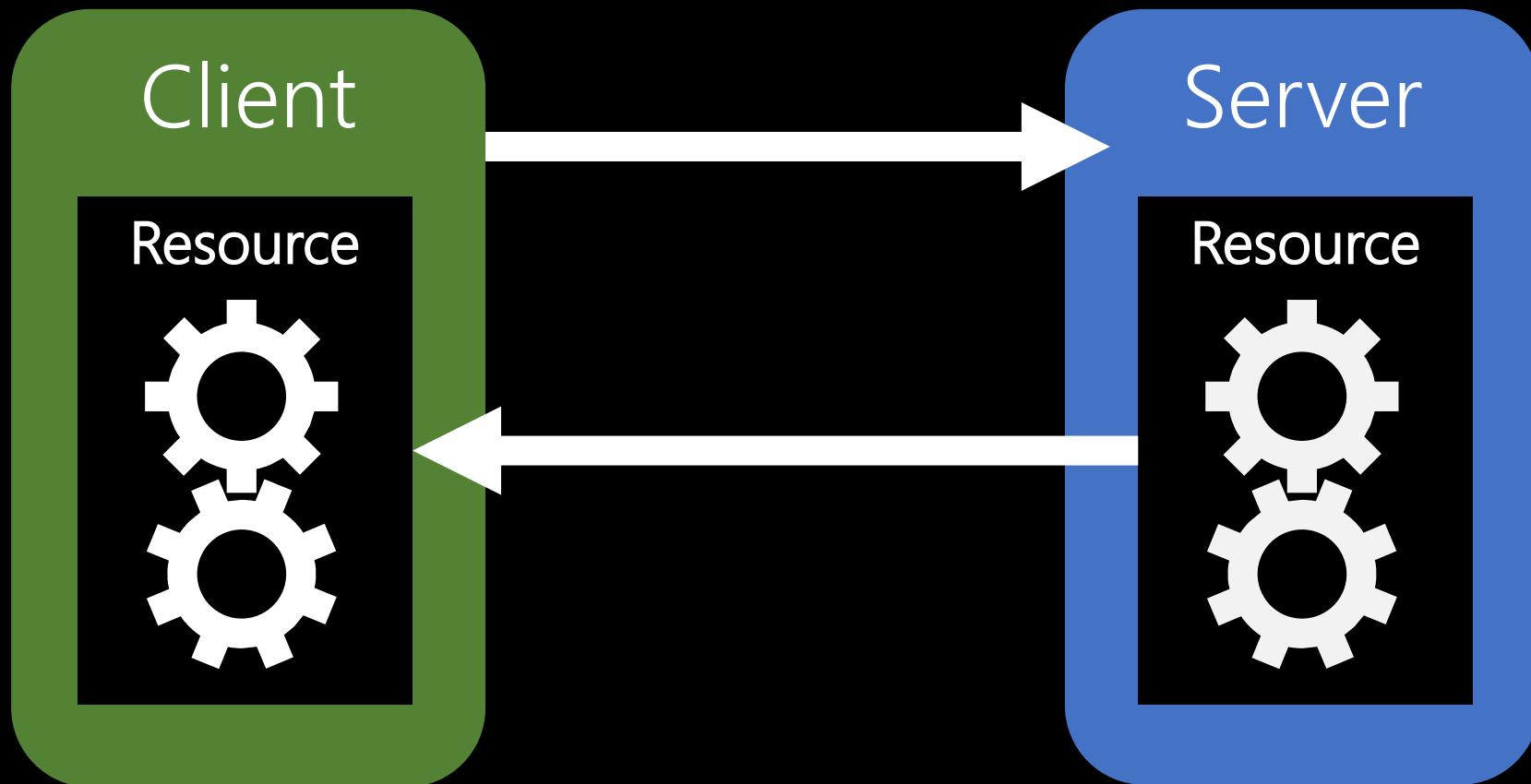
4. Layered System (continued)



5. Uniform Interface



6. Code on Demand



A Set of Architectural Constraints

1. Client-server
2. Stateless
3. Cacheable
4. Layered System
5. Uniformed Interface
6. Code-on-demand



OPTIONAL

Module 1:

Hypermedia as the
Engine of
Application State



GET / HTTP/1.1

200 OK

Content-Type: application/json

{

 "message": "Welcome to Herobook!"

}

GET /profiles HTTP/1.1

200 OK

Content-Type: application/json

[

{

 "username": "ironman",

 "name": "Tony Stark",

}

]

```
POST /profiles HTTP/1.1
```

```
{  
  "username" : "blackwidow"  
  "name": "Natasha Romanoff",  
}
```

201 Created

Location: <http://api.herobook.local/profiles/blackwidow>

```
{  
  "username": "blackwidow",  
  "name": "Natasha Romanoff"  
}
```

```
GET /profiles HTTP/1.1
```

```
200 OK
```

```
{
```

```
  "_links" : { ... },
```

```
  "items" : {
```

```
    { "username": "ironman", "name": "Tony Stark" }
```

```
    { "username": "blackwidow", "name": "Natasha Romanoff" },
```

```
    { "username": "spidey", "name": "Peter Parker" },
```

```
    { /* +2 million users! Wow! */ },
```

```
    { "username": "ducky", "name": "Howard the Duck" }
```

```
}
```

```
]
```

“Your API is using all
our bandwidth!”

“Your stupid API is
overloading our
database!”

GET /profiles?page=1 HTTP/1.1

200 OK

```
{  
  "_links" : { ... },  
  "items" : [  
    { "username":"ironman", "name":"Tony Stark" }  
    { "username":"spidey", "name":"Peter Parker" },  
    { "username":"blackwidow", "name": "Natasha Romanoff" },  
    { "username":"cap", "name": "Steve Rogers" },  
    { "username":"storm", "name": "Ororo Munroe" }  
  ]  
}
```

