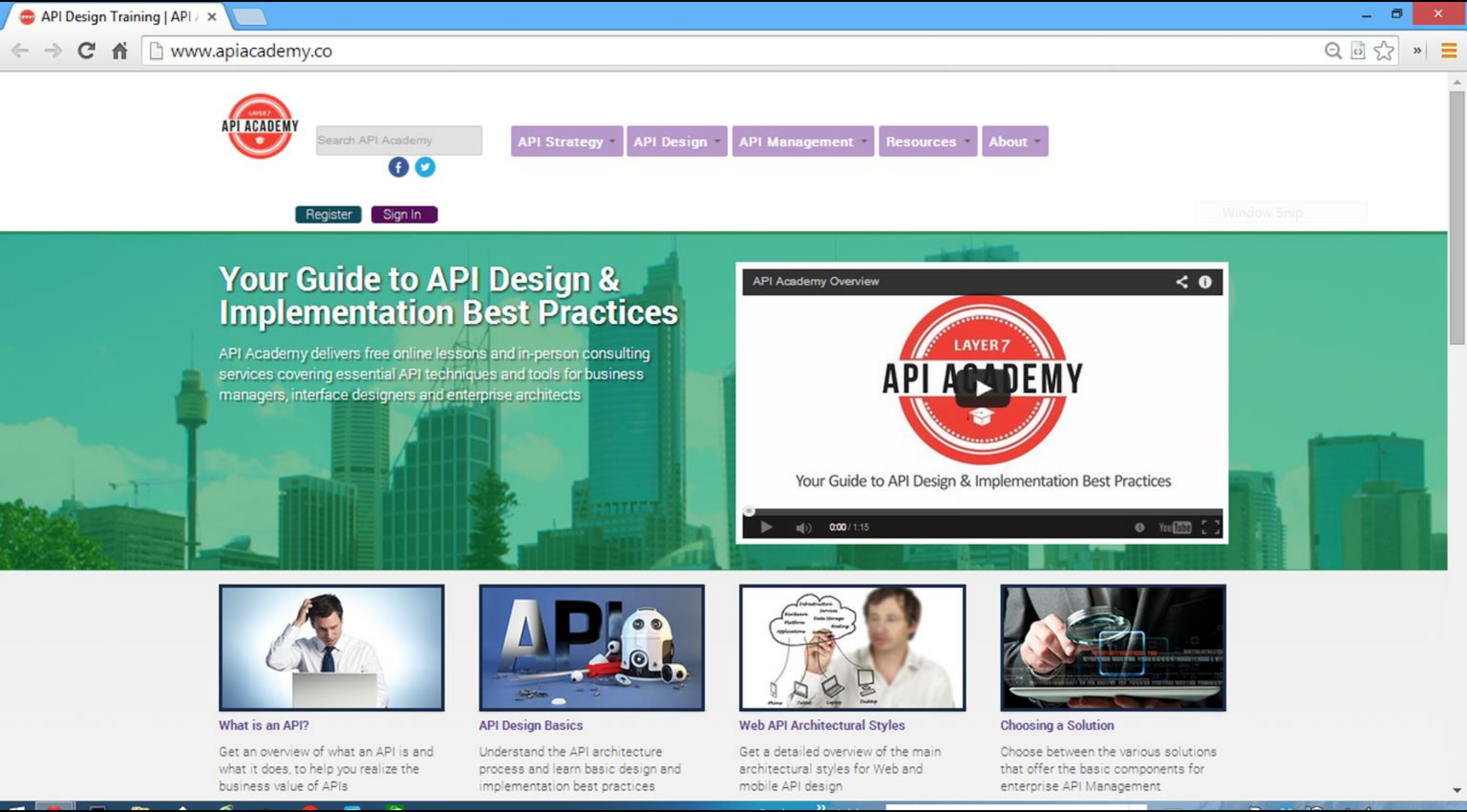
Learning Client Hypermedia from the Ground Up

Mike Amundsen,
API Academy / CA
@mamund

Introduction

PgUp
Ctrl
Mike Amundsen
@mamund



API Design Training | API

www.apiacademy.co

Search API Academy

API Strategy | API Design | API Management | Resources | About

Register | Sign In

Window Snip

Your Guide to API Design & Implementation Best Practices

API Academy delivers free online lessons and in-person consulting services covering essential API techniques and tools for business managers, interface designers and enterprise architects

What is an API?

Get an overview of what an API is and what it does, to help you realize the business value of APIs

API Design Basics

Understand the API architecture process and learn basic design and implementation best practices

Web API Architectural Styles

Get a detailed overview of the main architectural styles for Web and mobile API design

Choosing a Solution

Choose between the various solutions that offer the basic components for enterprise API Management

Goals

- Understand Hypermedia for applications
- Refactor a “plain/json” app into Hypermedia app
- Learn how to add features w/o versioning your app
- Identify the diff. between hypermedia Links and Templates
- Review two existing Hypermedia formats
- See how Hypermedia and Microservices work together

Show that you can increase client adaptability w/o increasing the code complexity/size

BROADCAST
INTERRUPTED

Hypermedia

Hypermedia?

"When I say hypertext, I mean the simultaneous presentation of information and controls such that the information becomes the affordance through which the user (or automaton) obtains choices and selects actions."

- Roy Fielding, 2009

"When I say hypertext, I mean the simultaneous presentation of information and controls such that ***the information becomes the affordance*** *through which the user (or automaton) obtains choices and selects actions."*

- Roy Fielding, 2009

*"When I say hypertext, I mean the simultaneous presentation of information and controls such that **the information becomes the affordance** through which the user (or automaton) **obtains choices and selects actions.**"*

- Roy Fielding, 2009

Stuart Landsborough's
PUZZLING WORLD!
Wanaka, New Zealand

Can you could just show me some examples?

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

```
<img src="http://www.example.org/images/logo" title="company logo" />
```

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

```
<img src="http://www.example.org/images/logo" title="company logo" />
```

```
<form method="get">
  <label>Search term:</label>
  <input name="query" type="text" value="" />
  <input type="submit" />
</form>
```

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

```
<img src="http://www.example.org/images/logo" title="company logo" />
```

```
<form method="get">
  <label>Search term:</label>
  <input name="query" type="text" value="" />
  <input type="submit" />
</form>
```

```
<form method="post" action="http://www.example.org/my-keywords"/>
  <label>Keywords:</label>
  <input name="keywords" type="text" value="" />
  <input type="submit" />
</form>
```

NOT Hypermedia Examples

```
"http://www.example.org/search"
```

```
"http://www.example.org/images/logo"
```

```
"http://www.example.org/my-keywords"
```

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

```
"http://www.example.org/images/logo"
```

```
"http://www.example.org/my-keywords"
```

Hypermedia Examples

```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

```
<img src="http://www.example.org/images/logo" title="company logo" />
```

```
"http://www.example.org/my-keywords"
```

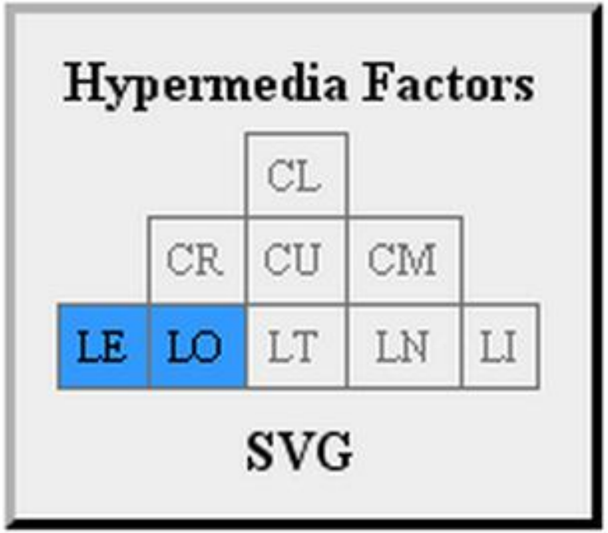
Hypermedia Examples

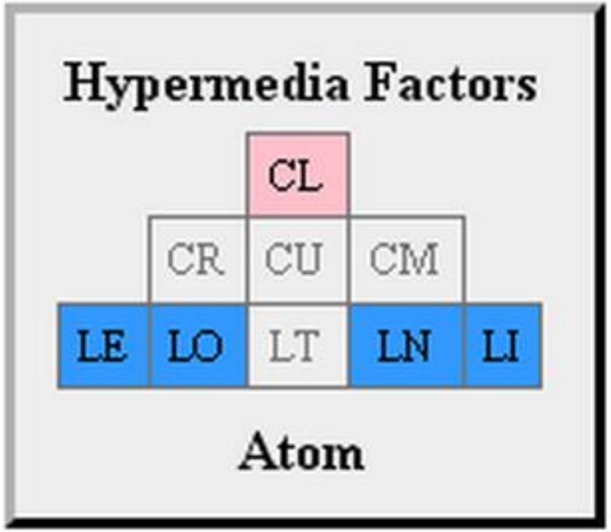
```
<a href="http://www.example.org/search" title="view search page">Search</a>
```

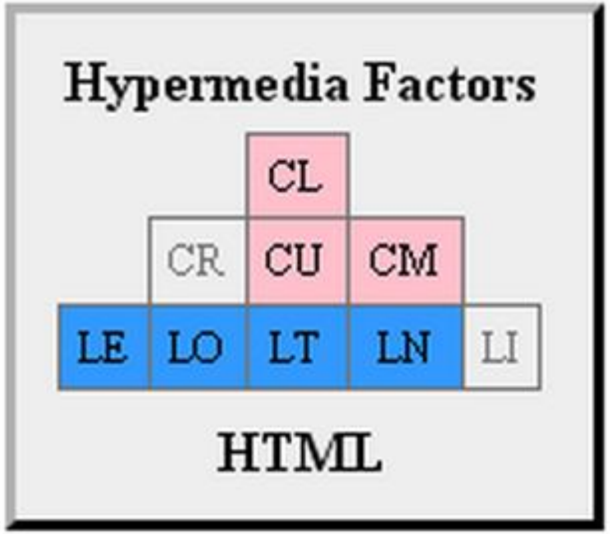
```
<img src="http://www.example.org/images/logo" title="company logo" />
```

```
<form method="get">
  <label>Search term:</label>
  <input name="query" type="text" value="" />
  <input type="submit" />
</form>
```