GET /profiles?page=2 HTTP/1.1

200 OK

GET /profiles?page=3 HTTP/1.1

200 OK

GET /profiles?page=4 HTTP/1.1

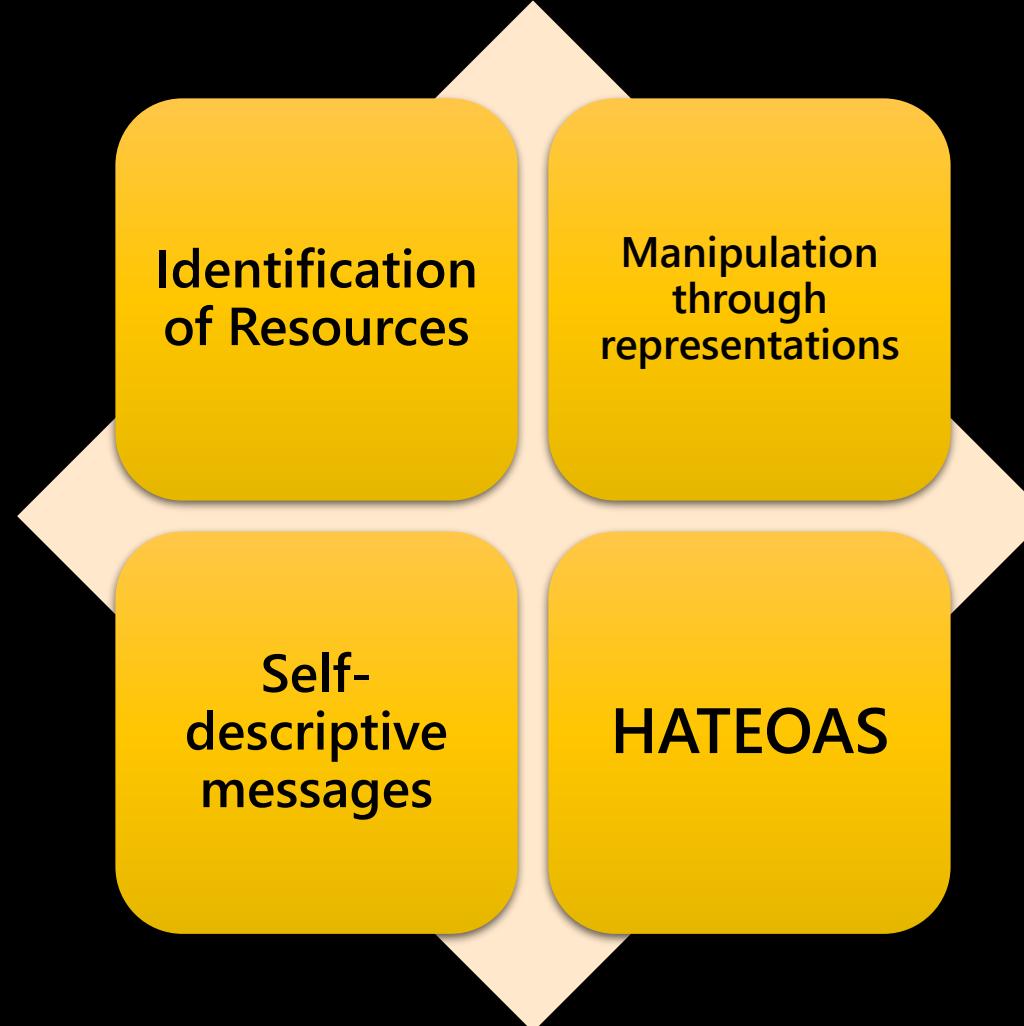
200 OK

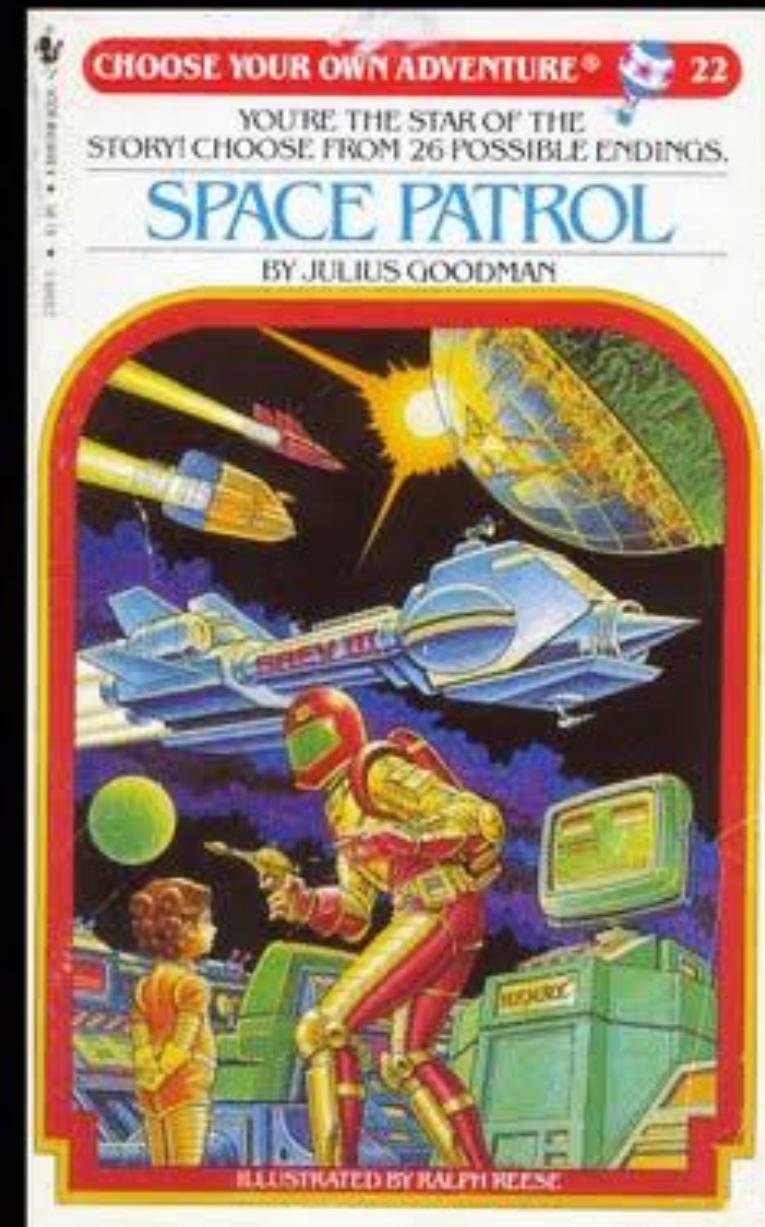
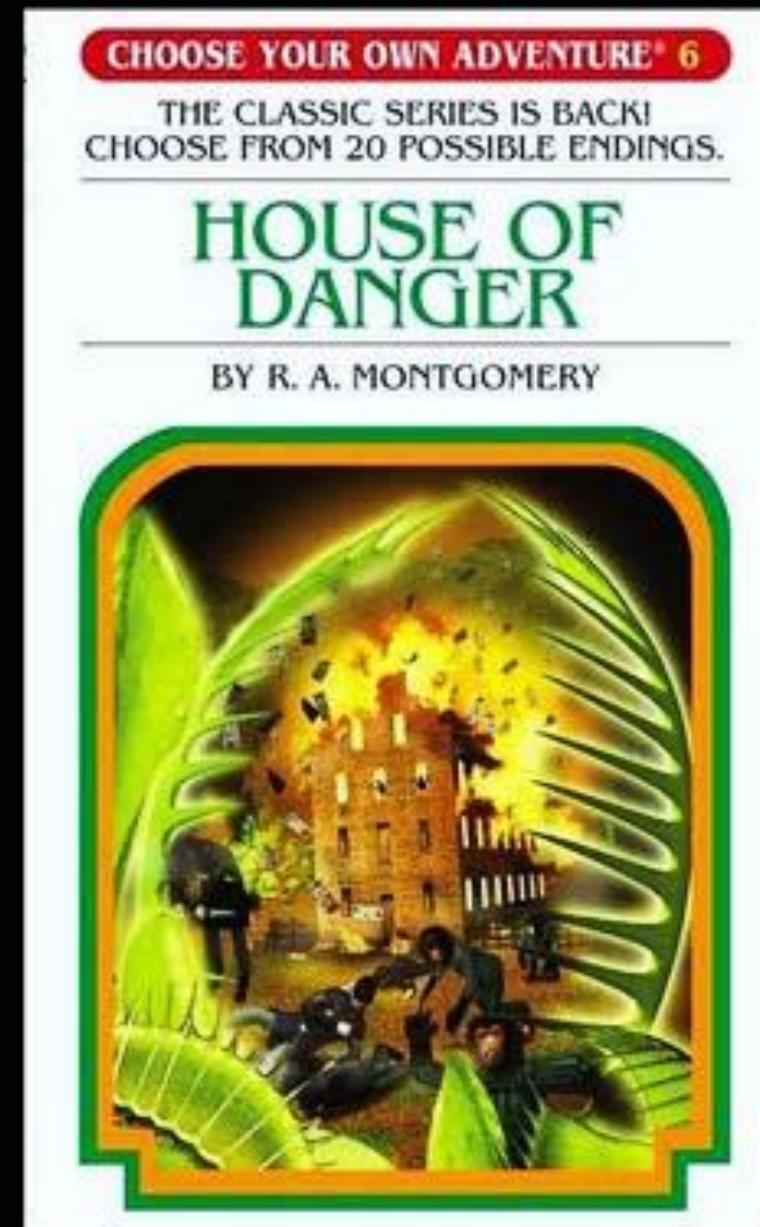
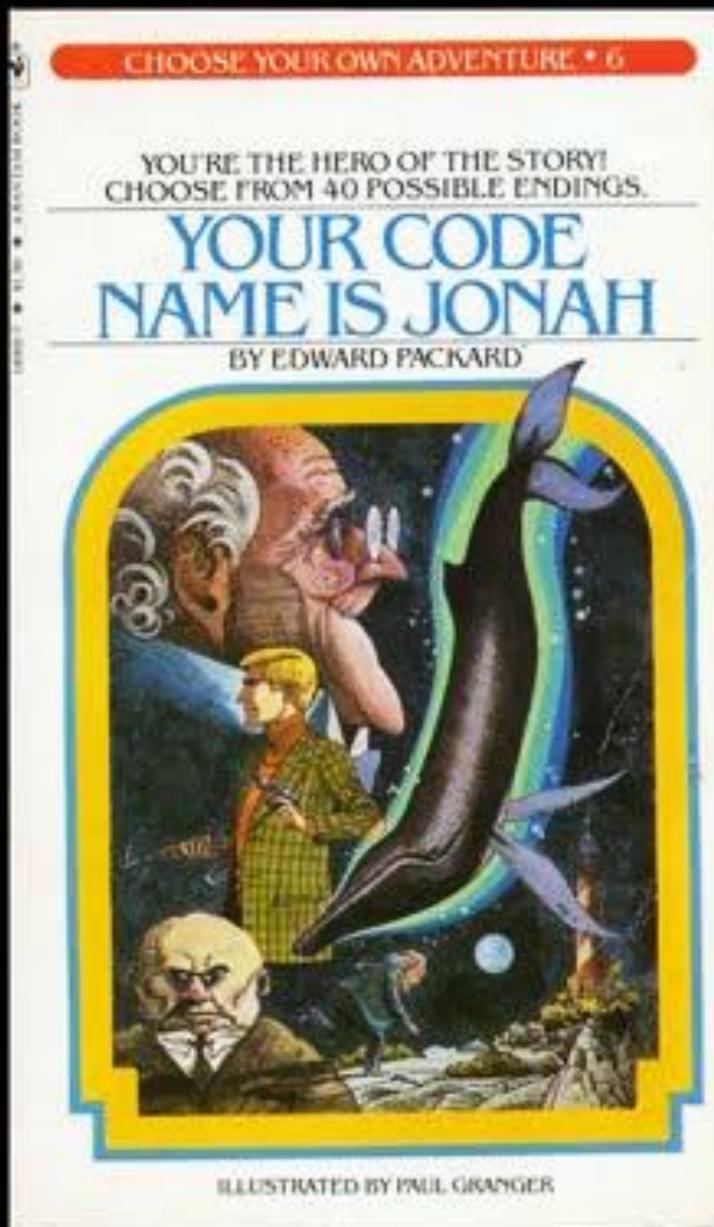
GET /profiles?page=5 HTTP/1.1

204 No Content

NOT REST

5. Uniform Interface





CHOOSE YOUR OWN ADVENTURE! #SELASDP

YOU'RE THE STAR OF THE STORY!
CHOOSE FROM 21 POSSIBLE ENDINGS!

HANDS-ON HYPERMEDIA

SELA DEVELOPER PRACTISE



It's Monday night in Tel Aviv... you've finished dinner, and you're deciding what to do later. You're going to a REST workshop at the Sela Developer Practise tomorrow, so you should probably get some sleep... but some friends are meeting up at Beerbazaar in Habima Square for drinks later, and you just found out Malox are playing tonight down at Rothschild 12. What are you going to do?

To go home and get some sleep, turn to page 24

To go to Beer Bazaar and try some Israeli craft beers, turn to page 41

To head down to Rothschild 12 and dance to Malox, turn to page 73

GET /profiles HTTP/1.1

200 OK

Content-Type: application/hal+json

```
{  
  "_links" : {  
  
  },  
  "count" : 5,  
  "index" : 0,  
  "total" : 223,  
  "items": [  
    { "username":"ironman", "name":"Tony Stark" }  
    { "username":"spidey", "name":"Peter Parker" },  
    { "username":"blackwidow", "name":"Натáша Романов" },  
    { "username":"pepper", "name":"Pepper Potts" },  
    { "username":"storm", "name":"Ororo Munroe" }  
  ]  
}
```

```
GET /profiles HTTP/1.1
```

```
200 OK
```

```
Content-Type: application/hal+json
```

```
{  
  "_links" : {  
    "self" : { "href" : "http://herobook/profiles?index=0" },  
    "next" : { "href" : "http://herobook/profiles?index=5" },  
    "last" : { "href" : "http://herobook/profiles?index=220" }  
  },  
  "count" : 5,  
  "index" : 0,  
  "total" : 223,  
  "items": [  
    { "username":"ironman", "name":"Tony Stark" }  
    { "username":"spidey", "name":"Peter Parker" },  
    { "username":"blackwidow", "name":"Натáша Романов" },  
    { "username":"pepper", "name":"Pepper Potts" },  
    { "username":"storm", "name":"Ororo Munroe" }  
  ]  
}
```

GET /profiles HTTP/1.1

200 OK

Content-Type: application/hal+json

```
{  
  "_links" : {  
    "self" : { "href" : "http://herobook/profiles?index=0" },  
    "next" : { "href" : "http://herobook/profiles?index=5" },  
    "last" : { "href" : "http://herobook/profiles?index=220" }  
  },  
  "count" : 5,  
  "index" : 0,  
  "total" : 223,  
  "items": [  

```

Module 1:

Hypermedia as the Engine of Application State

DEMO

← → ⌂ ⓘ www.herobook.local/explorer/explorer.html#/profiles?



Server:

<http://webapi.herobook.local>

Path:

/

Go [Reset](#)

Ready.

www.herobook.local/explorer/explorer.html#/



Server:

http://webapi.herobook.local

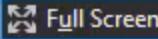
Path:

/

Go

[Reset](#)

Ready.



DynamicExtensions.cs

ProfilesController.cs*



index.html

Hal.cs

explorer.js

namespace RealWorldREST.WebAPI.Controllers {

```
[EnableCors(origins: "*", headers: "*", methods: "*")]
public class ProfilesController : ApiController {
    private readonly IDatabase db;

    public ProfilesController() {
        this.db = new DemoDatabase();
    }

    public object Get() {
        var profiles = db.ListProfiles();
        return profiles;
    }
}
```

184 %

Output

Ready

Ln 18

Col 46

Ch 46

INS

↑ 0

3

RealWorldREST

master ▾

← → ⌂ ⓘ www.herobook.local/explorer/explorer.html#/profiles

Server:

Path:

Go [Reset](#)

GET http://webapi.herobook.local/profiles

[

200 success

Pragma: no-cache

Content-Type: application/json; charset=utf-8

Cache-Control: no-cache

Expires: -1

[

{

 "Name": "Captain Lea Corbin",

 "Username": "4d"

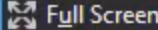
},

{

 "Name": "Steve Austin",

 "Username": "6milliondollarman"

},



DynamicExtensions.cs

ProfilesController.cs



index.html

Hal.cs

explorer.js

namespace RealWorldREST.WebAPI.Controllers

```
[EnableCors(origins: "*", headers: "*", methods: "*")]
public class ProfilesController : ApiController {
    private readonly IDatabase db;

    public ProfilesController() {
        this.db = new DemoDatabase();
    }

    public object Get() {
        var profiles = db.ListProfiles();

        var result = new {
            items = profiles
        };
        return result;
    }
}
```

184 %

Output

Ready

Ln 17

Col 30

Ch 30

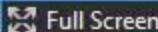
INS

↑ 0

3

RealWorldREST

master ▾



DynamicExtensions.cs

ProfilesController.cs

index.html

Hal.cs

explorer.js

namespace RealWorldREST.WebAPI.Controllers {

[EnableCors(origins: "*", headers: "*", methods: "*")]
public class ProfilesController : ApiController {

private readonly IDatabase db;

 public ProfilesController() {
 this.db = new DemoDatabase();
 } public object Get() {
 var RESULTS_PER_PAGE = 10;
 var profiles = db.ListProfiles().Take(RESULTS_PER_PAGE);

 var result = new {
 items = profiles
 };
 return result;
 }
}

184 %

Output

Ready

Ln 14

Col 42

Ch 42

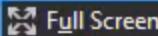
INS

↑ 0

3

RealWorldREST

master ▾



DynamicExtensions.cs

ProfilesController.cs



index.html



Hal.cs



explorer.js

```
private readonly IDatabase db;
```

```
public ProfilesController() {
    this.db = new DemoDatabase();
}
```

```
public object Get(int index = 0) {
    var RESULTS_PER_PAGE = 10;
    var profiles = db.ListProfiles()
        .Skip(index)
        .Take(RESULTS_PER_PAGE);
```

```
    var result = new {
        items = profiles
    };
    return result;
}
```

```
}
```

184 %

Output

Ready

Ln 23

Col 31

Ch 31

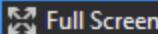
INS

↑ 0

3

RealWorldREST

master ▾



DynamicExtensions.cs ProfilesController.cs index.html Hal.cs* explorer.js

```
        }

        /// <summary>Generate a hypermedia links object containing first/final/prev/next links</summary>
        /// <param name="path">The absolute URL path to be decorated with paging querystring</param>
        /// <param name="index">The index of the first record on the current page</param>
        /// <param name="count">The count of items on each page</param>
        /// <param name="total">The total number of items in the collection</param>
        /// <returns></returns>

    public static dynamic Paginate(string path, int index, int count, int total) {
        dynamic _links = new ExpandoObject();
        var maxIndex = total - 1;
        _links.first = Href($"{path}?index=0");
        _links.final = Href($"{path}?index={maxIndex - maxIndex % count}");
        if (index > 0) _links.last = Href($"{path}?index={index - count}");
        if (index + count < maxIndex)
            _links.next = Href($"{path}?index={index + count}");
        return _links;
    }
}
```

167 %

Output

Ready

Ln 14

Col 9

Ch 9

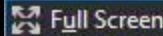
INS

↑ 0

4

RealWorldREST

master ▾



DynamicExtensions.cs

ProfilesController.cs

index.html

Hal.cs

explorer.js

```
        }

        public object Get(int index = 0) {
            var RESULTS_PER_PAGE = 10;
            var profiles = db.ListProfiles()
                .Skip(index)
                .Take(RESULTS_PER_PAGE);

            var result = new {
                _links = new {
                    next = new {
                        href = $"/profiles?index={index+RESULTS_PER_PAGE}"
                    }
                },
                items = profiles
            };
            return result;
        }
    }
```

- Workshop: Hypermedia as the Engine of Application State

Code:

<http://www.github.com/dylanbeattie/ndc2017/>

Follow instructions in workshop.pdf

Module 1: Hypermedia as the Engine of Application State

Discussion Points

- Do linked resources have to be on the same server?
- How could you design a partition scheme that would split your user profile data across multiple database servers?