```
<form method="post" action="http://www.example.org/my-keywords"/>
  <label>Keywords:</label>
  <input name="keywords" type="text" value="" />
  <input type="submit" />
</form>
```





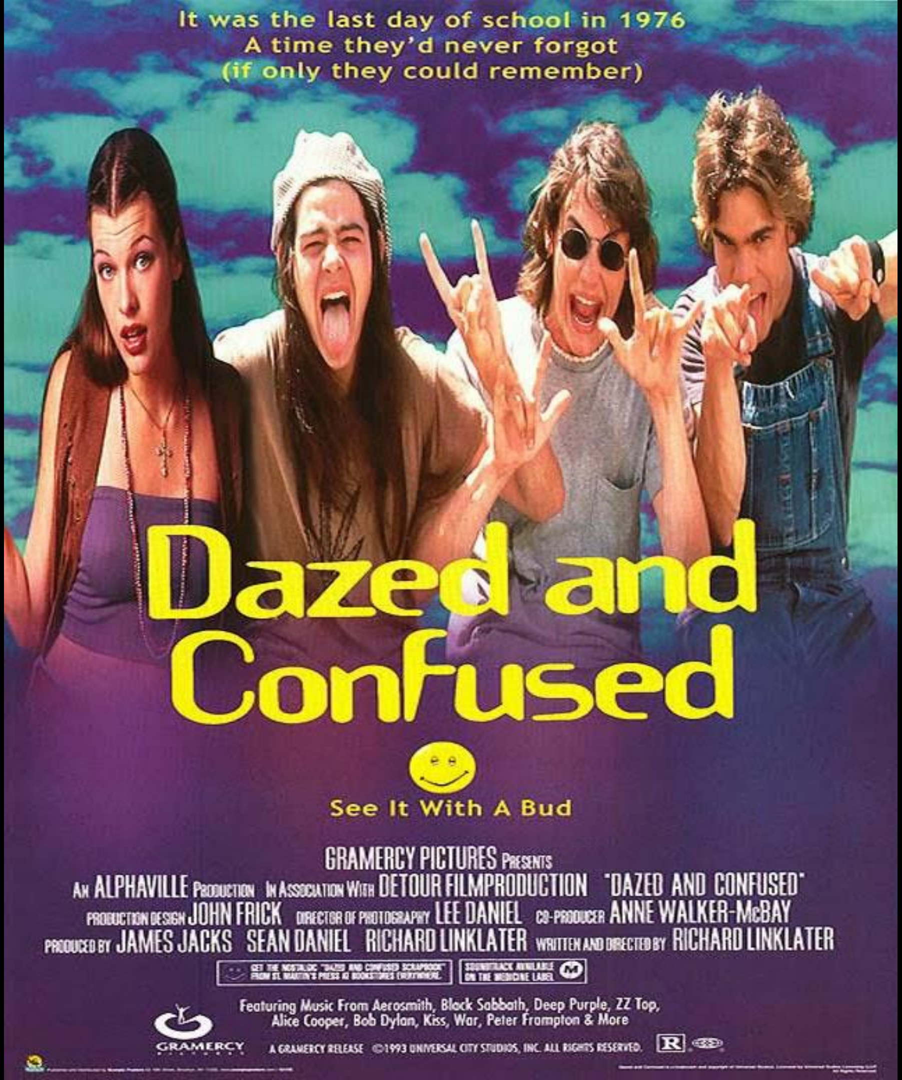


H-Factors and Hypermedia Types
http://amundsen.com/hypermedia/

Hypermedia controls provide inline descriptions of your application's actions.

Yeah, so why would you do that?

"Hypermedia as the engine of application state."


It was the last day of school in 1976
A time they'd never forgot
(if only they could remember)
Dazed and Confused
See It With A Bud
GRAMERCY PICTURES PRESENTS
AN ALPHAVILLE PRODUCTION IN ASSOCIATION WITH DETOUR FILMPRODUCTION "DAZED AND CONFUSED"
PRODUCTION DESIGN JOHN FRICK DIRECTOR OF PHOTOGRAPHY LEE DANIEL CO-PRODUCER ANNE WALKER-McBAY
PRODUCED BY JAMES JACKS SEAN DANIEL RICHARD LINKLATER WRITTEN AND DIRECTED BY RICHARD LINKLATER
GET THE NOSTALGIC "DAZED AND CONFUSED SCRAPBOOK" FROM ST. MARTIN'S PRESS AT BOOKSTORES EVERYWHERE.
SOUNDTRACK AVAILABLE ON THE MEDICINE LABEL
Featuring Music From Aerosmith, Black Sabbath, Deep Purple, ZZ Top, Alice Cooper, Bob Dylan, Kiss, War, Peter Frampton & More
GRAMERCY
A GRAMERCY RELEASE ©1993 UNIVERSAL CITY STUDIOS, INC. ALL RIGHTS RESERVED.
R

"Hypermedia as the engine of application state."

HATEOAS

"Hypermedia as the engine of application state."

~~HATEOAS~~

"Hypermedia as the engine of application state."

Hypermedia Constraint

Wait, Hypermedia ***Constraint****?*

In software architecture models, constraints are selected to promote system properties.

As part of the ***Uniform Interface*** *constraint,*
Fielding selected the Hypermedia
to promote:

*As part of the **Uniform Interface** constraint,*
Fielding selected the Hypermedia
to promote:

Evolvability

Evolvability

"Evolvability represents the degree to which a component implementation can be changed without negatively impacting other components."

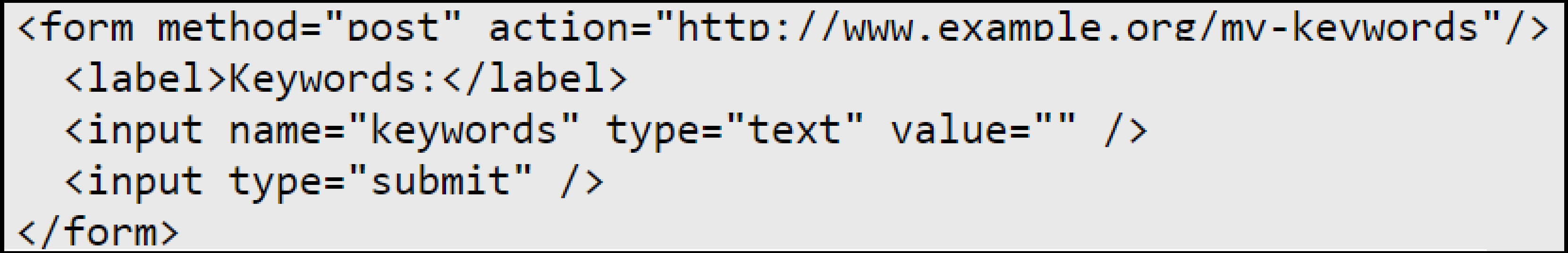
Fielding, Section 2.3.4.1

"Hypermedia as the engine of application state."

"Use Hypermedia to execute application actions."

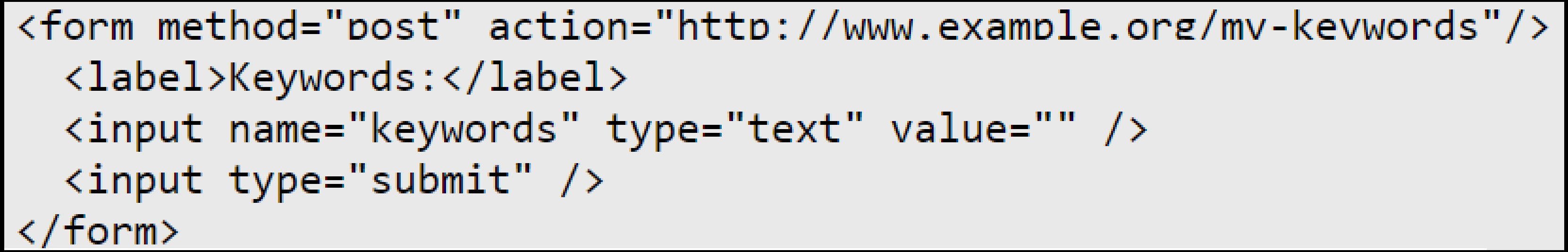
Use Hypermedia

Use Hypermedia



```
<form method="post" action="http://www.example.org/my-keywords"/>
  <label>Keywords:</label>
  <input name="keywords" type="text" value="" />
  <input type="submit" />
</form>
```

Use Hypermedia



```
<form method="post" action="http://www.example.org/my-keywords"/>
  <label>Keywords:</label>
  <input name="keywords" type="text" value="" />
  <input type="submit" />
</form>
```

to execute application actions.

AND NOW BACK TO OUR REGULARLY SCHEDULED PROGRAMMING

A Client w/o Hypermedia

Disclaimer…

"No frameworks or libraries where harmed in the making of these apps."

These example are NOT:

- Production-Ready
- Robust
- Pretty

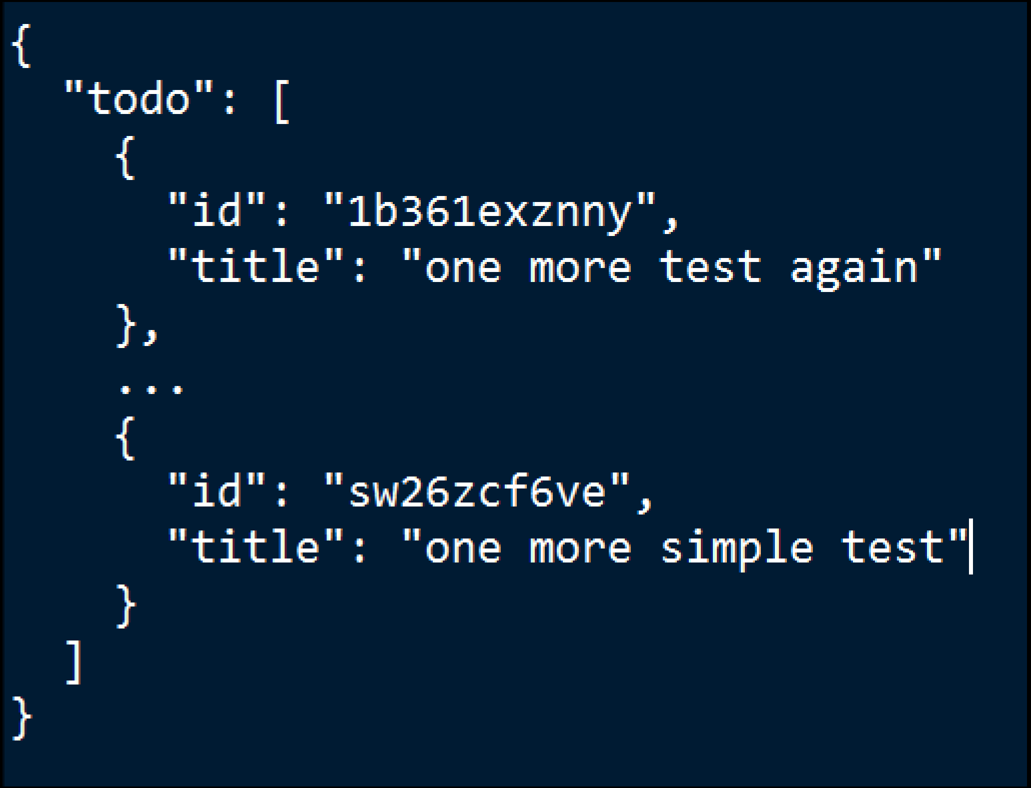
TL;DNR…

"I don't have time right now, can I just download the code?"

https://github.com/apiacademy/ndcoslo2015

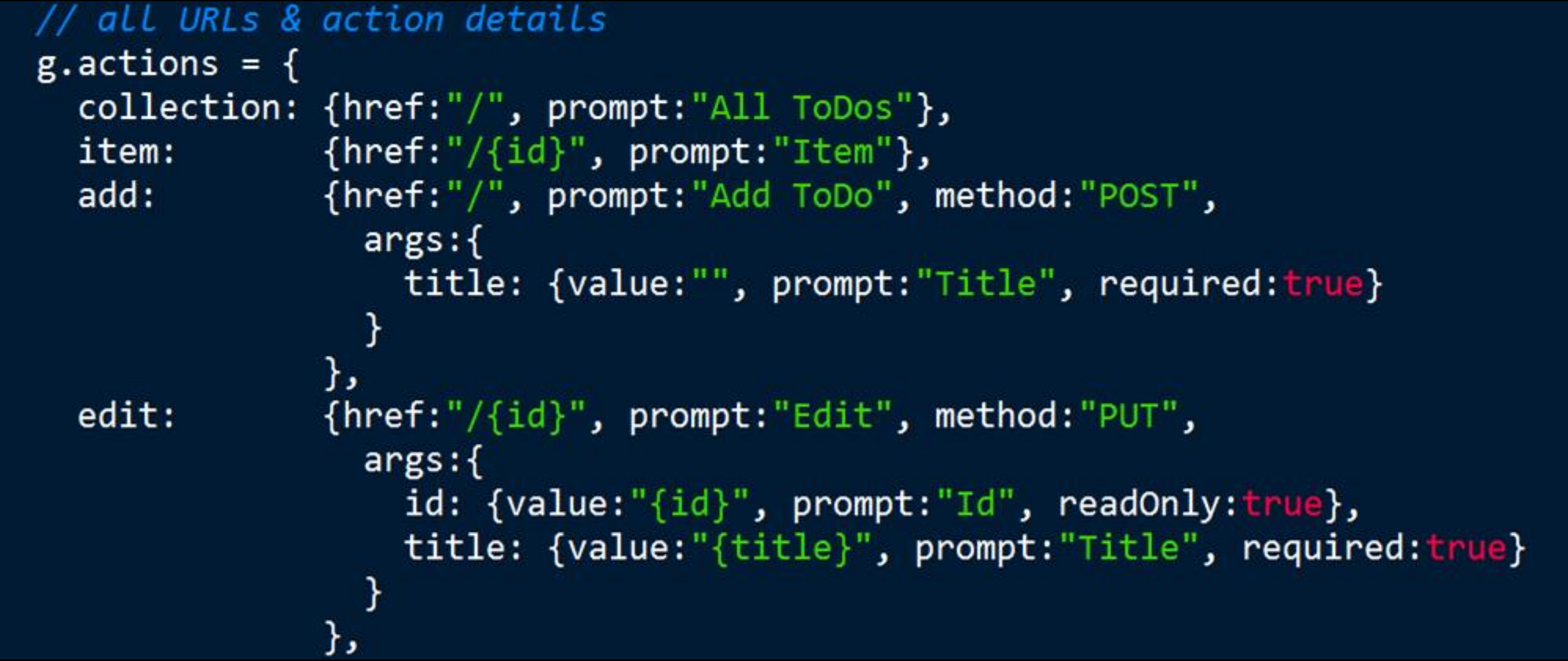
Client w/o Hypermedia

- Get JSON objects/collections from server
- Put Read/Write semantics in the client Code



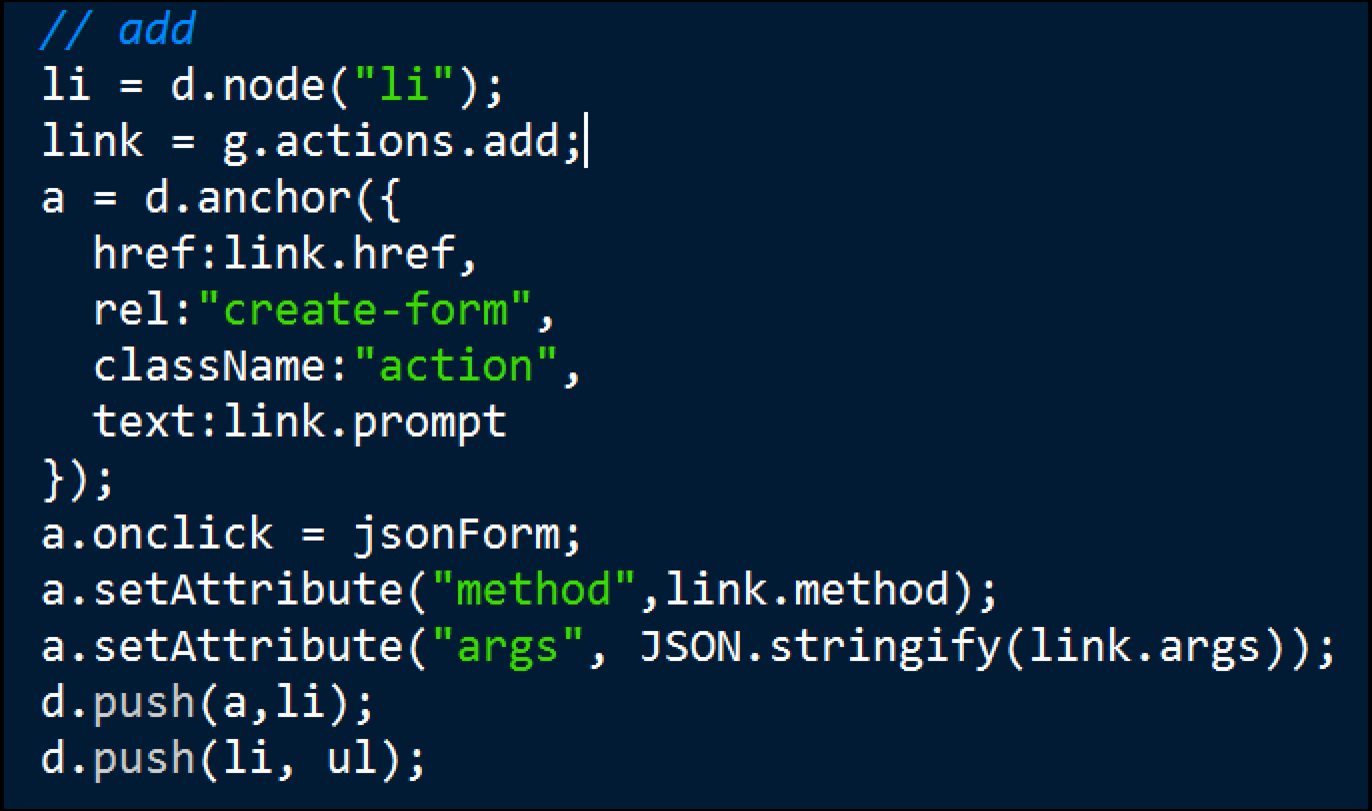
```
{
  "todo": [
    {
      "id": "1b361exznny",
      "title": "one more test again"
    },
    ...
    {
      "id": "sw26zcf6ve",
      "title": "one more simple test"
    }
  ]
}
```

JSON objects/collections from server



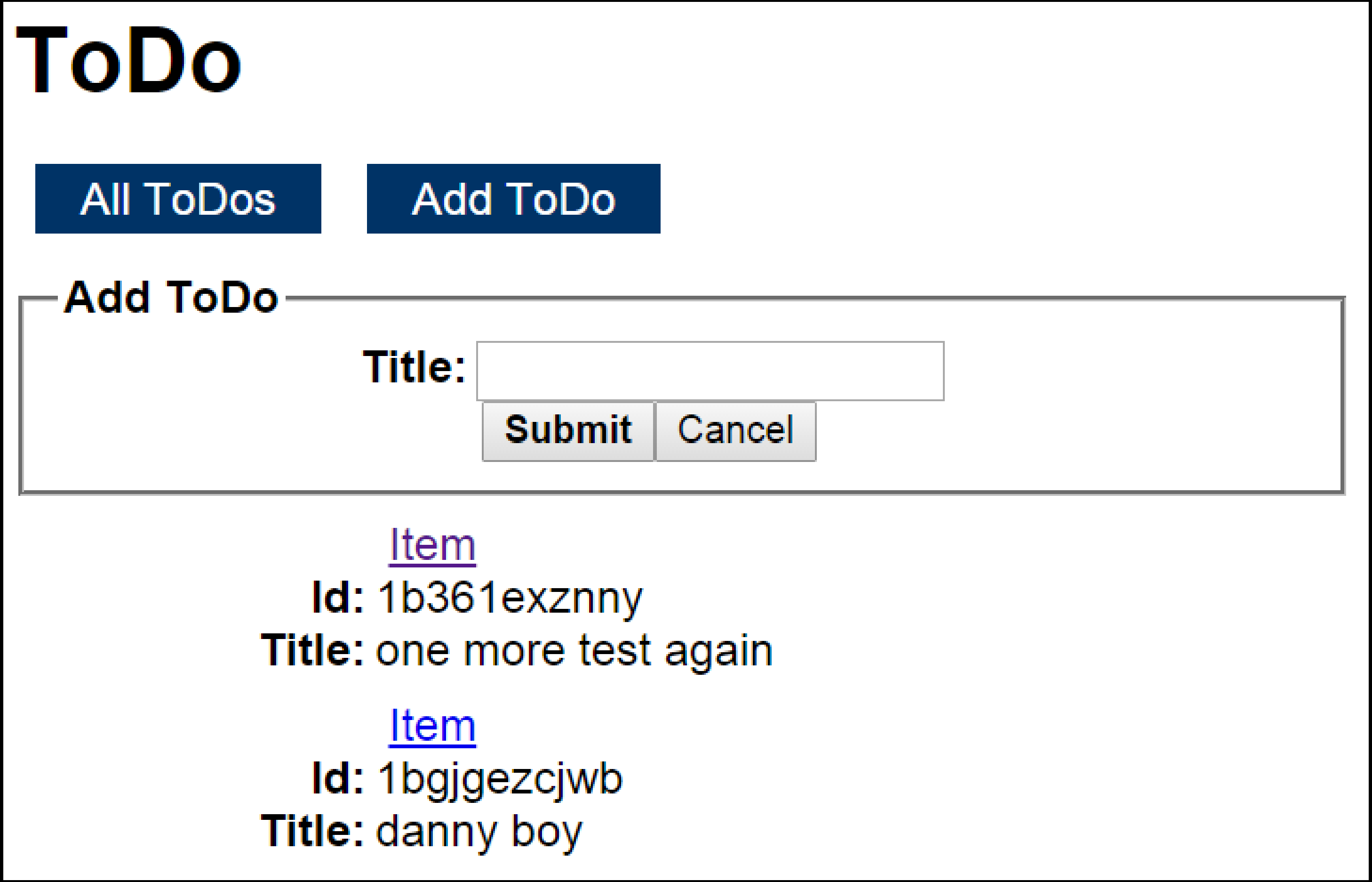
```
// all URLs & action details
g.actions = {
  collection: {href:"/", prompt:"All ToDos"},
  item:       {href:"/{id}", prompt:"Item"},
  add:        {href:"/", prompt:"Add ToDo", method:"POST",
                 args:{
                   title: {value:"", prompt:"Title", required:true}
                 }
              },
  edit:       {href:"/{id}", prompt:"Edit", method:"PUT",
                 args:{
                   id: {value:"{id}", prompt:"Id", readOnly:true},
                   title: {value:"{title}", prompt:"Title", required:true}
                 }
              },
```

Put Read/Write Semantics in the Client Code