Module 2:

Resource and Resource Composition

GET /profiles/ironman HTTP/1.1

200 OK

Content-Type: application/json

{

 "username": "ironman",

 "name": "Tony Stark",

 "friends" : [

 { "username":"ironman", "name":"Tony Stark" },

 { "username":"spidey", "name":"Peter Parker" },

 { "username":"blackwidow", "name":"Натáша Романов" },

 /* another 500 friends here ... */

]

}

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json
{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username": "hulk", "name": "Bruce Banner" },
    { "username": "spidey", "name": "Peter Parker" },
    { "username": "blackwidow", "name": "Натáша Романов" },
    /* another 500 friends here ... */
  ],
  "updates" : [
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json
{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username": "hulk", "name": "Bruce Banner" },
    { "username": "spidey", "name": "Peter Parker" },
    { "username": "blackwidow", "name": "Натáша Романов" },
    /* another 500 friends here ... */
  ],
  "updates" : [
    {
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json
{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username": "hulk", "name": "Bruce Banner" },
    { "username": "spidey", "name": "Peter Parker" },
    { "username": "blackwidow", "name": "Натáша Романов" },
    /* another 500 friends here ... */
  ],
  "updates" : [
    {
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
    {
      "update" : "Selfie stick broke. Oh well. Back to the drawing board",
      "posted" : "2016-04-17T08:26:13.511Z"
    }
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json
{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username": "hulk", "name": "Bruce Banner" },
    { "username": "spidey", "name": "Peter Parker" },
    { "username": "blackwidow", "name": "Натáша Романов" },
    /* another 500 friends here ... */
  ],
  "updates" : [
    { "id" : 1234,
      "update" : "Working a new Iron Man suit!",
      "posted" : "2016-02-23T18:25:43.511Z"
    },
    { "id" : 1543,
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
    { "id" : 1782,
      "update" : "Selfie stick broke. Oh well. Back to the drawing board",
      "posted" : "2016-04-17T08:26:13.511Z"
    }
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json

{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username": "hulk", "name": "Bruce Banner", "updates" : [ { . . . }, { . . . }, { . . . } ] },
    { "username": "spidey", "name": "Peter Parker", "updates" : [ { . . . }, { . . . }, { . . . } ] },
    { "username": "blackwidow", "name": "Натáша Романов", "updates" : [ { . . . }, { . . . }, { . . . } ] },
    /* another 500 friends here ... */
  ],
  "updates" : [
    { "id" : 1234,
      "update" : "Working a new Iron Man suit!",
      "posted" : "2016-02-23T18:25:43.511Z"
    },
    { "id" : 1543,
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
    { "id" : 1782,
      "update" : "Selfie stick broke. Oh well. Back to the drawing board",
      "posted" : "2016-04-17T08:26:13.511Z"
    }
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json

{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username":"hulk", "name":"Bruce Banner", "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [ { . . . }, { . . . }, { . . . } ],
    },
    { "username":"spidey", "name":"Peter Parker", "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [ { . . . }, { . . . }, { . . . } ],
    },
    { "username":"blackwidow", "name":"Натáша Романов", "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [ { . . . }, { . . . }, { . . . } ],
    },
    /* another 500 friends here... */
  ],
  "updates" : [
    { "id" : 1234,
      "update" : "Working a new Iron Man suit!",
      "posted" : "2016-02-23T18:25:43.511Z"
    },
    { "id" : 1543,
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
    { "id" : 1782,
      "update" : "Selfie stick broke. Oh well. Back to the drawing board",
      "posted" : "2016-04-17T08:26:13.511Z"
    }
  ]
}
```

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json

{
  "username": "ironman",
  "name": "Tony Stark",
  "friends" : [
    { "username":"hulk", "name":"Bruce Banner",
      "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [
        { "updates" : { . . . } },
        { "updates" : { . . . } },
        { "updates" : { . . . } }
      ]
    },
    { "username":"spidey", "name":"Peter Parker",
      "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [
        { "updates" : { . . . } },
        { "updates" : { . . . } },
        { "updates" : { . . . } }
      ]
    },
    { "username":"blackwidow", "name":"Натáша Романов",
      "updates" : [ { . . . }, { . . . }, { . . . } ],
      "friends" : [
        { "updates" : { . . . } },
        { "updates" : { . . . } },
        { "updates" : { . . . } }
      ]
    },
    /* another 500 friends here... */
  ],
  "updates" : [
    { "id" : 1234,
      "update" : "Working a new Iron Man suit!",
      "posted" : "2016-02-23T18:25:43.511Z"
    },
    { "id" : 1543,
      "update" : "New suit's gonna have a selfie stick! Oh yeah!",
      "posted" : "2016-04-16T18:25:43.511Z"
    },
    { "id" : 1782,
      "update" : "Selfie stick broke. Oh well. Back to the drawing board",
      "posted" : "2016-04-17T08:26:13.511Z"
    }
  ]
}
```

BIG DATA!

"Your API is using
all our
bandwidth...
AGAIN!"

"Your API is
overloading the
database. Again.

...I told you we
should have just
stuck with PHP..."

GET /profiles/ironman HTTP/1.1

200 OK

Content-Type: application/hal+json

{

 "_links": {
 "self": { "href" "/profiles/ironman" },
 "friends": { "href" : "/profiles/ironman/friends" },
 "photos": { "href" : "/profiles/ironman/photos" },
 "updates": { "href" : "/profiles/ironman/updates" }
 },

 "username" : "ironman",

 "name" : "Tony Stark"

}

Module 2:

Demo:

Resources and Navigation Properties

Decompositing Aggregate Resources

Module 2:

Workshop

Module 2: Resource Composition

Discussion Points

- Are there situations where this might be the best solution?
- What are the similarities between this and eager vs lazy loading in an object-relational mapper?

Module 3:

Sub-resources and Resource Expansion

GET /profiles/ironman HTTP/1.1

200 OK

GET /profiles/ironman/friends HTTP/1.1

200 OK

GET /profiles/ironman/updates HTTP/1.1

200 OK

GET /profiles/ironman/photos HTTP/1.1

200 OK

GET /profiles/ironman/photos/1234 HTTP/1.1

200 OK

GET /profiles/ironman/photos/1234/comments HTTP/1.1

200 OK

GET /profiles/ironman/photos/1345 HTTP/1.1

200 OK

GET /profiles/ironman/photos/1345/comments HTTP/1.1

200 OK

GET /profiles/ironman/photos/1456 HTTP/1.1

200 OK

GET /profiles/ironman/photos/1456/comments HTTP/1.1

200 OK

"I need to make 50 API calls just to draw one web page!"

"Our web server crashed because the IIS logs filled up the C: drive!
We never had this problem with PHP..."

GET /profiles/ironman

HTTP/1.1

200 OK

Content-Type: application/json

```
{  
  "_links": {  
    "self" : { "href": "/profiles/ironman" },  
    "friends" : { "href": "/profiles/ironman/friends" },  
    "photos" : { "href": "/profiles/ironman/photos" },  
    "updates" : { "href": "/profiles/ironman/updates" }  
  },  
  "username" : "ironman",  
  "name" : "Tony Stark"  
}
```

```
GET /profiles/ironman?expand=updates HTTP/1.1
200 OK
Content-Type: application/json
{
  "_links": {
    "self" : { "href": "/profiles/ironman" },
    "friends" : { "href": "/profiles/ironman/friends" },
    "photos" : { "href": "/profiles/ironman/photos" },
    "updates" : { "href": "/profiles/ironman/updates" }
  },
  "username" : "ironman",
  "name" : "Tony Stark"
  "_embedded" : {
  }
}
```

```
GET /profiles/ironman?expand=updates HTTP/1.1
200 OK
Content-Type: application/json
{
  "_links": {
    "self" : { "href": "/profiles/ironman" },
    "friends" : { "href": "/profiles/ironman/friends" },
    "photos" : { "href": "/profiles/ironman/photos" },
    "updates" : { "href": "/profiles/ironman/updates" }
  },
  "username" : "ironman",
  "name" : "Tony Stark"
  "_embedded" : {
    "updates" : [
      {
        "update" : "Working a new Iron Man suit – with a built-in selfie stick!",
        "posted" : "2016-02-23T18:25:43.511Z"
      },
      {
        "update" : "Selfie stick broke. Oh well. Back to the drawing board",
        "posted" : "2016-04-17T08:26:13.511Z"
      }
    ]
  }
}
```

Module 3:

Demo:

Resource Expansion

www.herobook.local/explorer/explorer.html#/profiles/ironman



Server:

http://nancy.herobook.local

Path:

endpoint

Go [Reset](#)

Ready.

← → ⌂ ⓘ www.herobook.local/explorer/explorer.html#/profiles/ironman

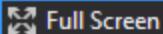


Server:

Path:

Go [Reset](#)

Ready.



DynamicExtensions.cs

ProfilesModule.cs

```
namespace RealWorldRest.NancyFX.Modules {
    public class ProfilesModule : NancyModule {
        private readonly IDatabase db;

        public ProfilesModule(IDatabase db) {
            this.db = db;
            Get["/profiles"] = _ => GetProfiles();
            Get["/profiles/{username}"] = args => GetProfile(args.username);
            Get["/profiles/{name}/friends"] = args => GetFriends(args.name);

            Post["/profiles"] = args => PostProfile(args);
            Post["/profiles/{name}/friends"] = args => PostFriendship(args);
        }

        private dynamic GetProfile(string username) {
            var profile = db.LoadProfile(username);
            return profile;
        }
    }
}
```

184 %

Output

Item(s) Saved

Ln 11

Col 3

Ch 3

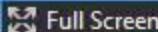
INS

↑ 0

2

RealWorldREST

master ▾



DynamicExtensions.cs

ProfilesModule.cs

```
Get["/profiles"] = _ => GetProfiles();
Get["/profiles/{username}"] = args => GetProfile(args.username);
Get["/profiles/{name}/friends"] = args => GetFriends(args.name);

Post["/profiles"] = args => PostProfile(args);
Post["/profiles/{name}/friends"] = args => PostFriendship(args);
}

private dynamic GetProfile(string username) {
    var profile = db.LoadProfile(username);
    return profile;
}

private dynamic GetProfiles() {
    var index = (int?)Request.Query["index"] ?? 0;
    var count = 10;
    var profiles = db.ListProfiles().Skip(index).Take(count);
    var total = db.CountProfiles();
    var links = Hal_Paginate("/profiles", index, count, total);
```

184 %

Output

Ready

Ln 25

Col 46

Ch 46

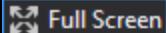
INS

↑ 0

↓ 2

RealWorldREST

master ▾



DynamicExtensions.cs ✎ ProfilesModule.cs

```
namespace RealWorldRest.NancyFX.Modules {
    public static class ObjectExtensions {
        public static dynamic ToDynamic(this object value) {
            IDictionary<string, object> expando = new ExpandoObject();
            var properties = TypeDescriptor.GetProperties(value.GetType());
            foreach (PropertyDescriptor property in properties) {
                expando.Add(property.Name, property.GetValue(value));
            }
            return (ExpandoObject)expando;
        }

        public static IDictionary<string, object> ToDictionary(this object d) {
            return d.GetType().GetProperties()
                .ToDictionary(x => x.Name, x => x.GetValue(d, null));
        }
    }
}
```

184 %

Output

Ready

Ln 22

Col 2

Ch 2

INS

↑ 0

↓ 2

RealWorldREST

master ▾

```
Get["/profiles"] = _ => GetProfiles();
Get["/profiles/{username}"] = args => GetProfile(args.username);
Get["/profiles/{name}/friends"] = args => GetFriends(args.name);

Post["/profiles"] = args => PostProfile(args);
Post["/profiles/{name}/friends"] = args => PostFriendship(args);
}

private dynamic GetProfile(string username) {
    var profile = db.LoadProfile(username);

    profile._links = new { };
    return profile;
}

private dynamic GetProfiles() {
    var index = (int?)Request.Query["index"] ?? 0;
    var count = 10;
    var profiles = db.ListProfiles().Skip(index).Take(count).
```

(local variable) int count

www.herobook.local/explorer/explorer.html#/profiles/ironman

Server:

Path:

[Go](#) [Reset](#)

```
GET http://nancy.herobook.local/profiles/ironman
```

200 success

Content-Type: application/json; charset=utf-8

Cache-Control: private

```
{
  "name": "Tony Stark",
  "username": "ironman"
}
```

```
private dynamic GetProfile(string username) {
    var profile = db.LoadProfile(username).ToDynamic();

    profile._links = new {
        self = new { href = $"{"/profiles/{username}"}" },
        friends = new { href = $"{"/profiles/{username}/friends"}" }
    };

    return profile;
}

private dynamic GetProfiles() {
    var index = (int?)Request.Query["index"] ?? 0;
    var count = 10;
    var profiles = db.ListProfiles().Skip(index).Take(count);
    var total = db.CountProfiles();
    var links = Hal.Paginate("/profiles", index, count, total);
}
```

Module 3:

Workshop Exercise:

Resource Expansion

Module 4:

Going beyond GET:

Supporting Actions with Hypermedia

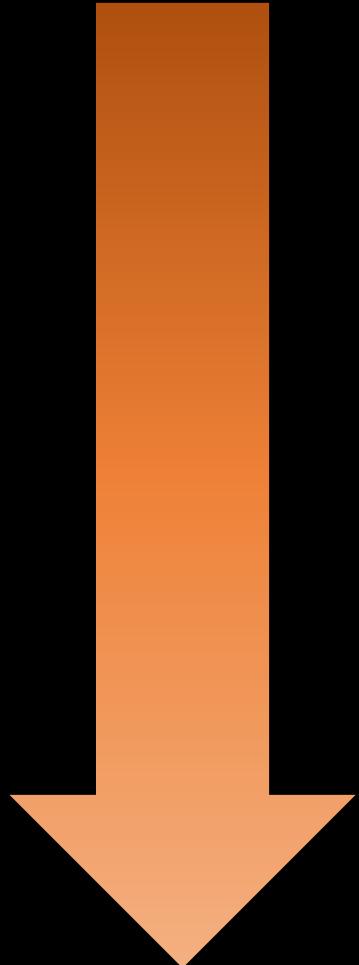
The Richardson Maturity Model

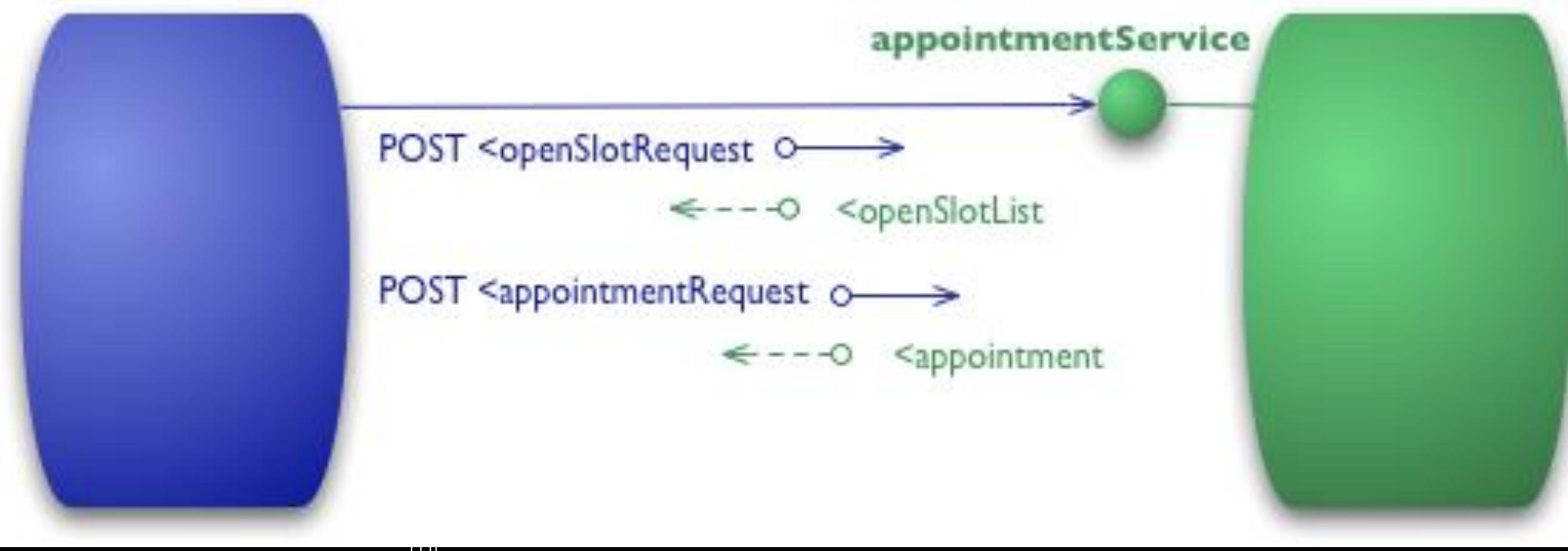
Level 0: The Swamp of POX

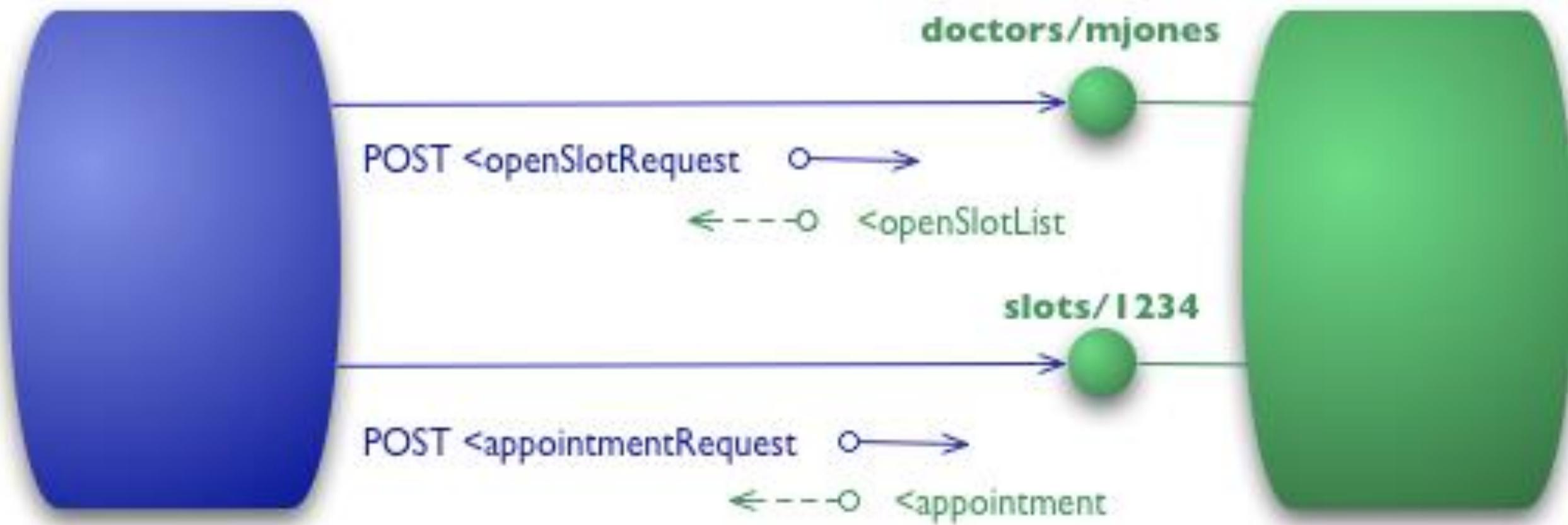
Level 1: Resources

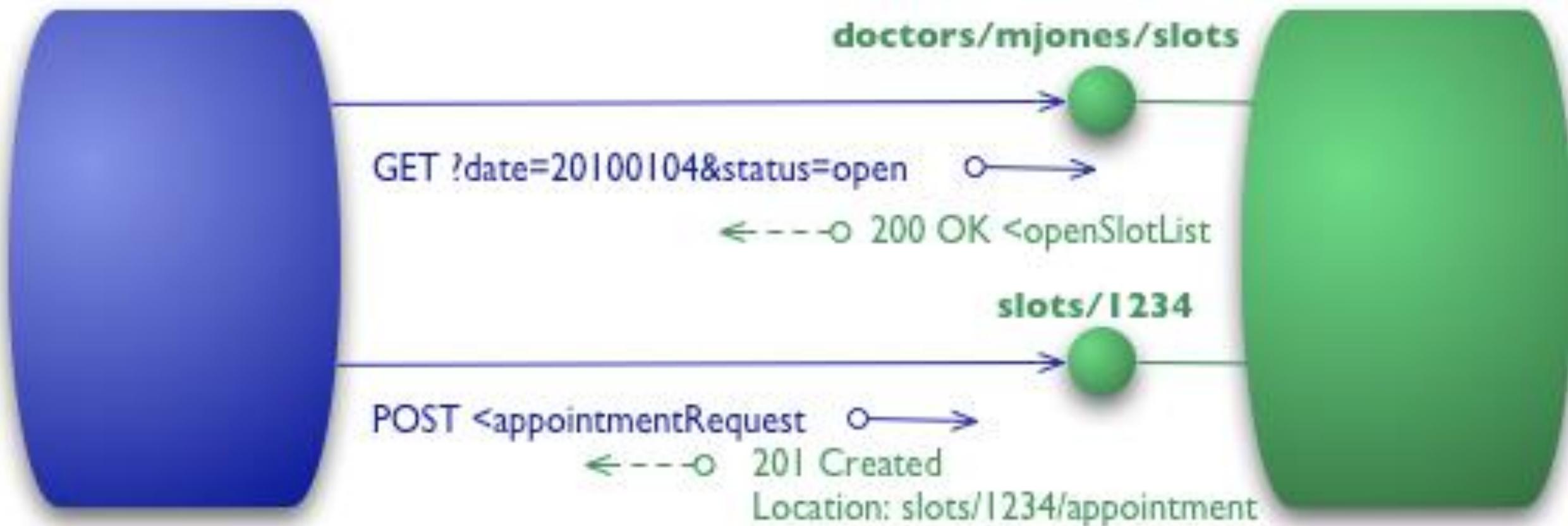
Level 2: HTTP verbs

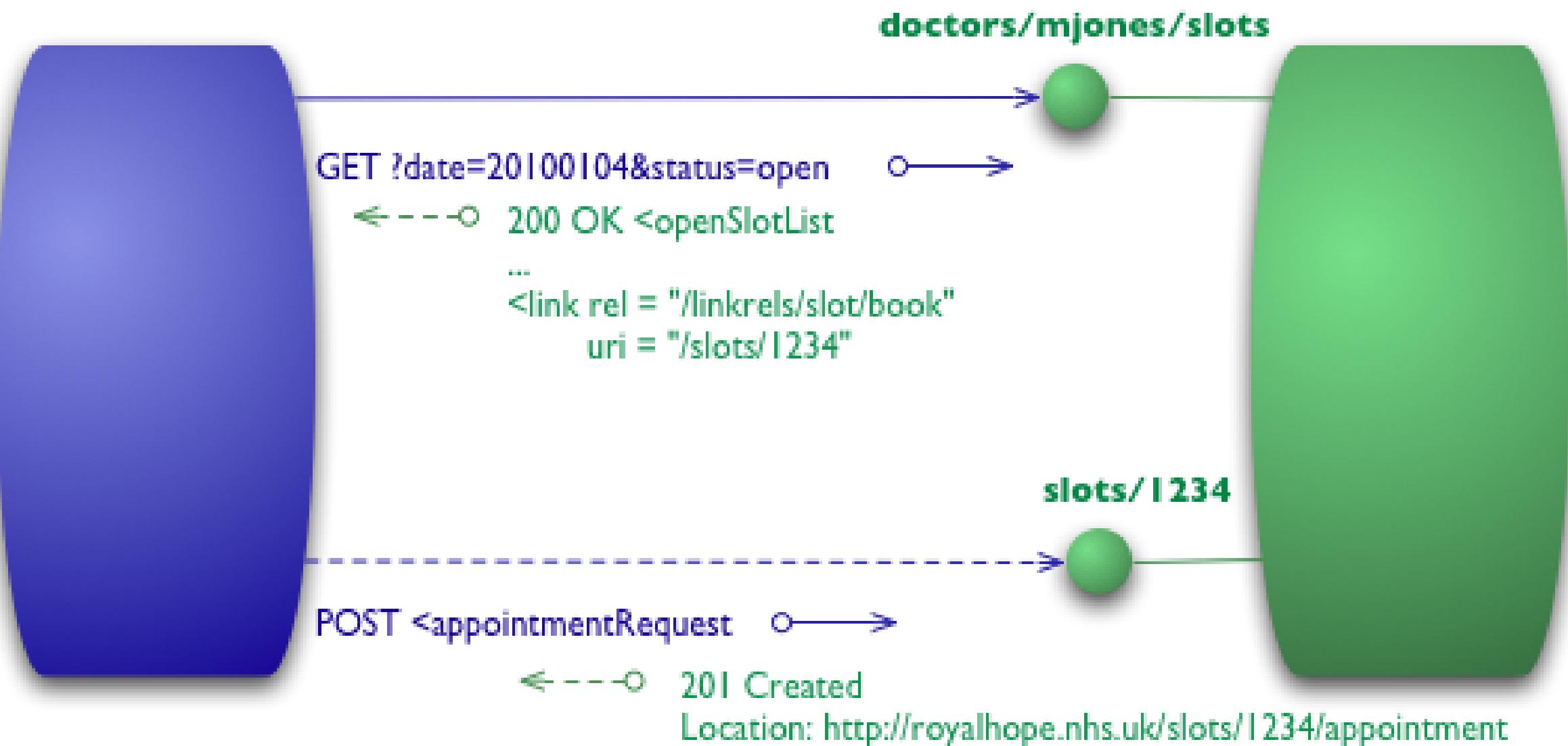
Level 3: Hypermedia Controls











```
GET /profiles HTTP/1.1
```

```
200 OK
```

```
Content-Type: application/hal+json
```

```
{
```

```
    "items" : [  
        { "username": "ironman", "name": "Tony Stark" },  
        { "username": "blackwidow", "name": "Natasha Romanoff" }  
    ]  
}
```

```
GET /profiles HTTP/1.1
```

```
200 OK
```

```
Content-Type: application/hal+json
```

```
{  
  "_actions" : {  
    "create" : {  
      "name": "Create a Profile",  
      "type": "application/json",  
      "href" : "/profiles",  
      "method": "POST"  
    }  
  },  
  "items" : [  
    { "username": "ironman", "name": "Tony Stark" },  
    { "username": "blackwidow", "name": "Natasha Romanoff" }  
  ]  
}
```

```
GET /profiles HTTP/1.1
```

```
200 OK
```

```
Content-Type: application/vnd.herobook.hal+json
```

```
{  
    "_actions" : {  
        "create" : {  
            "name": "Create a Profile",  
            "type": "application/json",  
            "href" : "/profiles",  
            "method": "POST"  
        }  
    },  
    "items" : [  
        { "username": "ironman", "name": "Tony Stark" },  
        { "username": "blackwidow", "name": "Natasha Romanoff" }  
    ]  
}
```

Module 4:

Walkthrough:

Supporting Actions with Hypermedia

Module 4:

Workshop Exercise:

Supporting Actions with Hypermedia

Module 5:

Implementing

HTTP PATCH

```
GET /profiles/ironman HTTP/1.1
200 OK
Content-Type: application/json
{
  "_links": {
    "self" : { "href": "/profiles/ironman" },
    "friends" : { "href": "/profiles/ironman/friends" },
    "photos" : { "href": "/profiles/ironman/photos" },
    "updates" : { "href": "/profiles/ironman/updates" }
  },
  "name": "Tony Stark",
  "username": "ironman",
  "height": 192,
  "weight": 85,
  "location": { "latitude": 59.93, "longitude": 30.33 },
  "status": "Out saving the world. Again.",
  "hometown": "Malibu, USA"
  "email": "tony@stark.com",
  "website": "www.ironman.com",
  "last_modified" : "2015-08-12T19:45:43.511Z"
}
```

```
PUT /profiles/ironman HTTP/1.1
Content-Type: application/json
{
  "name": "Tony Stark",
  "username": "ironman",
  "height": 192,
  "weight": 85,
  "location": { "lat": 34.02, "lon": -118.77 },
  "status" : "Just got back from saving the world. Again.",
  "hometown": "Malibu, USA"
  "email": "tony@stark.com",
  "website": "www.ironman.com",
  "birthdate": "1972-01-24"
}
```

409 Conflict

"Why do I need to
PUT the entire
profile just to
update location?