```
// add
li = d.node("li");
link = g.actions.add;
a = d.anchor({
  href:link.href,
  rel:"create-form",
  className:"action",
  text:link.prompt
});
a.onclick = jsonForm;
a.setAttribute("method",link.method);
a.setAttribute("args", JSON.stringify(link.args));
d.push(a,li);
d.push(li, ul);
```

Read/Write Semantics in the Client Code



ToDo

Add ToDo

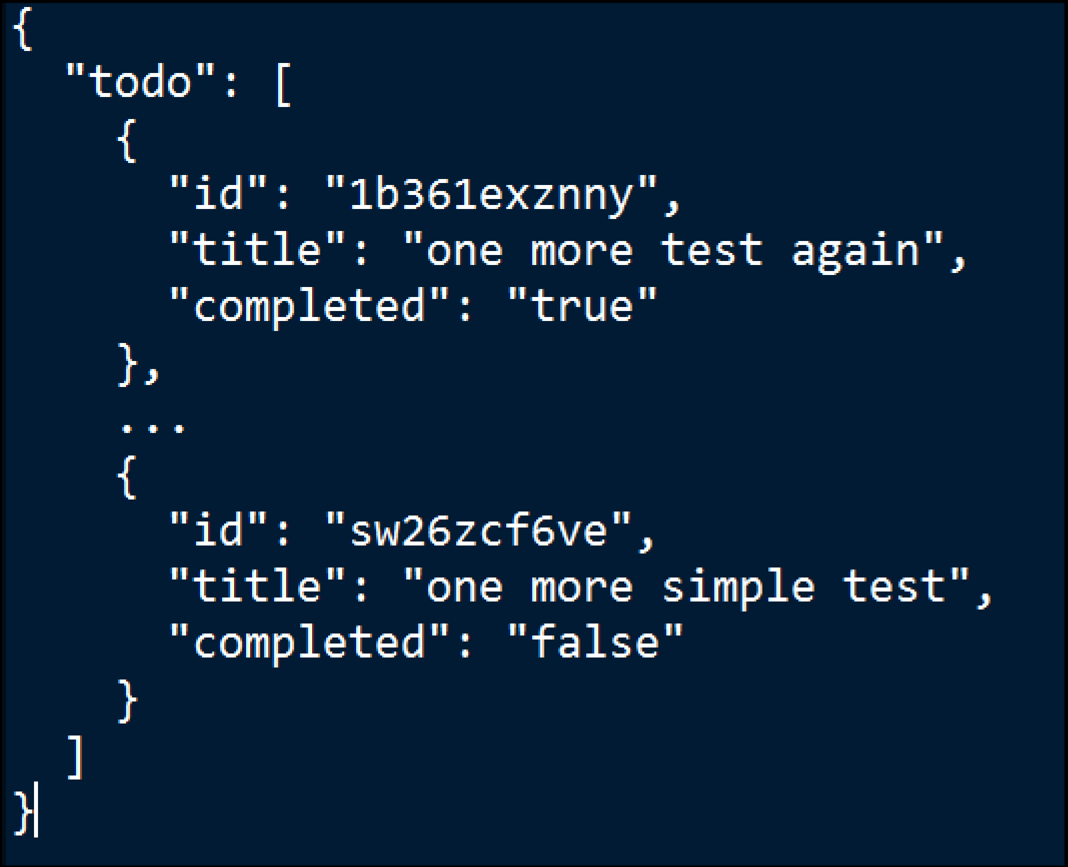
Title:

Item
Id: 1b361exznny
Title: one more test again

Item
Id: 1bgjgezcjwb
Title: danny boy

Live Demo of Client w/o Hypermedia

Then you want to add some features...



```
{
  "todo": [
    {
      "id": "1b361exznny",
      "title": "one more test again",
      "completed": "true"
    },
    ...
    {
      "id": "sw26zcf6ve",
      "title": "one more simple test",
      "completed": "false"
    }
  ]
}
```

Let's add a new field ("completed") and some filters

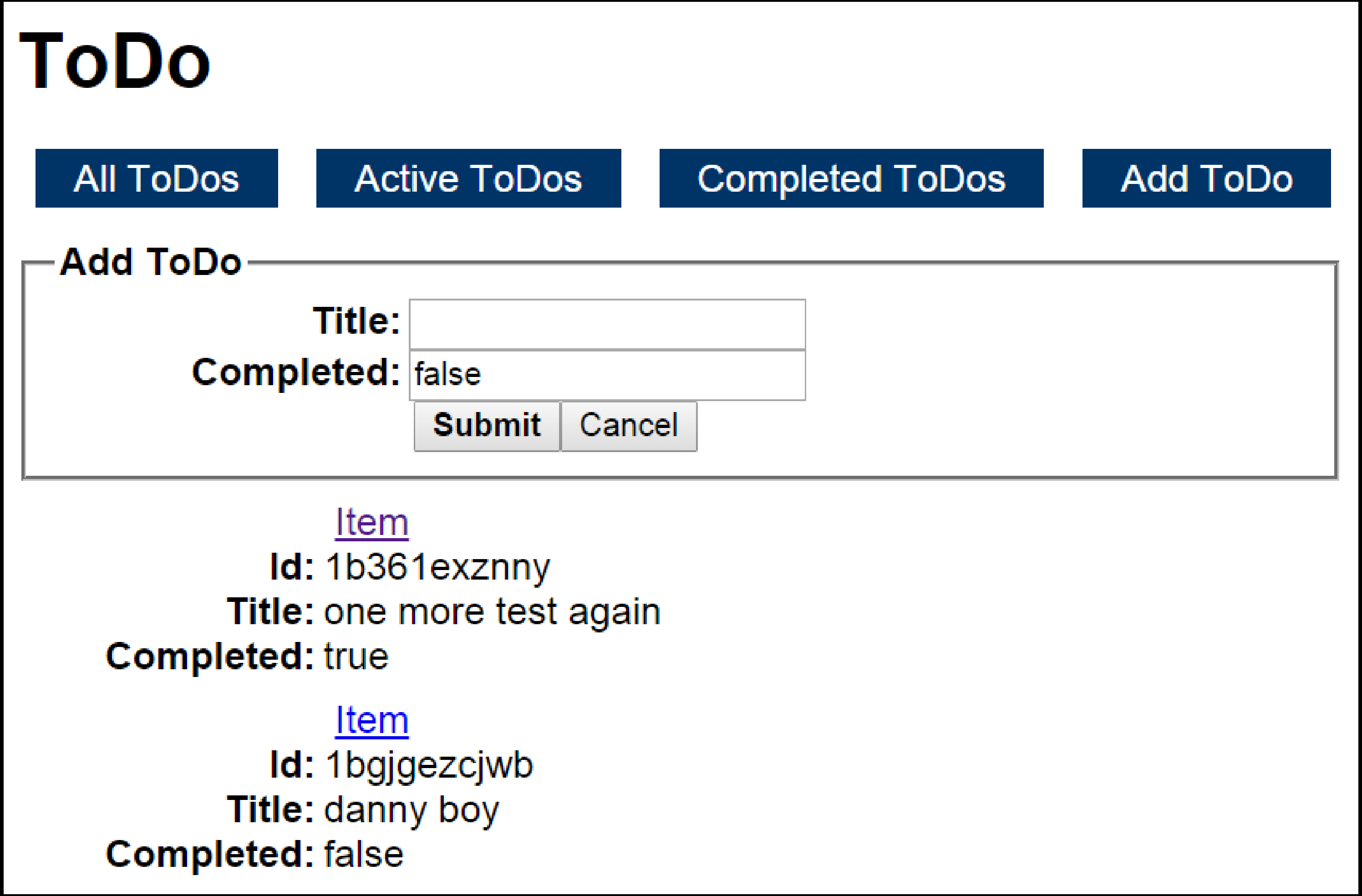
But the installed app IGNORES the new stuff!

Oh, right. Now I need a new VERSION!

```
// the only fields to process
g.fields = ["id","title","completed"];

// all URLs & action details
g.actions = {
  collection: {href:"/", prompt:"All ToDos"},
  active:     {href:"/?completed=false", prompt:"Active ToDos"},
  completed:  {href:"/?completed=true", prompt:"Completed ToDos"},
  item:       {href:"/{id}", prompt:"Item"},
  add:        {href:"/", prompt:"Add ToDo", method:"POST",
                args:{
                  title: {value:"", prompt:"Title", required:true},
                  completed: {value:"false", prompt:"Completed", pattern:"^(true|false)$"}
                }
              },
  edit:       {href:"/{id}", prompt:"Edit", method:"PUT",
                args:{
                  id: {value:"{id}", prompt:"Id", readOnly:true},
                  title: {value:"{title}", prompt:"Title", required:true},
                  completed: {value:"{completed}", prompt:"Completed", pattern:"^(true|false)$"}
                }
              },
```

NEW Read/Write Semantics in the Client Code

OK, V2 code/test is done, RE-DEPLOY!



Add ToDo

Title:

Completed: false

Id: 1b361exznny
Title: one more test again
Completed: true

Id: 1bgjgezcjwb
Title: danny boy
Completed: false

Live Demo of VERSIONED Client w/o Hypermedia

Well, that's not too bad in the browser.

Well, that's not too bad in the browser.

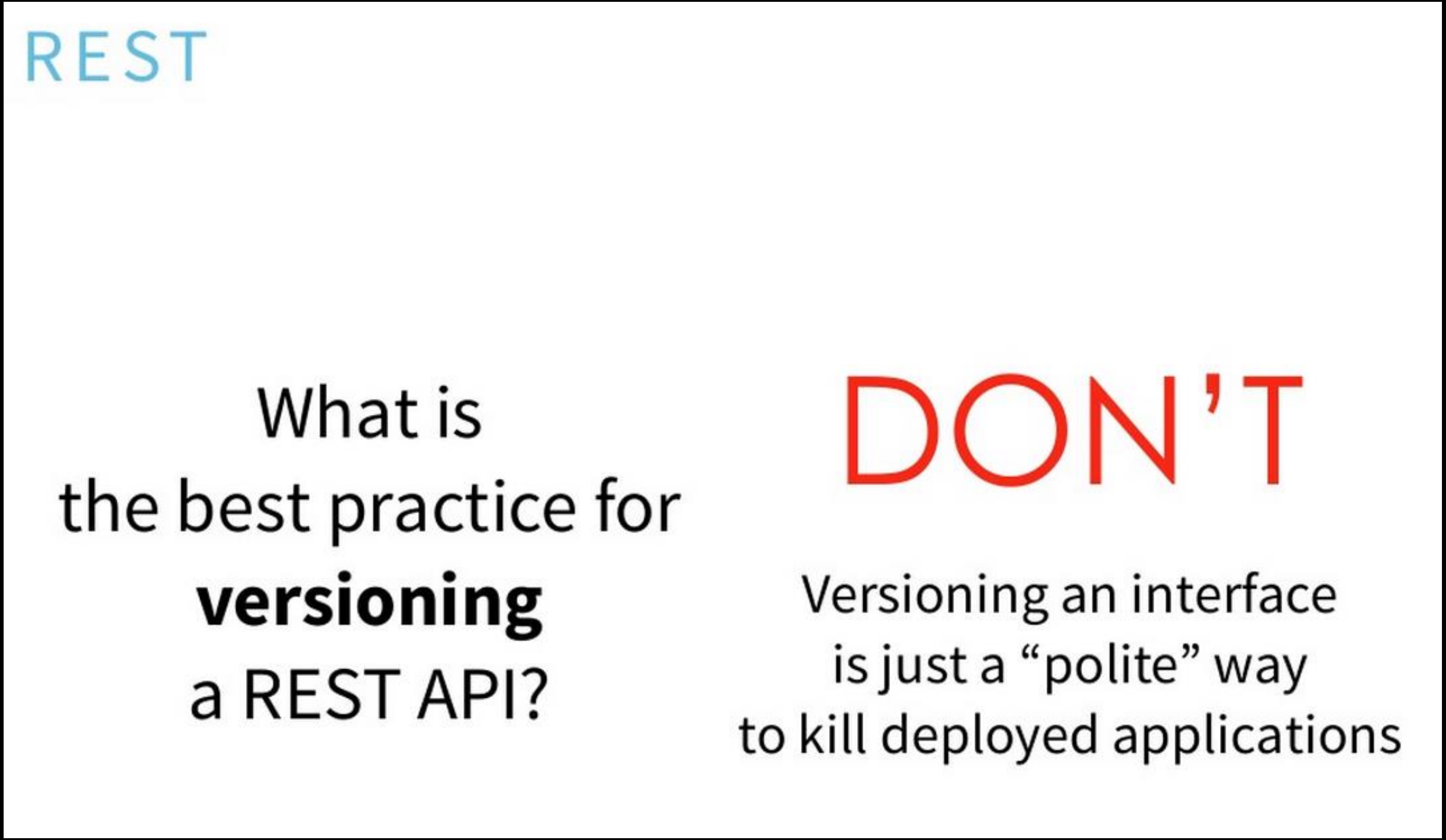
But those App Stores will be a pain!

Client w/o Hypermedia (Review)

- Get JSON objects/collections from server
- Put Read/Write semantics in the client Code

- Changing URLs == new version
- Adding new URLs == new version
- Adding new features == new version

Hey, what would Roy Do?



REST

What is
the best practice for
versioning
a REST API?

DON'T

Versioning an interface
is just a "polite" way
to kill deployed applications

Slide #22 from Roy Fielding's talk at Adobe's Evolve 2013

CAN'T TELL
IF SERIOUS.....
memegenerator.net

Huh, OK. Show me.

A Client with Hypermedia Links

Client with Hypermedia Links

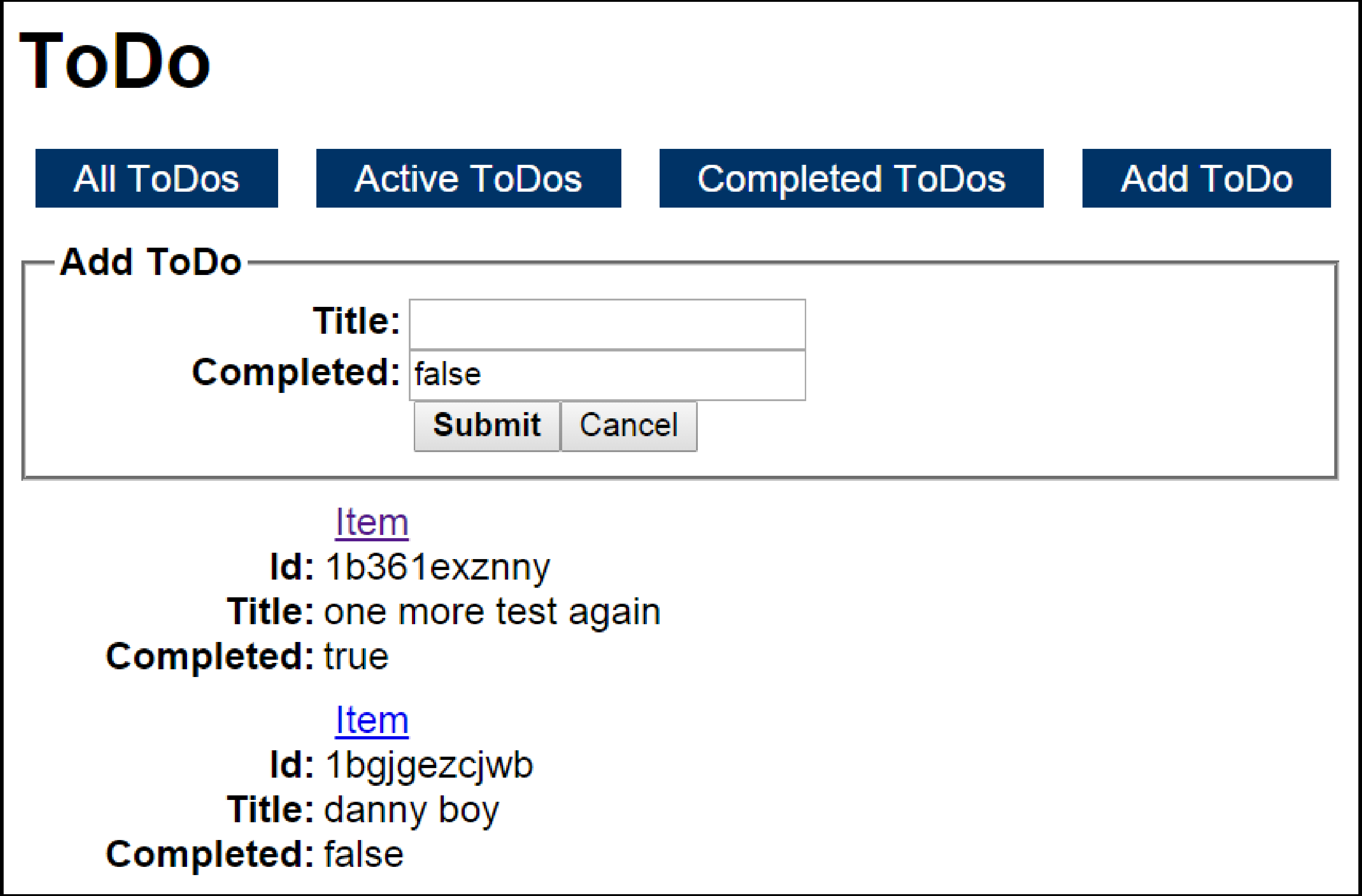
- Get the server to emit URLs
- Get the client to use the server's URLs

```
{
  "todo": [
    {
      "id": "1b361exznny",
      "title": "one more test again",
      "completed": "true",
      "href": "//localhost:8181/1b361exznny"
    },
    ...
    {
      "id": "sw26zcf6ve",
      "title": "one more simple test",
      "completed": "false",
      "href": "//localhost:8181/sw26zcf6ve"
    }
  ],
  "actions": {
    "collection": "//localhost:8181/",
    "active": "//localhost:8181/?completed=false",
    "completed": "//localhost:8181/?completed=true"
  }
}
```

Get the server to emit URLs