"My updates
keep failing
with a 409
Conflict! Help!"

```
PUT /profiles/ironman HTTP/1.1
Content-Type: application/json
{
    "status" : "Thinking about a disco Iron Man suit."
/* note JSON doesn't include ANY other fields */
}
```

204 No Content

PUT /profiles/ironman/status HTTP/1.1

"Finished saving the world. I need a drink."

204 No Content

The **PATCH** method requests that
a set of changes

described in the request entity be applied
to the **resource identified by the Request URI**.

The set of changes is represented in a format called
a "patch document" identified by a media type.

PATCH /file.txt HTTP/1.1

Host: www.example.com

Content-Type: application/example

If-Match: "e0023aa4e"

Content-Length: 100

[description of changes]

PATCH /file.txt HTTP/1.1

Host: www.api.herobook.local

Content-Type: application/example

If-Match: "e0023aa4e"

Content-Length: 100

[description of changes]

PATCH /profiles/ironman HTTP/1.1

Content-Type: application/x-unix-diff

11c11

<

"status": "Just got home from saving the world.",

>

"status": "World saved. I need a drink.",

200 OK

```
PATCH /profiles/ironman HTTP/1.1
Content-Type: application/json-patch+json
If-Match: "abc123"
[
  {
    "op": "replace",
    "path": "/status",
    "value": "Finished saving the world. I need a drink."
  }, {
    "op": "move",
    "from": "/friends/hulk",
    "path": "/friends[0]"
  }
]
```

200 OK

Module 5:

Walkthrough:

Introducing HTTP PATCH

Module 5:

Workshop exercise:

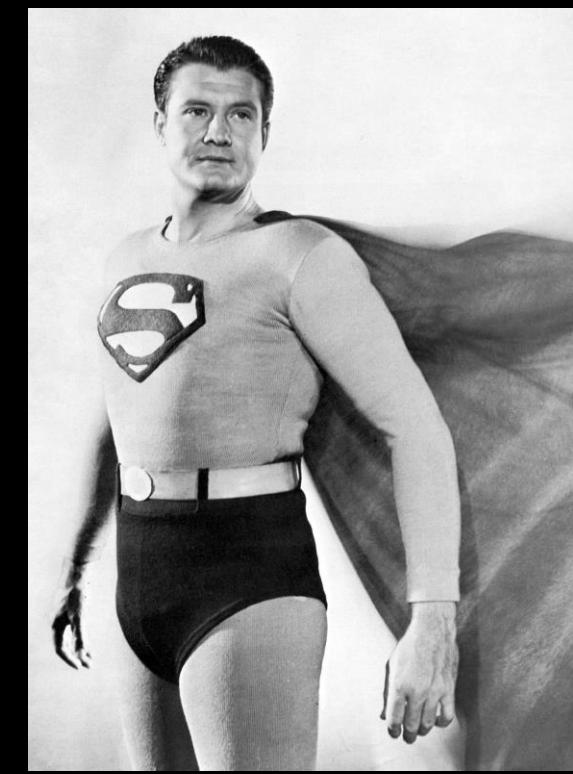
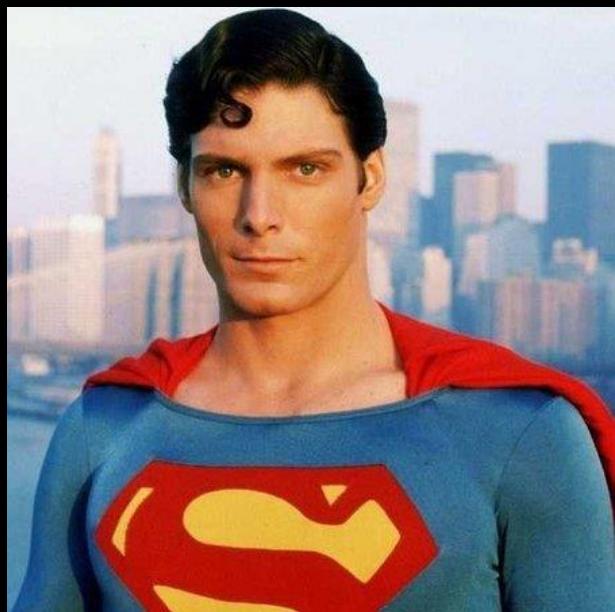
Introducing HTTP PATCH

Module 6:

Content Negotiation

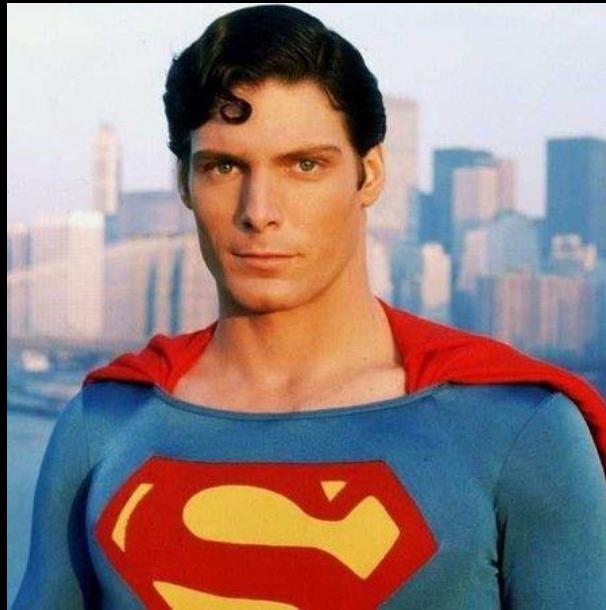
Resources and Representations







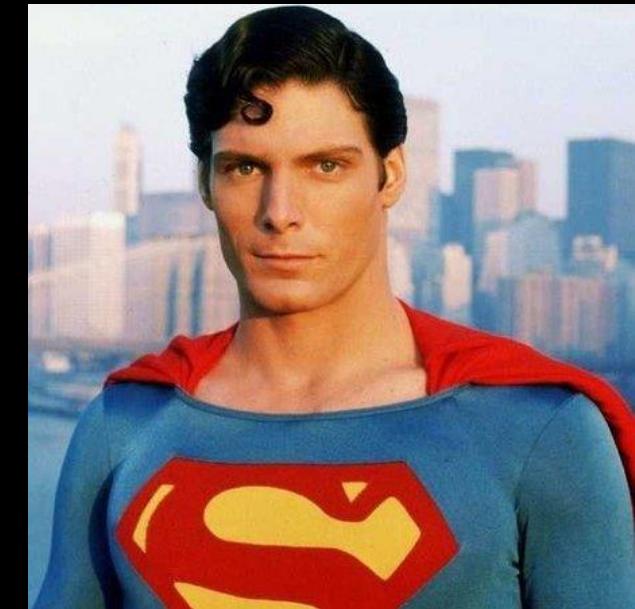
'concept'



Representation
(image/jpeg)



Resource
("this picture of Superman")



Representation
(image/png)

Module 6:

Walkthrough:

Resources and Representations

Module 6:

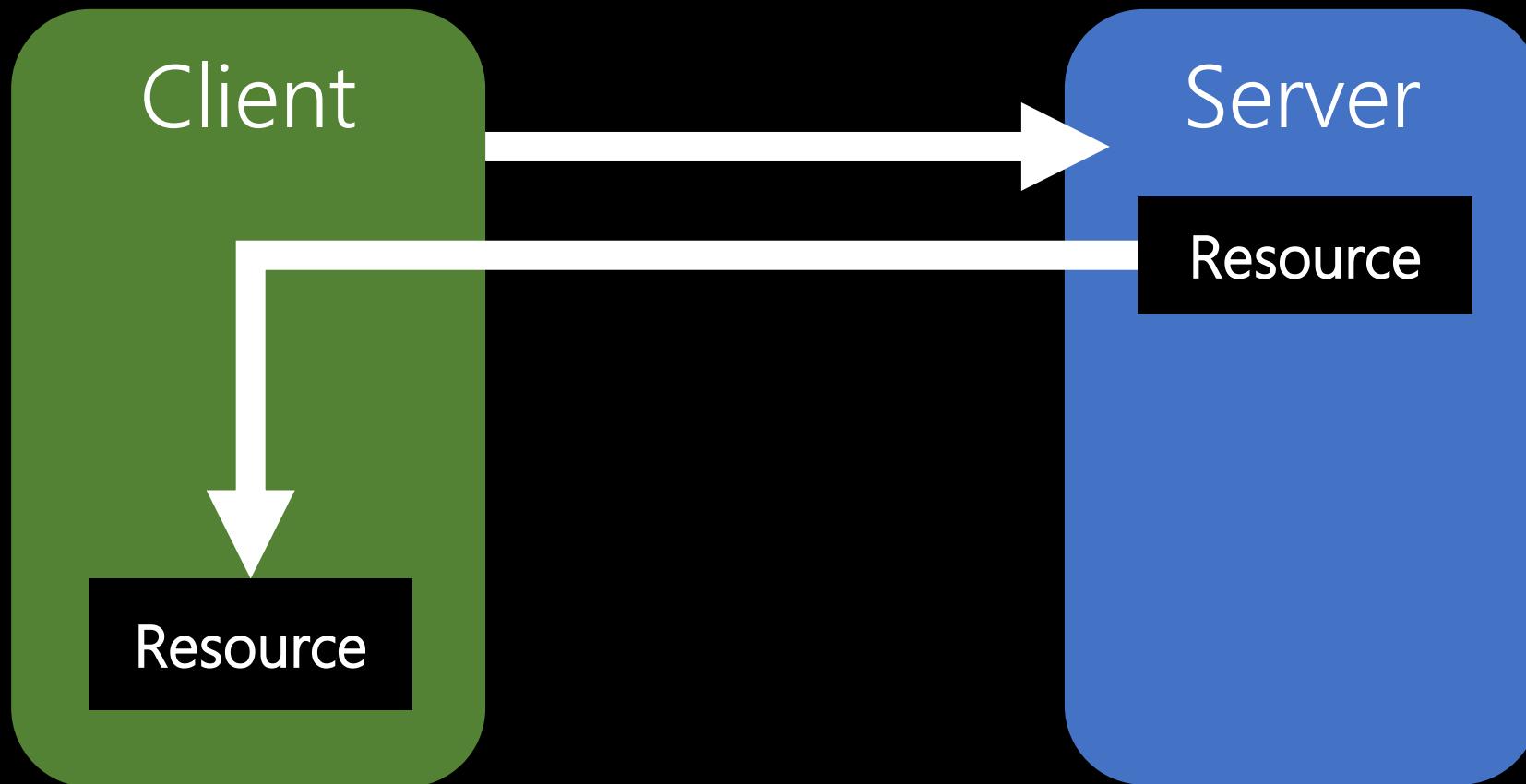
Coding Exercise:

Resources and Representations

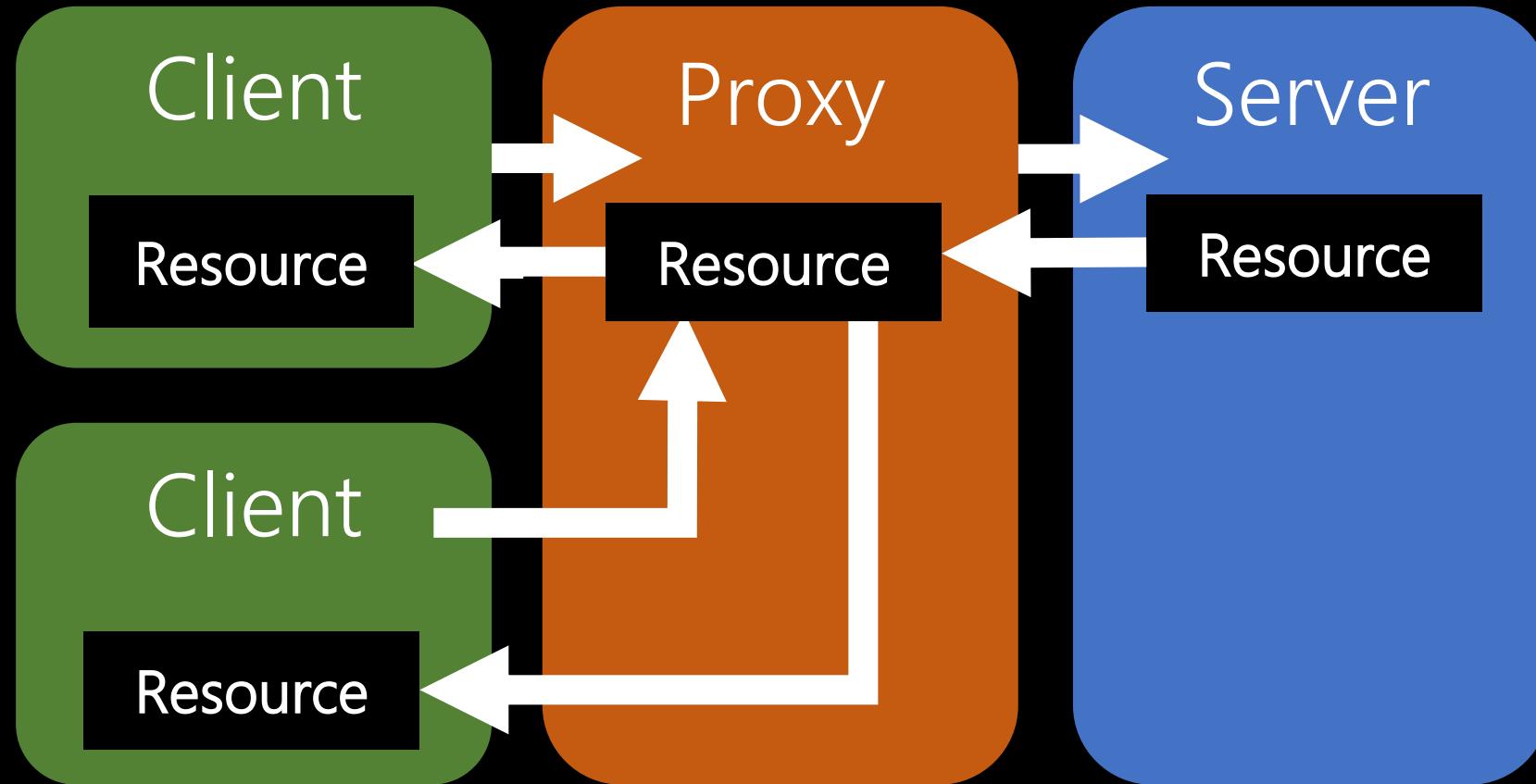
Module 7:

Scaling REST APIs: Caching and Layering

3. Cacheable



4. Layered System



```
GET /profiles/ironman HTTP/1.1  
Accept: application/json
```

200 OK

Content-Type: application/json

```
{  
  "name" : "Tony Stark",  
  "username": "ironman"  
}
```

GET /profiles/ironman HTTP/1.1
Accept: application/json

200 OK

Content-Type: application/json

Date: Tue, 22 May 2017 12:45:22 GMT

```
{  
  "name" : "Tony Stark",  
  "username": "ironman"  
}
```

GET /profiles/ironman HTTP/1.1

Accept: application/json

200 OK

Content-Type: application/json

Date: Tue, 22 May 2017 12:45:22 GMT

Expires: Tue, 22 May 2017 13:45:22 GMT

{

 "name" : "Tony Stark",

 "username": "ironman"

}

GET /profiles/ironman HTTP/1.1
Accept: application/json

GET /profiles/ironman HTTP/1.1

Accept: application/json

If-Modified-Since: Tue, 22 May 2017 12:45:22 GMT

GET /profiles/ironman HTTP/1.1

Accept: application/json

If-Modified-Since: Tue, 22 May 2017 12:45:22 GMT

304 Not Modified

Date: Tue, 22 May 2017 12:55:22 GMT

Expires: Tue, 22 May 2017 13:45:22 GMT

```
GET /profiles/ironman HTTP/1.1  
Accept: application/json
```

200 OK

Content-Type: application/json

```
{  
  "name" : "Tony Stark",  
  "username": "ironman"  
}
```

```
GET /profiles/ironman HTTP/1.1  
Accept: application/json
```

```
200 OK  
Content-Type: application/json  
Etag: "771320498712341098"  
{  
  "name" : "Tony Stark",  
  "username": "ironman"  
}
```

GET /profiles/ironman HTTP/1.1
Accept: application/json

GET /profiles/ironman HTTP/1.1

Accept: application/json

If-None-Match: "771320498712341098"

GET /profiles/ironman HTTP/1.1

Accept: application/json

If-None-Match: "771320498712341098"

304 Not Modified

PUT /profiles/ironman HTTP/1.1

Content-Type: application/json

{

 "name" : "Tony Stark",

 "username" : "ironman",

 "location" : "Tel Aviv, Israel"

}

```
PUT /profiles/ironman HTTP/1.1
Content-Type: application/json
If-None-Match: "167826786287782"

{
    "name" : "Tony Stark",
    "username" : "ironman",
    "location" : "Tel Aviv, Israel"
}
```

```
PUT /profiles/ironman HTTP/1.1
Content-Type: application/json
If-None-Match: "167826786287782"

{
    "name" : "Tony Stark",
    "username" : "ironman",
    "location" : "Tel Aviv, Israel"
}
```

409 Conflict

Module 7:

Walkthrough:

Caching and Layering

Module 7:

Workshop Exercise:

Caching and Layering

Module 8:

Security:

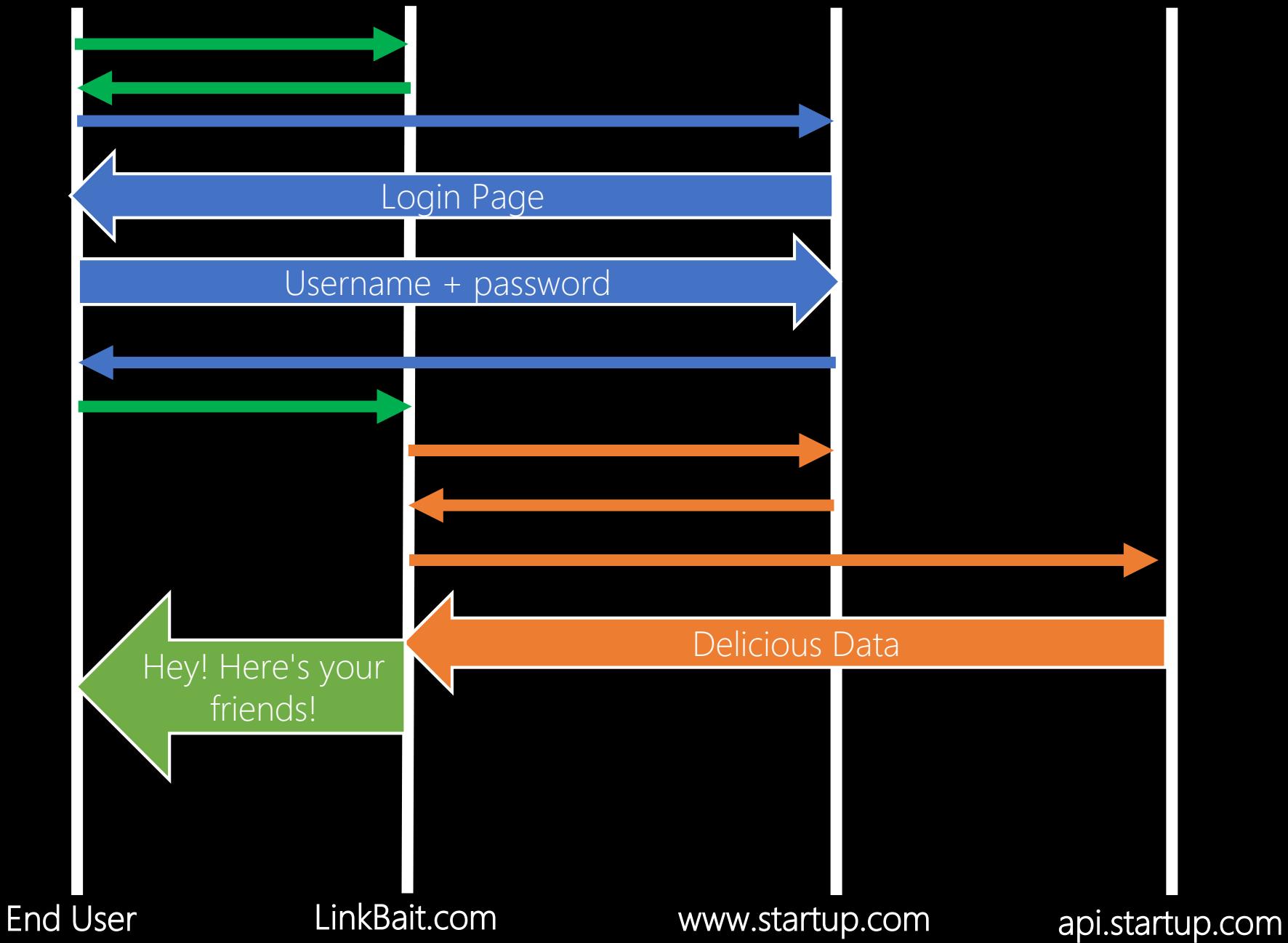
Authentication and Authorization

"We hired the
CFO's teenage son
to build us a mobile
app"

"We just found out
LinkBait.com is
asking for our
users' login details!"







Module 8:

Walkthrough:

Security and Authentication

Module 8:

Workshop:

Security and Authentication

Module 9:

Managing Changes:

API Versioning

"We need to expose forenames and surname separately"

"We're replacing hometown with an ISO3166 country code"

```
GET /profiles/ironman HTTP/1.1
```

```
200 OK
```

```
Content-Type: application/hal+json
```

```
{
```

```
  "name": { "firstname" : "Tony", "lastname" : "Stark" },
  "username": "ironman",
  "height": 192,
  "weight": 85,
  "location": { "lat": 34.02, "lon": -118.77 },
  "status": "Out saving the world. Again.",
  "hometown": {
    "city": "Malibu",
    "country": "US"
  },
  "email": "tony@stark.com",
  "website": "www.ironman.com",
  "birthdate": "1972-01-24",
  "last_modified": "2015-08-12T19:45:43.511Z"
}
```

"Hey! Your API broke
our app!"

"Our website just
stopped working"

"None of our
reports work!"

**"YOU BROKE
THE INTERNET!"**

oops.

The easiest thing to do is
...never break anything!

...but : "REST is software design on

the scale of

decades"

GET /v2/profiles/ironman

Accept: application/json

200 OK

{ ... }

GET /profiles/ironman

Accept: application/vnd.myapi.v2+json

200 OK

{ ... }

GET /profiles/ironman

Accept: application/json

X-MyApi-Version: 2

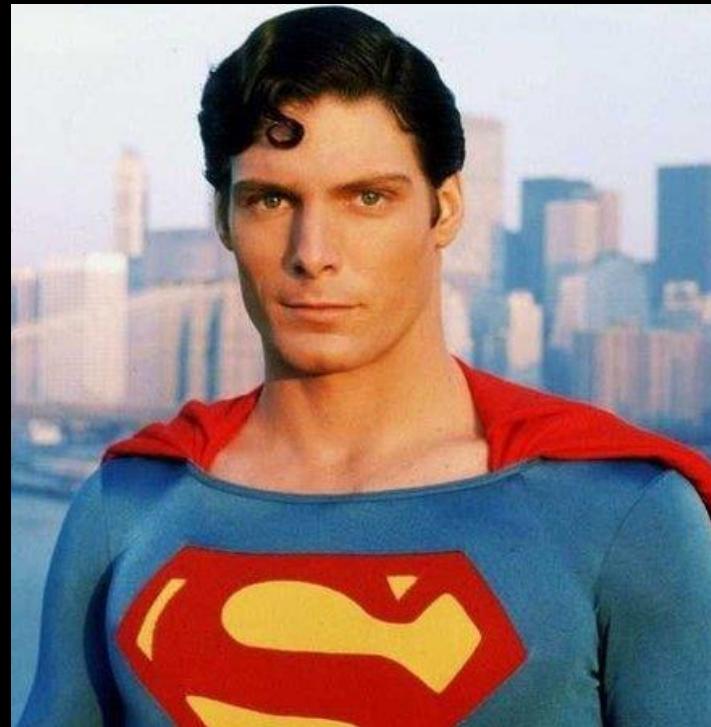
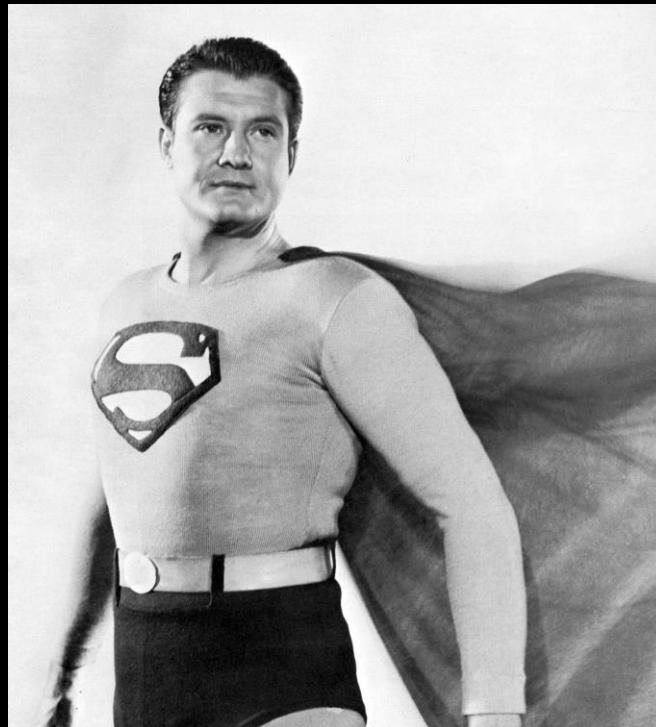
200 OK

{ ... }

"is version 2 of a profile
the *same resource*
as version 1?"