```
// pull actions from message
for(var link in g.msg.actions) {
  li = d.node("li");
  a = d.anchor({
    href:g.msg.actions[link],
    rel:link,
    className:"action",
    text:link
  });
  a.onclick = httpGet;
  a.style.textTransform="capitalize";
  d.push(a,li);
  d.push(li, ul);
}
```

Client now pulls the URLs from the message.


ToDo

Add ToDo

Title:

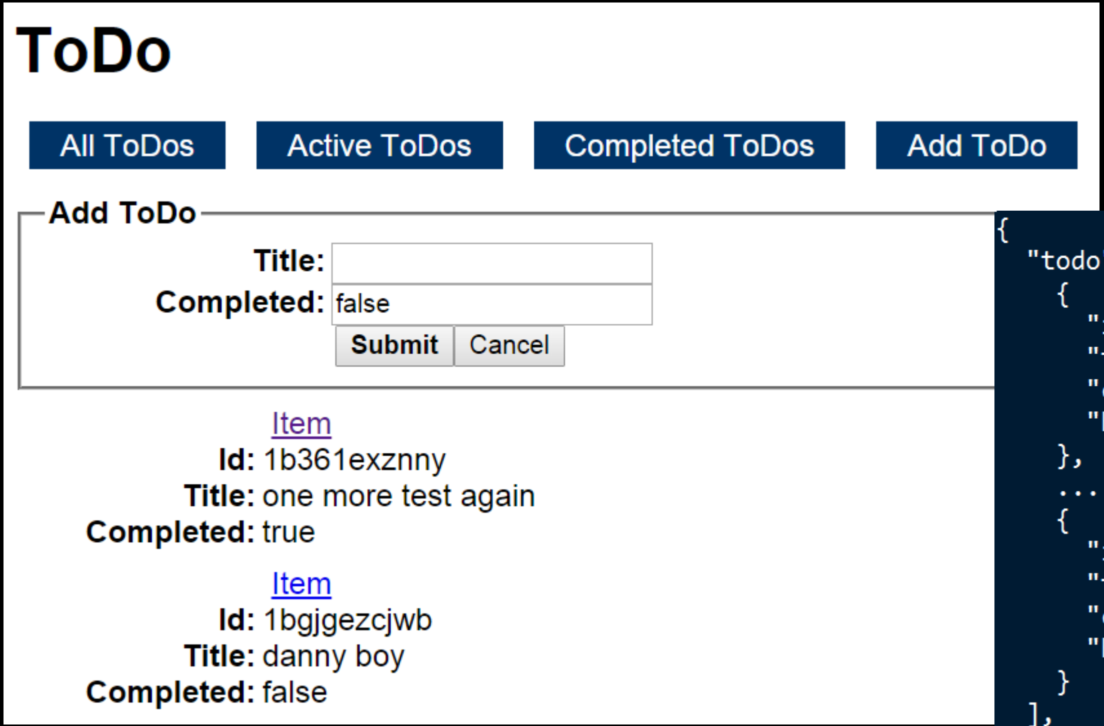
Completed: false

Item
Id: 1b361exznny
Title: one more test again
Completed: true

Item
Id: 1bgjgezcjwb
Title: danny boy
Completed: false

Live Demo of Client with Hypermedia Links

Hold on. I think you skipped something!


ToDo

All ToDos | Active ToDos | Completed ToDos | Add ToDo

Add ToDo

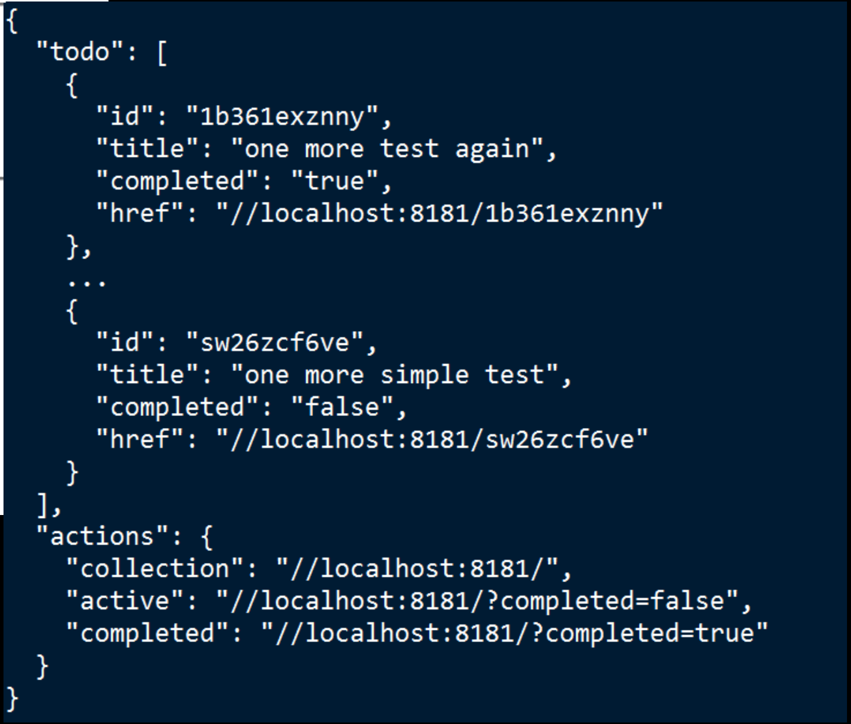
Title:

Completed: false

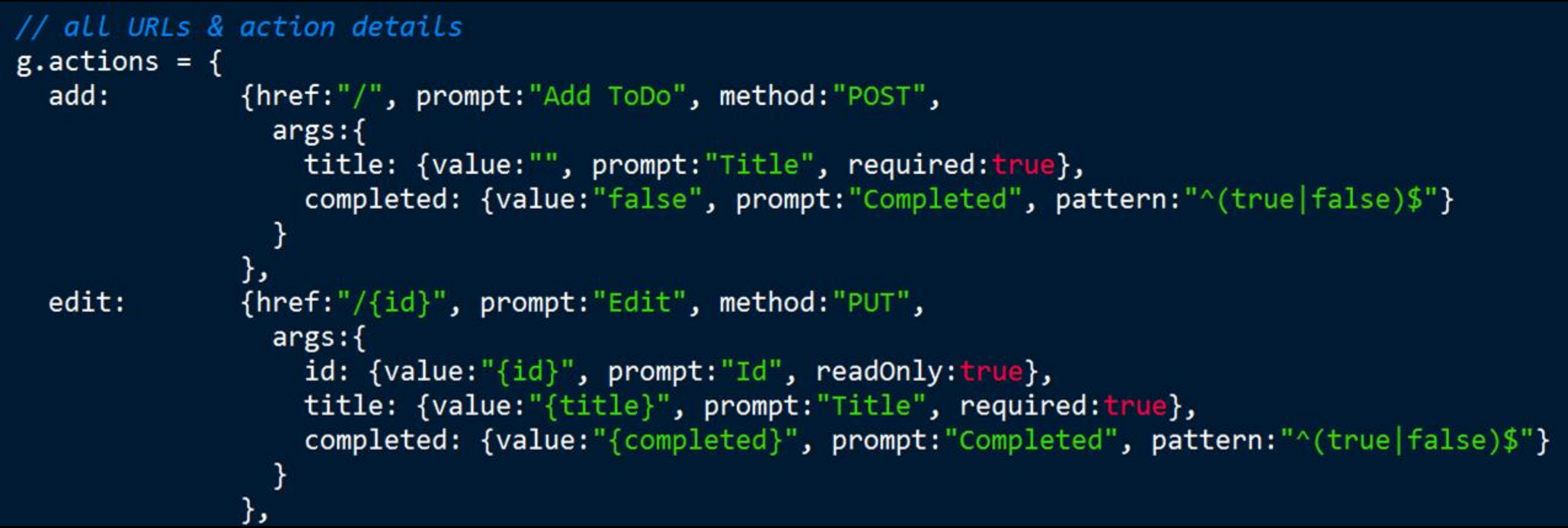
Submit | Cancel

Item
Id: 1b361exznny
Title: one more test again
Completed: true

Item
Id: 1bgjgezcjwb
Title: danny boy
Completed: false


```
{
  "todo": [
    {
      "id": "1b361exznny",
      "title": "one more test again",
      "completed": "true",
      "href": "//localhost:8181/1b361exznny"
    },
    ...
    {
      "id": "sw26zcf6ve",
      "title": "one more simple test",
      "completed": "false",
      "href": "//localhost:8181/sw26zcf6ve"
    }
  ],
  "actions": {
    "collection": "//localhost:8181/",
    "active": "//localhost:8181/?completed=false",
    "completed": "//localhost:8181/?completed=true"
  }
}
```

Where did the "Add ToDo" come from?



```
// all URLs & action details
g.actions = {
  add:          {href:"/", prompt:"Add ToDo", method:"POST",
                  args:{
                    title: {value:"", prompt:"Title", required:true},
                    completed: {value:"false", prompt:"Completed", pattern:"^(true|false)$"}
                  }
                },
  edit:         {href:"/{id}", prompt:"Edit", method:"PUT",
                  args:{
                    id: {value:"{id}", prompt:"Id", readOnly:true},
                    title: {value:"{title}", prompt:"Title", required:true},
                    completed: {value:"{completed}", prompt:"Completed", pattern:"^(true|false)$"}
                  }
                },
```

We're only sending empty links. We still need to code all the payload operations!

Then you want to add some features…

How about "Remove ToDo"?

But the installed app IGNORES the new stuff!

Oh, right. Now I need a new VERSION!

```
// all URLs & action details
g.actions = {
  add:          {href:"/", prompt:"Add ToDo", method:"POST",
                  args:{
                    title: {value:"", prompt:"Title", required:true},
                    completed: {value:"false", prompt:"Completed", pattern:"^(true|false)$"}
                  }
                },
  edit:         {href:"/{id}", prompt:"Edit", method:"PUT",
                  args:{
                    id: {value:"{id}", prompt:"Id", readOnly:true},
                    title: {value:"{title}", prompt:"Title", required:true},
                    completed: {value:"{completed}", prompt:"Completed", pattern:"^(true|false)$"}
                  }
                },
  remove:       {href:"/{id}", prompt:"Delete", method:"DELETE"}
};
```

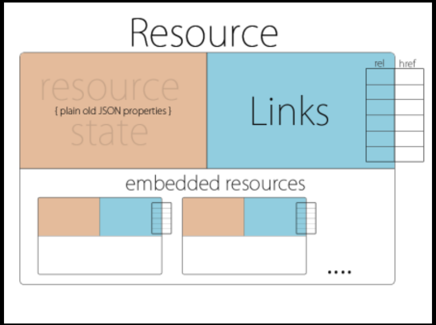
Changes in payload operations are NOT FREE

Client with Hypermedia Links (Review)

- Get the server to emit URLs
- Get the client to use the server's URLs

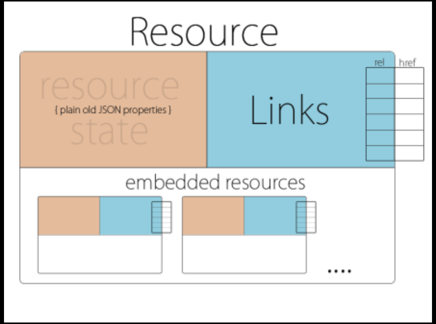
- Changing URLs is "free"
- Adding new URLs is "free"
- Adding new payload operations === new version

SIDEBAR: HAL Media Type

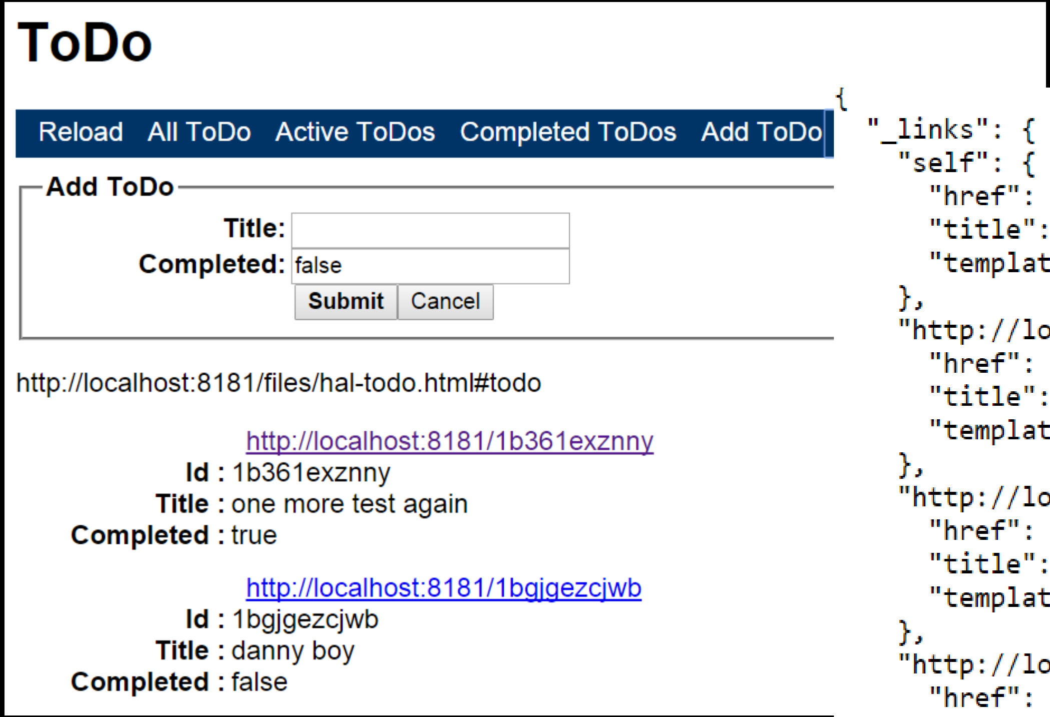


HAL Media Type

- Designed by Mike Kelly, 2011
- Makes sending LINKS easy, standardized
- Lots of library support
- Used by Amazon and others



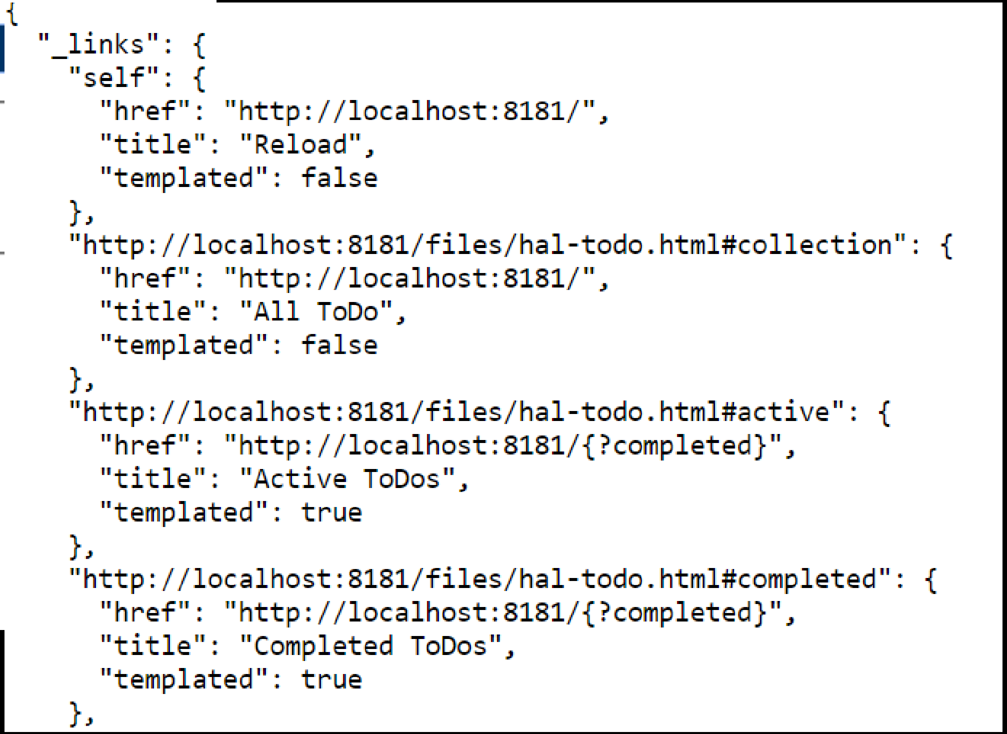
```json
{
    "_links": {
        "self": { "href": "/orders" },
        "curies": [{ "name": "ea", "href": "http://example.com/docs/rels/{rel}", "templated": true }],
        "next": { "href": "/orders?page=2" },
        "ea:find": {
            "href": "/orders{?id}",
            "templated": true
        },
        "ea:admin": [{
            "href": "/admins/2",
            "title": "Fred"
        }, {
            "href": "/admins/5",
            "title": "Kate"
        }]
    },
    "currentlyProcessing": 14,
    "shippedToday": 20,
    "_embedded": {
        "ea:order": [{
            "_links": {
                "self": { "href": "/orders/123" },
                "ea:basket": { "href": "/baskets/98712" },
                "ea:customer": { "href": "/customers/7809" }
            },
            "total": 30.00,
            "currency": "USD",
            "status": "shipped"
        }, {
            "_links": {
                "self": { "href": "/orders/124" },
                "ea:basket": { "href": "/baskets/97213" },
```



http://localhost:8181/files/hal-todo.html#todo

http://localhost:8181/1b361exznny
Id : 1b361exznny
Title : one more test again
Completed : true

http://localhost:8181/1bgjgezcjwb
Id : 1bgjgezcjwb
Title : danny boy
Completed : false



```
{
  "_links": {
    "self": {
      "href": "http://localhost:8181/",
      "title": "Reload",
      "templated": false
    },
    "http://localhost:8181/files/hal-todo.html#collection": {
      "href": "http://localhost:8181/",
      "title": "All ToDo",
      "templated": false
    },
    "http://localhost:8181/files/hal-todo.html#active": {
      "href": "http://localhost:8181/{?completed}",
      "title": "Active ToDos",
      "templated": true
    },
    "http://localhost:8181/files/hal-todo.html#completed": {
      "href": "http://localhost:8181/{?completed}",
      "title": "Completed ToDos",
      "templated": true
    },
```

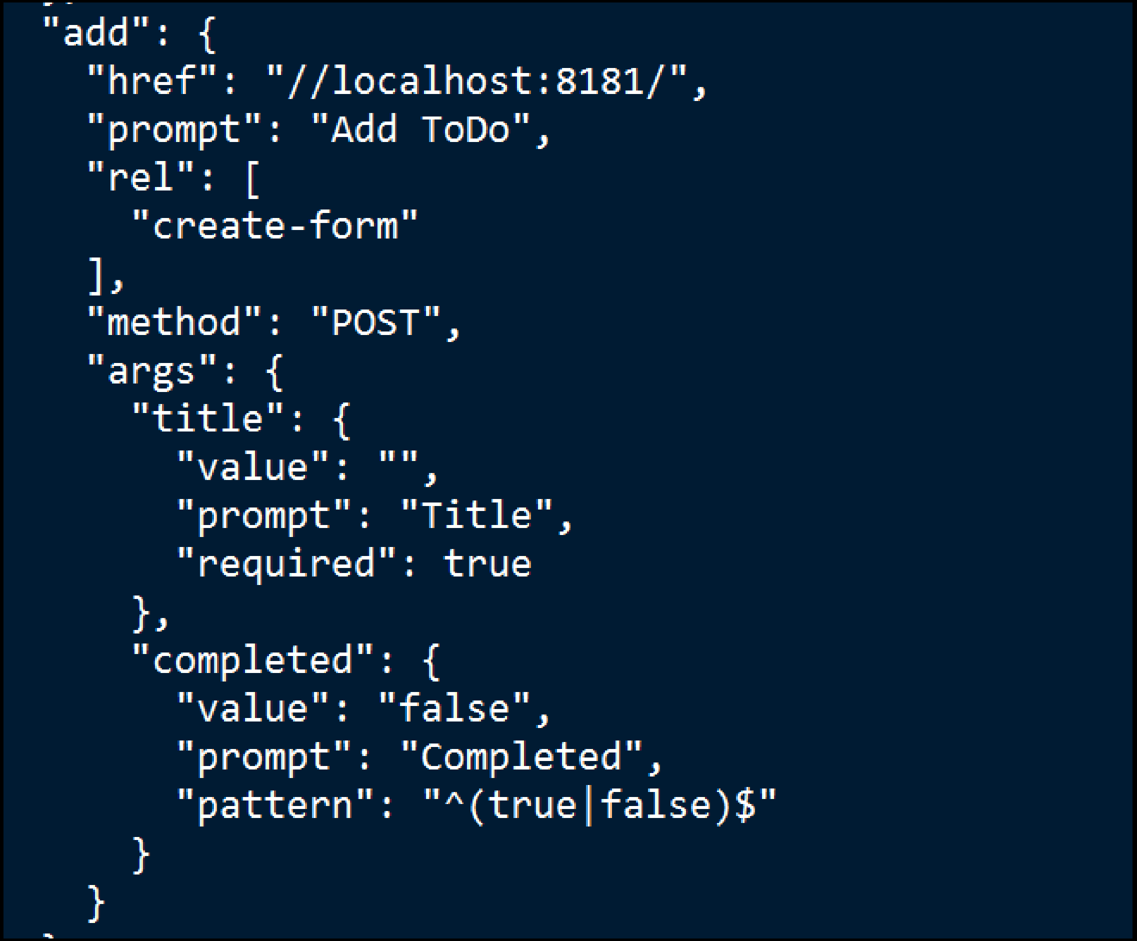
ToDo App using HAL

OK, how can I get all new features for "free"?

A Client with Hypermedia Templates

Client with Hypermedia Templates

- Get the server to emit Templates
- Get the client to use the server's Templates

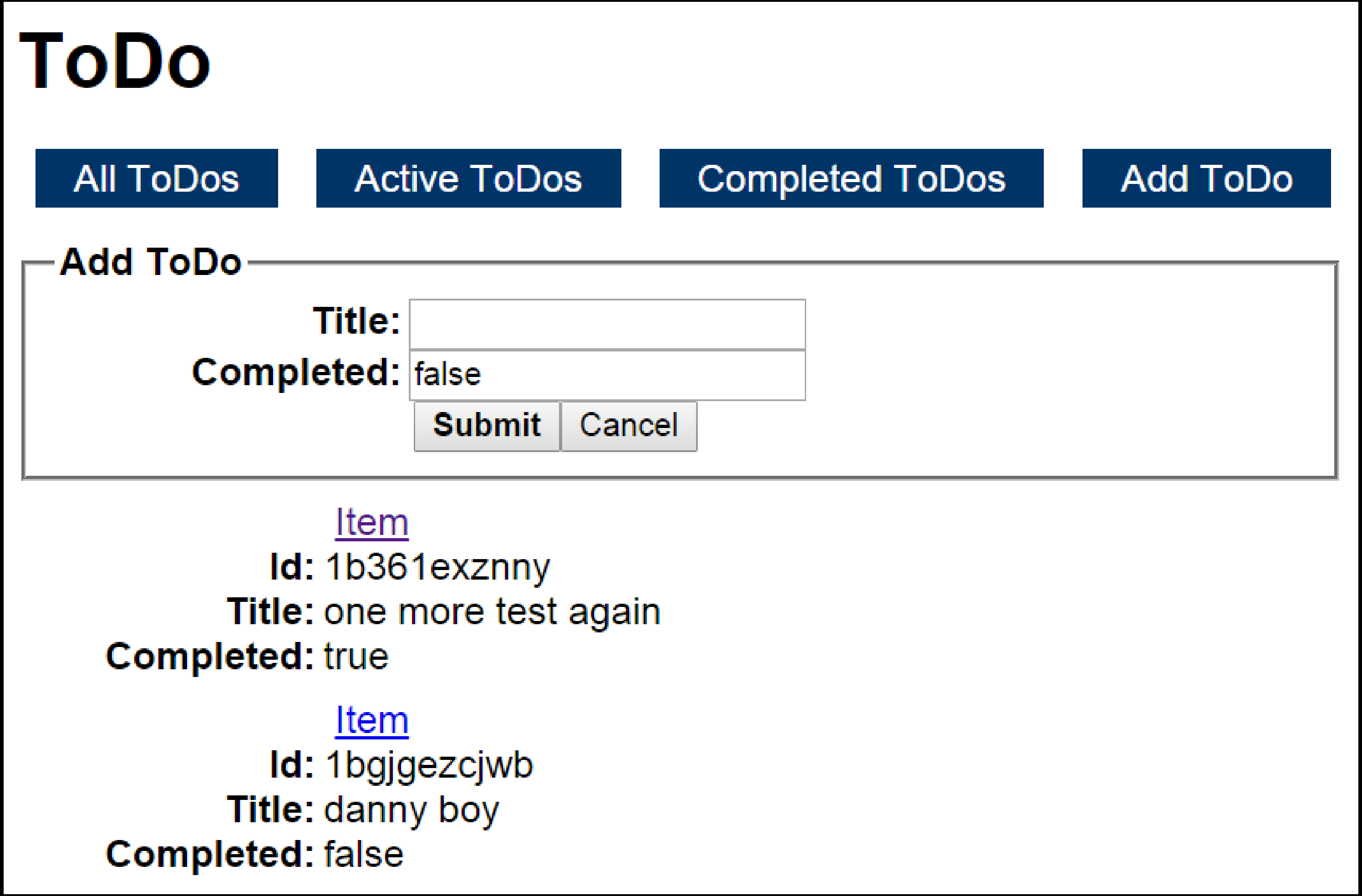


```
"add": {
  "href": "//localhost:8181/",
  "prompt": "Add ToDo",
  "rel": [
    "create-form"
  ],
  "method": "POST",
  "args": {
    "title": {
      "value": "",
      "prompt": "Title",
      "required": true
    },
    "completed": {
      "value": "false",
      "prompt": "Completed",
      "pattern": "^(true|false)$"
    }
  }
```

Get Server to emit templates.