"Of course they are!
They represent the
same person!"

"Don't be daft.
They're completely
different."





"Of course they are! They represent the same person!"



GET /api/profiles

Accept: application/vnd.myapi.v2+json

"Don't be daft. They're completely different."



GET /api/v2/profiles

Accept: application/json

"Oh, forget ReST! Let's just use custom headers"



GET /api/profiles

X-MyApi-Version: 2

...and on the server?

Branch the
codebase
for separate
versions?

One
codebase,
many
versions?

```
[VersionedRoute("api/profiles/{name}", 2)]
[Route("api/v2/profiles/{name}")]
public Profile GetV2(string name) {

    /* write your program here! */

}
```

Module 9:

Walkthrough:

API Versioning

Module 9:

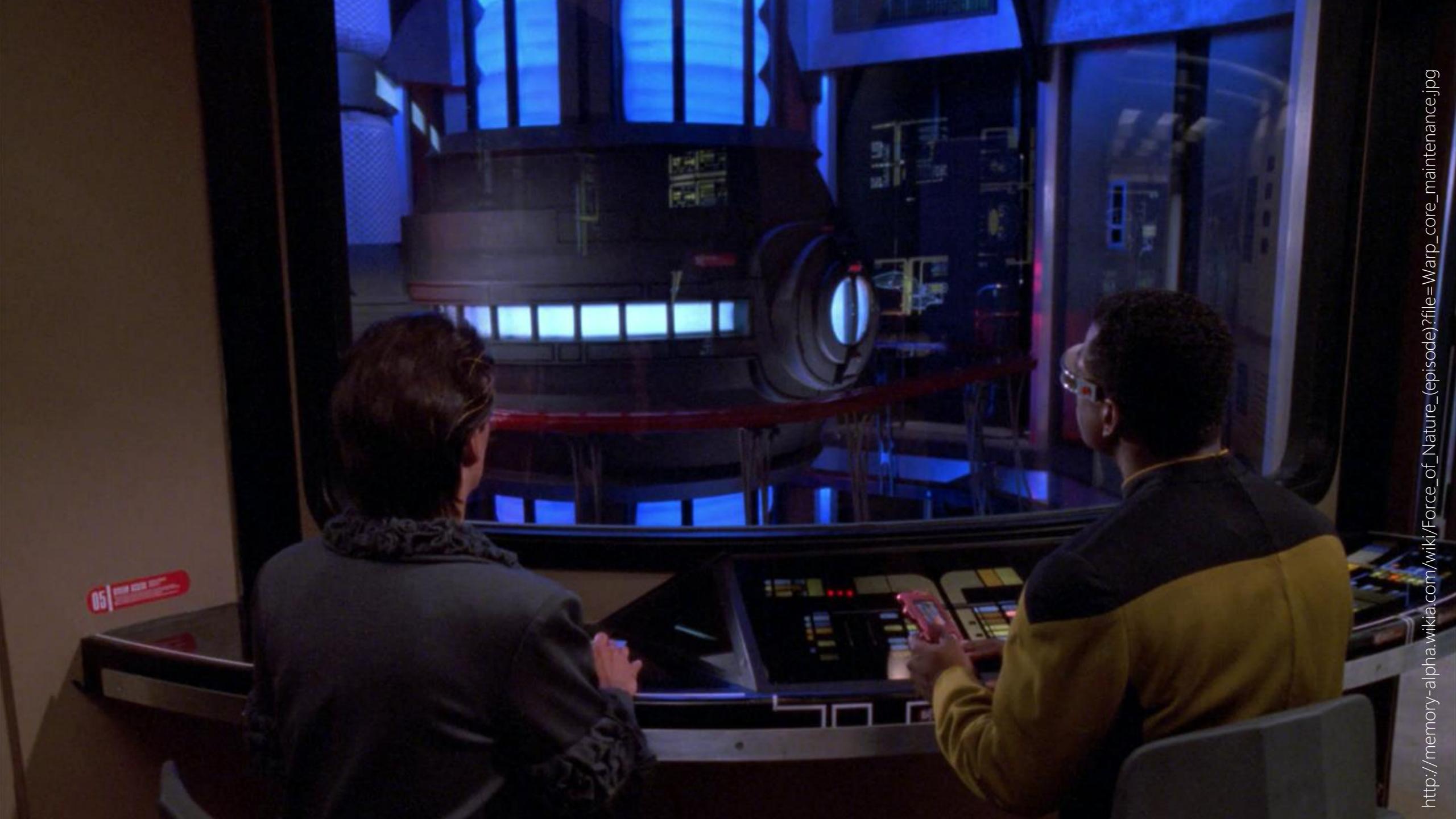
Workshop Exercise:

API Versioning

Module 10:

Testing and Monitoring

Hypermedia APIs





NANCY

Dashboard



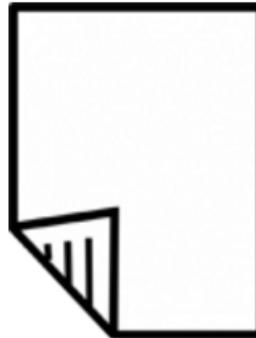
Information

Information about Nancy's current runtime configuration. A useful starting point for diagnosing a misbehaving site.



Interactive Diagnostics

Interactive diagnostics allow you to poke and prod at the guts of your Nancy site, from querying configuration, to clearing caches, through to executing parts of the pipeline and visualising the results.



Request Tracing

Provides tracing information about requests including request/response headers and a tracelog of each request.



Settings

Configure some of Nancy's runtime static configuration.



Overviews

Alerts

Analysis

Configuration



Administrator

Log out | Help | Give feedback

redgate

Analysis Graph ?

Time range:

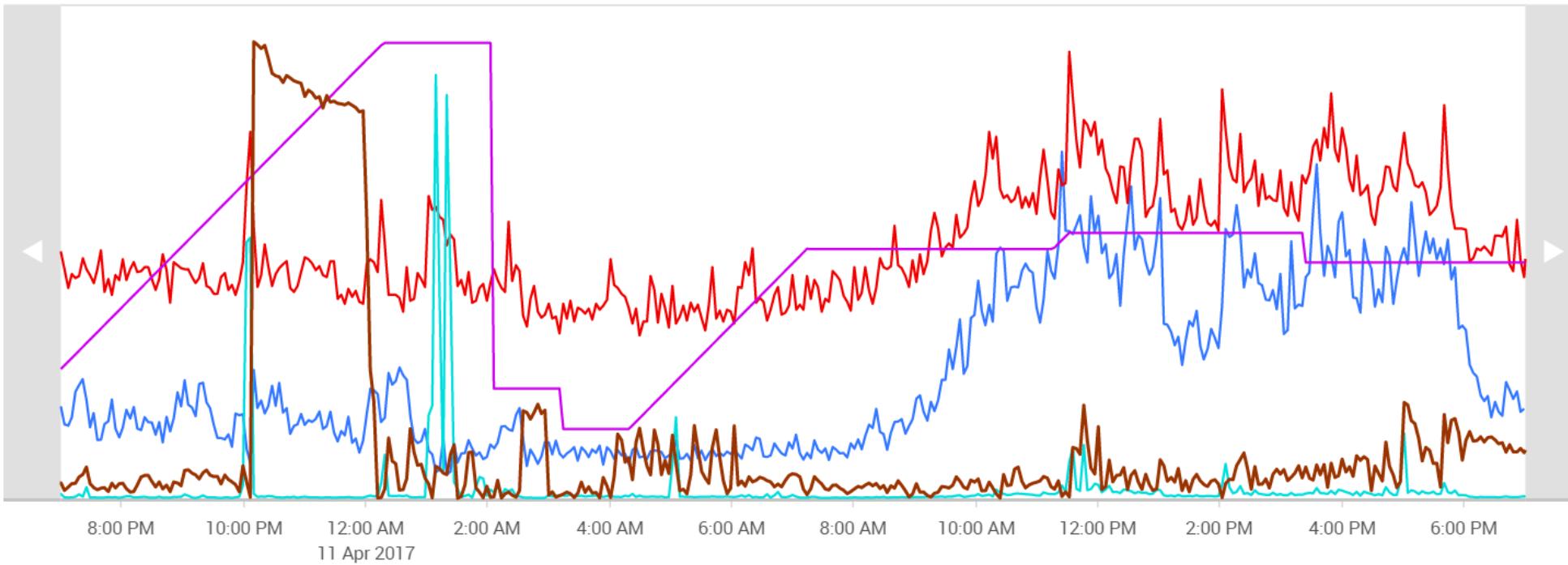
24 hours

Mon, Apr 10 19:00

Tue, Apr 11 19:00



COMPARE BASELINE

[Last 1h](#) [Last 6h](#) [Last 24h](#) [Last 7 days](#) [Today](#) [This week](#) | [Export \(csv\)](#)

- SQL Server: processor time [REDACTED]\(local)
- User connections [REDACTED]\(local)
- Buffer page life expectancy [REDACTED]\(local)
- Latch wait time [REDACTED]\(local)
- SQL Server: free memory [REDACTED]\(local)

► Show: SQL Server: processor time For: [REDACTED]\(local) ×

► And: User connections For: [REDACTED]\(local) ×

► And: Buffer page life expectancy For: (As above) ×

Description

Statistics

FREE MEMORY

The total amount of dyn...

Module 10:

Walkthrough:

Testing and Monitoring

Hypermedia APIs

Module 10:

Exercise:

Testing and Monitoring

Hypermedia APIs

DISCUSSION

API Strategy

Questions?

Questions?

Thank you!

www.github.com/dylanbeattie/ndc2017/

@dylanbeattie

dylan@dylanbeattie.net

Hypermedia support in JSON

JSON+Hypermedia Formats

- JSON-LD
 - Web Standard - <http://www.w3.org/TR/json-ld/>
- HYDRA
 - extends JSON-LD with available actions
- JSONAPI
 - <http://jsonapi.org/>
- Collection+JSON
 - <http://amundsen.com/media-types/collection/>
- SIREN
 - <https://github.com/kevinswiber/siren>
- HAL
 - http://stateless.co/hal_specification.html

JSON-LD

```
{  
  "@context": {  
    "name": "https://schema.org/name",  
    "alternateName": "https://schema.org/alternateName",  
    "image": {  
      "@id": "https://schema.org/image",  
      "@type": "@id"  
    }  
  },  
  "@id": "https://api.herobook.local/profiles/ironman",  
  "name": "Tony Stark",  
  "alternateName": "ironman",  
  "image":  
    "https://api.herobook.local/profiles/ironman/photos/avatar.pn  
g"
```

HYDRA

```
{  
  @id: "http://api.herobook.local/profiles",  
  @type: "http://www.w3.org/ns/hydra/core#Link",  
  description: "The collection of all superheroes",  
  label: "heroes",  
  range: "http://www.w3.org/ns/hydra/core#Collection",  
  supportedOperations: [  
    {  
      @id: "_:id1",  
      @type: "http://www.w3.org/ns/hydra/core#Operation",  
      label: "Creates a new Issue entity",  
      expects: "http://api.herobook.local/hydra/hero-api/vocab#Hero",  
      method: "POST",  
      returns: "http://api.herobook.local/hydra/hero-api/vocab#Hero",  
      statusCodes: [ {  
        @id: "_:id2",  
        description: "If the superhero was added successfully.",  
        code: 201  
      } ]  
    },  
    {  
      @id: "_:id3",  
      @type: "http://www.w3.org/ns/hydra/core#Operation",  
      label: "Retrieves all superhero entities",  
      expects: null,  
      method: "GET",  
      returns: "http://www.w3.org/ns/hydra/core#Collection",  
      statusCodes: [ ]  
    }  
  ]  
}
```

Collection+JSON

```
{ "collection" :  
{  
  "version" : "1.0",  
  "href" : "http://api.herobook.local/profiles/",  
  
  "links" : [  
    {"rel" : "feed", "href" : "http://api.herobook.local/blogs"},  
    {"rel" : "queries", "href" : "http://api.herobook.local/profiles/?queries"},  
  ],  
  
  "items" : [  
    {  
      "href" : "http://api.herobook.local/profiles/ironman",  
      "data" : [  
        {"name" : "username", "value" : "ironman", "prompt" : "Username"},  
        {"name" : "name", "value" : "Tony Stark", "prompt" : "Full Name"}  
      ],  
      "links" : [  
        {  
          "rel" : "friends",  
          "href" : "http://api.herobook.local/profiles/ironman/friends",  
          "prompt" : "Friends"  
        },  
        {  
          "rel" : "photos",  
          "href" : "http://api.herobook.local/profiles/ironman/photos",  
          "prompt" : "Photos",  
          "render" : "image"  
        }  
      ]  
    }  
  ]  
}
```