```
// pull actions from message
for(var idx in g.msg.actions) {
  link = g.msg.actions[idx];
  li = d.node("li");
  a = d.anchor({
    href:link.href,
    rel:link.rel.split(" ")||"action",
    className:"action",
    text:link.prompt
  });
  if(link.method) {
    a.onclick = jsonForm;
    a.setAttribute("args",JSON.stringify(link.args));
    a.setAttribute("method",link.method);
  }
  else {
    a.onclick = httpGet;
  }
```

Client pulls all payload operation details from the message



Add ToDo

Title:

Completed: false

Item

Id: 1b361exznny

Title: one more test again

Completed: true

Item

Id: 1bgjgezcjwb

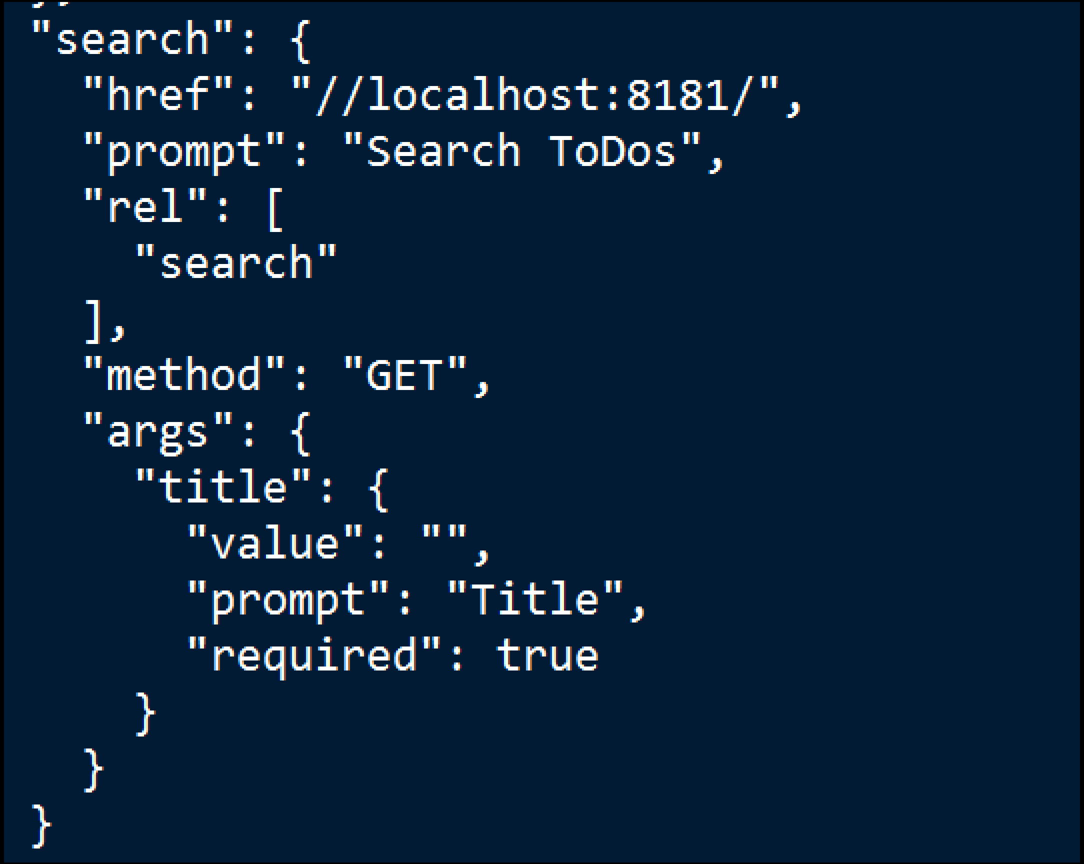
Title: danny boy

Completed: false

Live Demo of Client with Hypermedia Templates

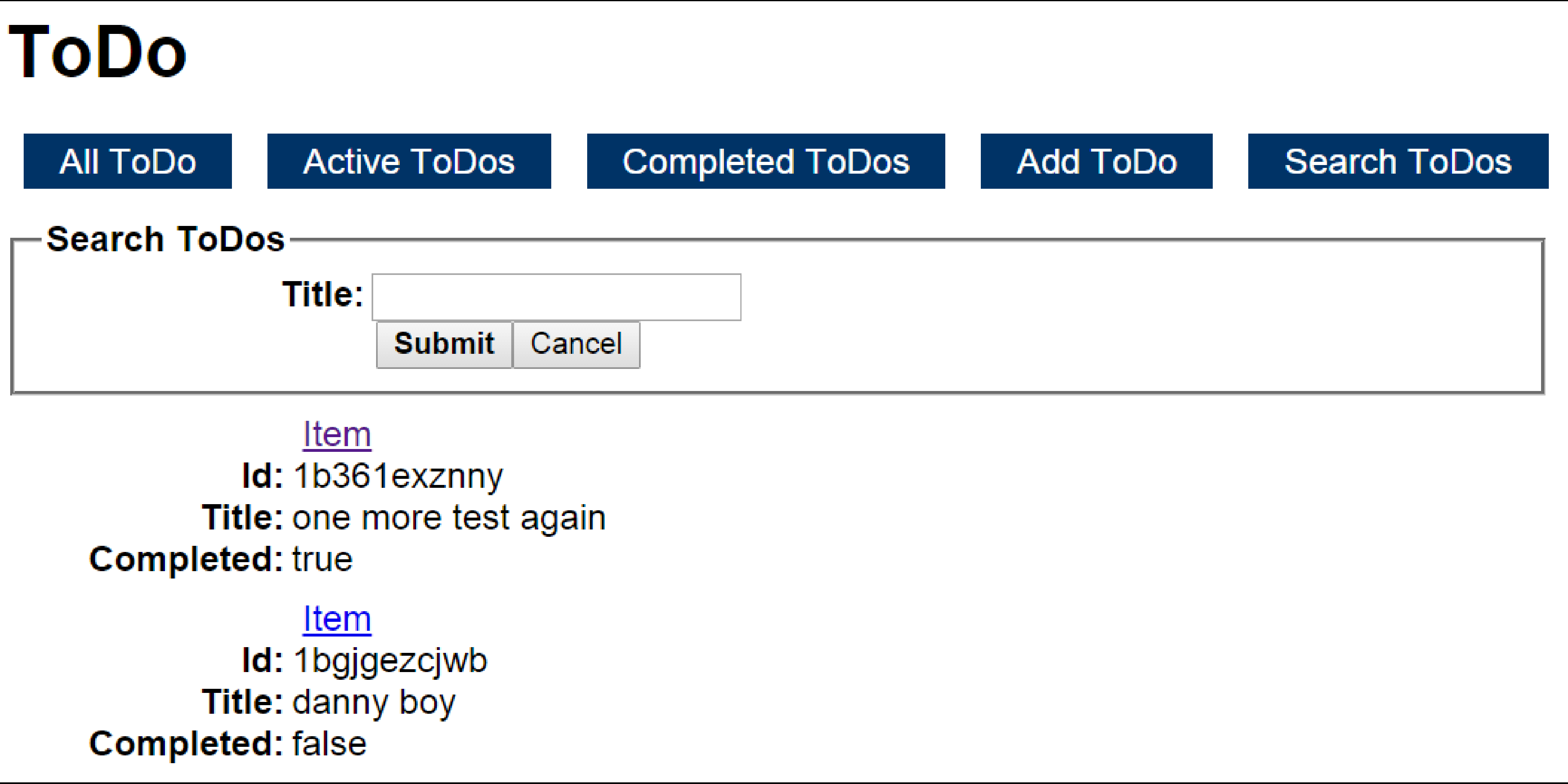
Then you want to add some features…

How about "Search ToDo"?



```
"search": {
  "href": "//localhost:8181/",
  "prompt": "Search ToDos",
  "rel": [
    "search"
  ],
  "method": "GET",
  "args": {
    "title": {
      "value": "",
      "prompt": "Title",
      "required": true
    }
  }
}
```

Get Server to emit the new Search template.



ToDo

Search ToDos

Title:

Item
Id: 1b361exznny
Title: one more test again
Completed: true

Item
Id: 1bgjgezcjwb
Title: danny boy
Completed: false

Bask in the awesomeness of hypermedia.

No Versions.

No Versions.

No Re-deploy.

No Versions.

No Re-deploy.

Just Hypermedia!

No Versions.

No Re-deploy.

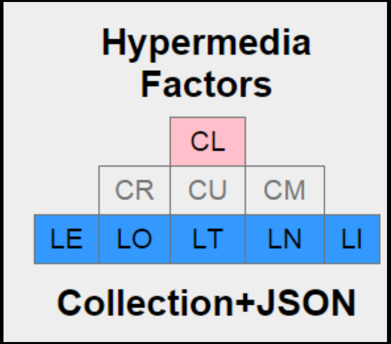
Just Hypermedia!

Client with Hypermedia Templates (Review)

- Get the server to emit Templates
- Get the client to use the server's Templates

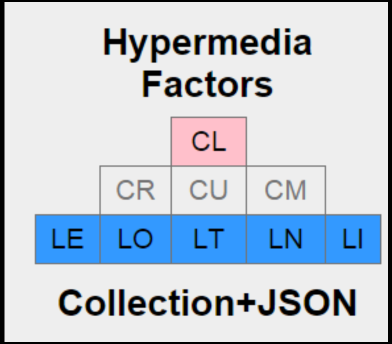
- Changing Templates are "free"
- Adding new Templates is "free"
- Adding new payload operations is "free"

SIDEBAR: Collection+JSON

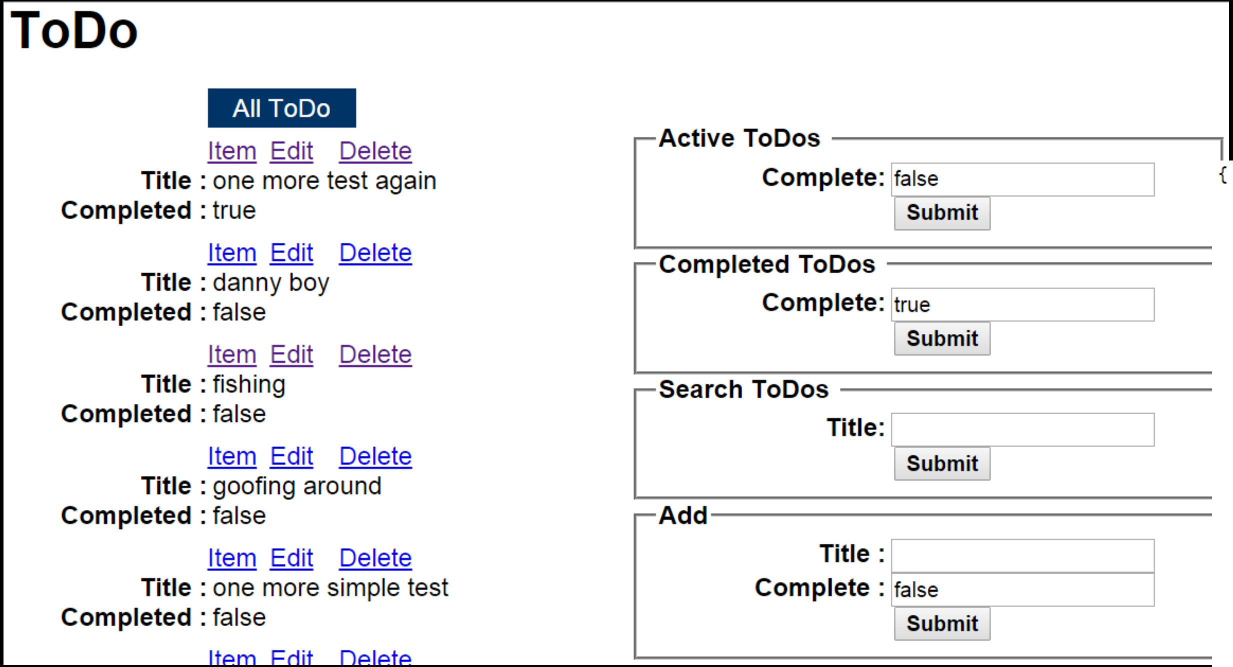


Collection+JSON Media Type

- Designed by Mike Amundsen, 2011
- Makes sending TEMPLATES easy, standardized
- Decent library support
- Used by NPR and others



```
{ "collection" :
  {
    "version" : "1.0",
    "href" : "http://example.org/friends/",

    "links" : [
      {"rel" : "feed", "href" : "http://example.org/friends/rss"}
    ],

    "items" : [
      {
        "href" : "http://example.org/friends/jdoe",
        "data" : [
          {"name" : "full-name", "value" : "J. Doe", "prompt" : "Full Name"},
          {"name" : "email", "value" : "jdoe@example.org", "prompt" : "Email"}
        ],
        "links" : [
          {"rel" : "blog", "href" : "http://examples.org/blogs/jdoe", "prompt" : "Blog"},
          {"rel" : "avatar", "href" : "http://examples.org/images/jdoe", "prompt" : "Avatar", "render" : "image"}
        ]
      }
    ],

    "queries" : [
      {"rel" : "search", "href" : "http://example.org/friends/search", "prompt" : "Search",
        "data" : [
          {"name" : "search", "value" : ""}
        ]
      }
    ],

    "template" : {
      "data" : [
        {"name" : "full-name", "value" : "", "prompt" : "Full Name"},
        {"name" : "email", "value" : "", "prompt" : "Email"},
        {"name" : "blog", "value" : "", "prompt" : "Blog"},
        {"name" : "avatar", "value" : "", "prompt" : "Avatar"}

      ]
    }
```


ToDo

All ToDo

Item Edit Delete
Title : one more test again
Completed : true

Item Edit Delete
Title : danny boy
Completed : false

Item Edit Delete
Title : fishing
Completed : false

Item Edit Delete
Title : goofing around
Completed : false

Item Edit Delete
Title : one more simple test
Completed : false

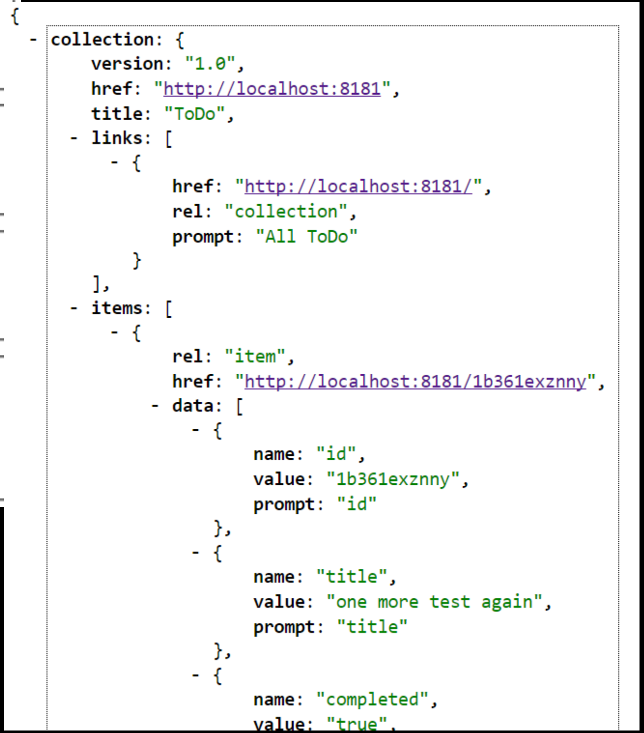
Item Edit Delete

Active ToDos
Complete: false
Submit

Completed ToDos
Complete: true
Submit

Search ToDos
Title:
Submit

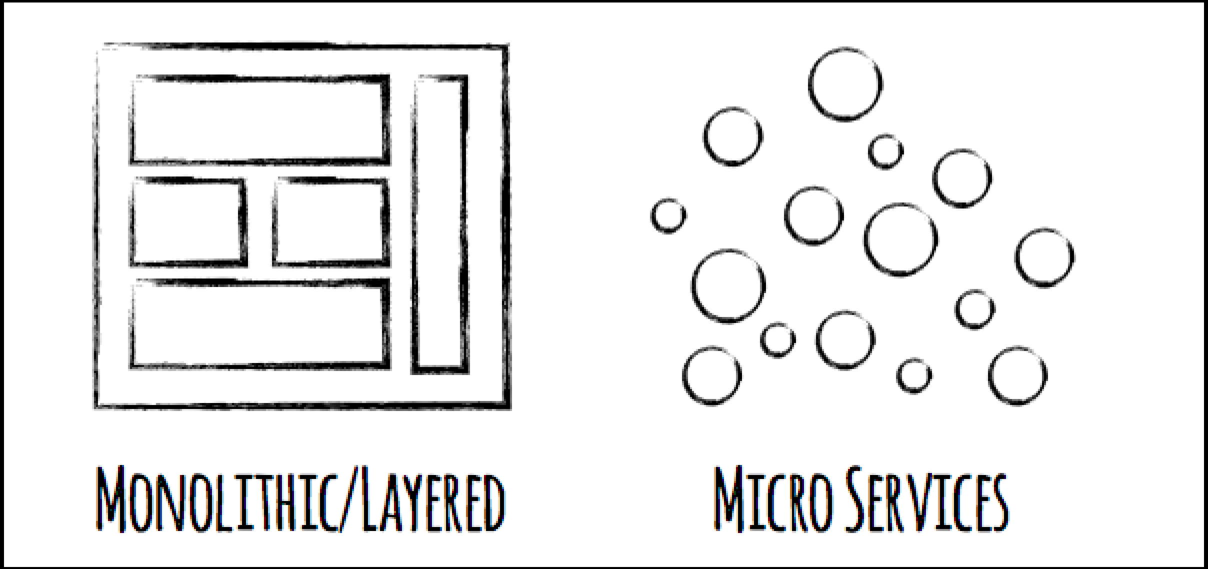
Add
Title :
Complete : false
Submit