JSONAPI

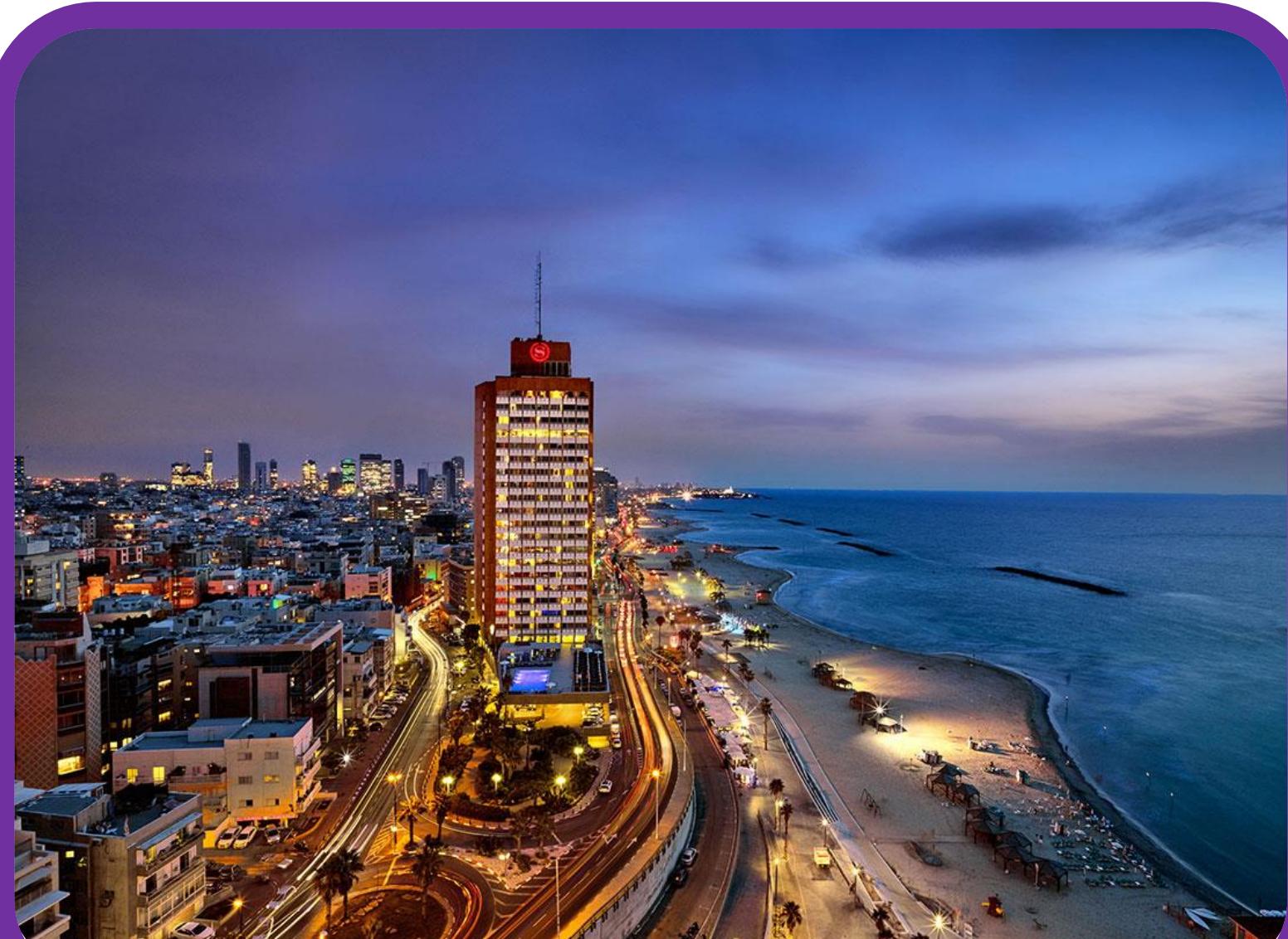
```
{  
  "links": {  
    "self": "http://api.herobook.local/posts",  
    "next": "http://api.herobook.local/posts?page[offset]=2",  
    "last": "http://api.herobook.local/posts?page[offset]=10"  
  },  
  "data": [{  
    "type": "posts",  
    "id": "1",  
    "attributes": { "title": "Iron Man wows crowds with new selfie stick!" },  
    "relationships": {  
      "author": {  
        "links": {  
          "self": "http://api.herobook.local/posts/1/relationships/author",  
          "related": "http://api.herobook.local/posts/1/author"  
        },  
        "data": { "type": "people", "id": "9" }  
      },  
      "comments": {  
        "links": {  
          "self": "http://api.herobook.local/posts/1/relationships/comments",  
          "related": "http://api.herobook.local/posts/1/comments"  
        },  
        "data": [  
          { "type": "comments", "id": "5" },  
          { "type": "comments", "id": "12" }  
        ]  
      }  
    },  
    "links": { "self": "http://api.herobook.local/posts/1" }  
  }]
```

SIREN

```
{  
  "class": [ "mission_list" ],  
  "properties": { "username": "ironman" },  
  "entities": [  
    {  
      "class": [ "missions", "collection" ],  
      "rel": [ "http://api.herobook.local/rels/missions" ],  
      "href": "http://api.herobook.local/profiles/ironman/missions"  
    },  
    {  
      "class": [ "hero", "customer" ],  
      "rel": [ "http://api.herobook.local/rels/hero" ],  
      "properties": { "username": "ironman", "name": "Tony Stark" },  
      "links": [ { "rel": [ "self" ], "href":  
        "http://api.herobook.local/profiles/ironman" } ]  
    }  
  ],  
  "actions": [  
    { "name": "assign-mission", "title": "Assign Mission",  
      "method": "POST", "href": "http://api.herobook.local/missions",  
      "type": "application/x-www-form-urlencoded",  
      "fields": [  
        { "name": "hero", "type": "hidden", "value": "ironman" },  
        { "name": "latitude", "type": "number" },  
        { "name": "longitude", "type": "number" },  
        { "name": "time", "type": "datetime" }  
        { "name": "objective", "type": "text" }  
      ]  
    }  
  ]  
}
```

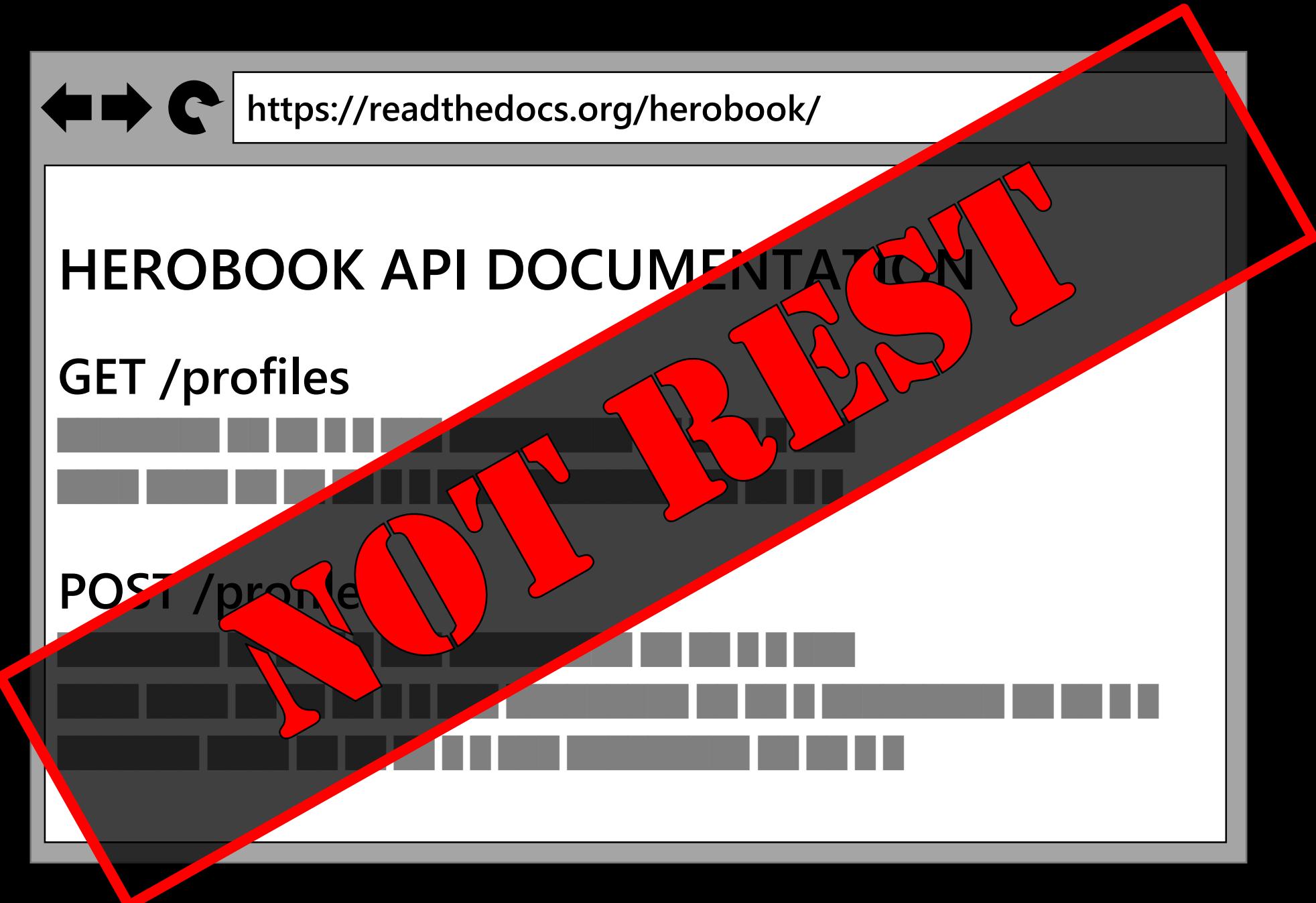
HYPERMEDIA

SELA DEVELOPER PRACTISE



"Hey- where's your
API documentation?"

"How do I register a
user on your new
system?"



GET / HTTP/1.1

200 OK

Content-Type: application/json

{

"message": "Welcome to Herobook!"

}

GET / HTTP/1.1

200 OK

Content-Type: application/hal+json

{

"message": "Welcome to Herobook!"

}

GET / HTTP/1.1

200 OK

Content-Type: application/hal+json

```
{  
  "_links": {  
    "profiles" : {  
      "href" : "/profiles"  
    }  
  },  
  "message": "Welcome to Herobook!"  
}
```

FATAL

ERROR

WARN

INFO

DEBUG

FATAL

- Application is completely unresponsive
- Multiple users affected with immediate effect
- Immediate attention. Stop what you're doing
and look into it.

ERROR & WARN

- They *will* happen.
- **ERROR** – one person noticed, maybe?
- **WARN** – nobody noticed but it's still odd.
- Individual messages probably aren't useful



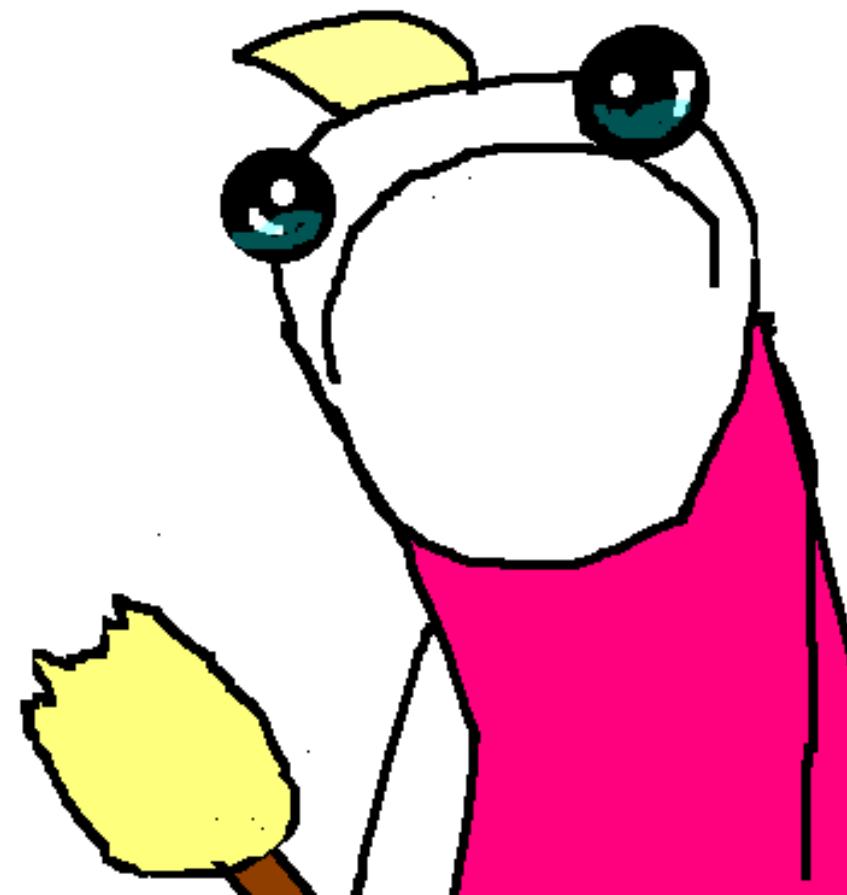
INFO

- Everything's fine
- Startup, shutdown, cache recycle
- In a perfect world, INFO is all you'll ever see.

LOG ALL THE
THINGS!



wait... all the things?





3:17 AM