```
{
- collection: {
      version: "1.0",
      href: "http://localhost:8181",
      title: "ToDo",
    - links: [
        - {
              href: "http://localhost:8181/",
              rel: "collection",
              prompt: "All ToDo"
          }
      ],
    - items: [
        - {
              rel: "item",
              href: "http://localhost:8181/1b361exznny",
            - data: [
                - {
                      name: "id",
                      value: "1b361exznny",
                      prompt: "id"
                  },
                - {
                      name: "title",
                      value: "one more test again",
                      prompt: "title"
                  },
                - {
                      name: "completed",
                      value: "true",
```

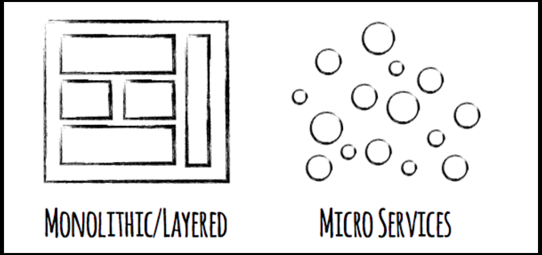
ToDo App using Collection+JSON

Microservice Bonus Slides

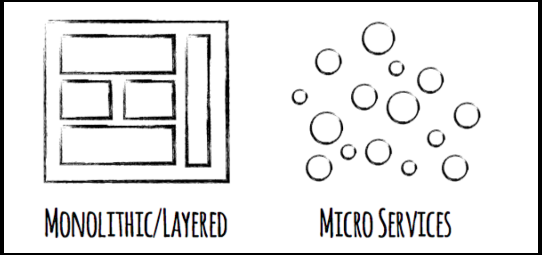


Microservices and Hypermedia

- Relocating Components
- Refactoring Features
- Re-Deploying Clients



Hypermedia with Links Supports Dynamic Relocating



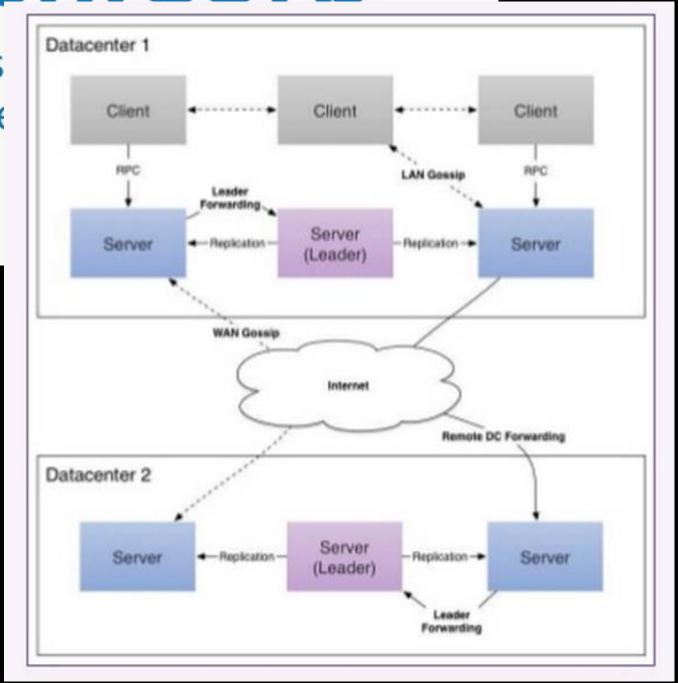
HashiCorp

Towards a
Software Managed Datacenter

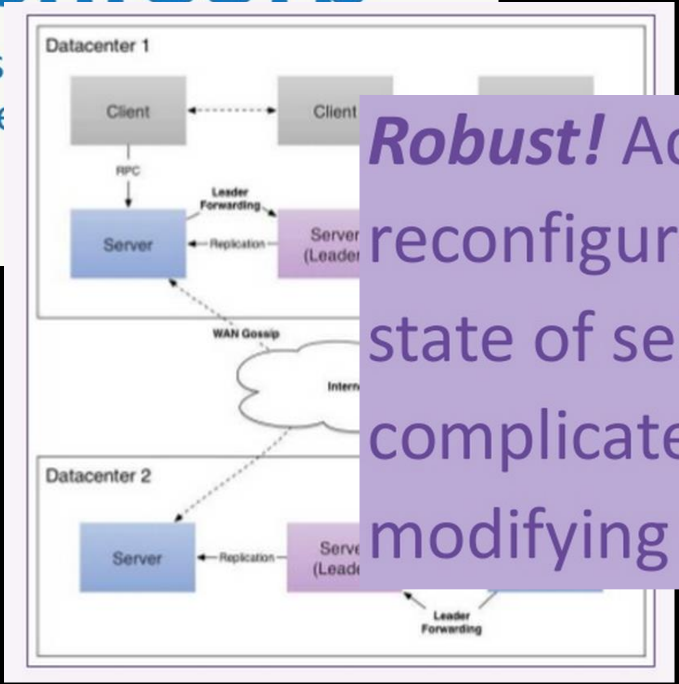
http://www.slideshare.net/DigitalOceanSlides/mitchell-hashimoto-building-robust-systems-w-service-discovery-configuration
Mitchell Hashimoto

HashiCorp

Towards
Software



http://www.slideshare.net/DigitalOceanSlides/mitchell-hashimoto-building-robust-systems-w-service-discovery-configuration
Mitchell Hashimoto



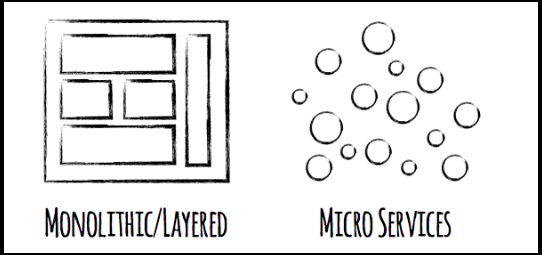
Robust! Add/remove services, reconfigure services, see global state of services without complicated logic. And without modifying application code.

http://www.slideshare.net/DigitalOceanSlides/mitchell-hashimoto-building-robust-systems-w-service-discovery-configuration
Mitchell Hashimoto

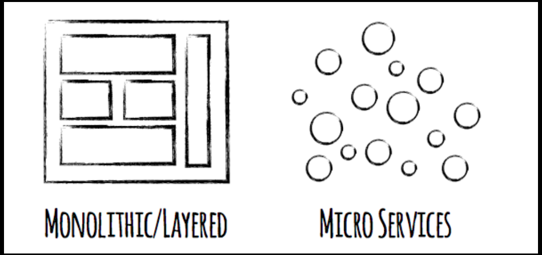
When Services Relocate…

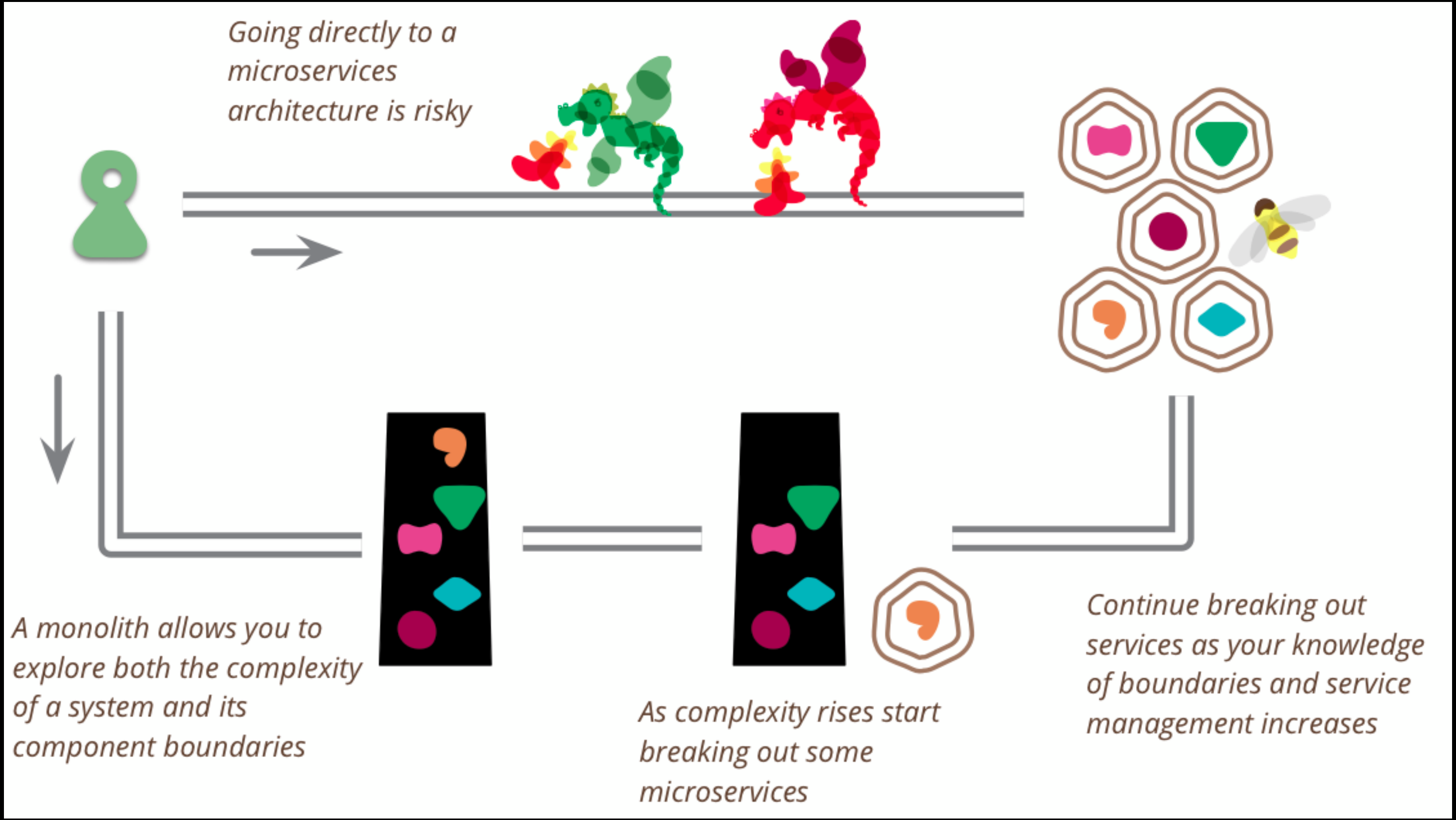
… URLs can change

Using Hypermedia Links in responses makes it easier/safer to relocate services



Hypermedia with Templates Supports Runtime Refactoring



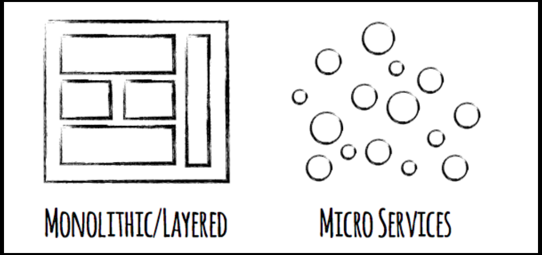

Going directly to a microservices architecture is risky
A monolith allows you to explore both the complexity of a system and its component boundaries
As complexity rises start breaking out some microservices
Continue breaking out services as your knowledge of boundaries and service management increases

http://martinfowler.com/bliki/MonolithFirst.html Martin Fowler

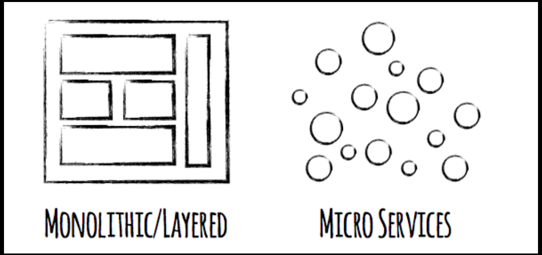
Refactoring Running Systems...

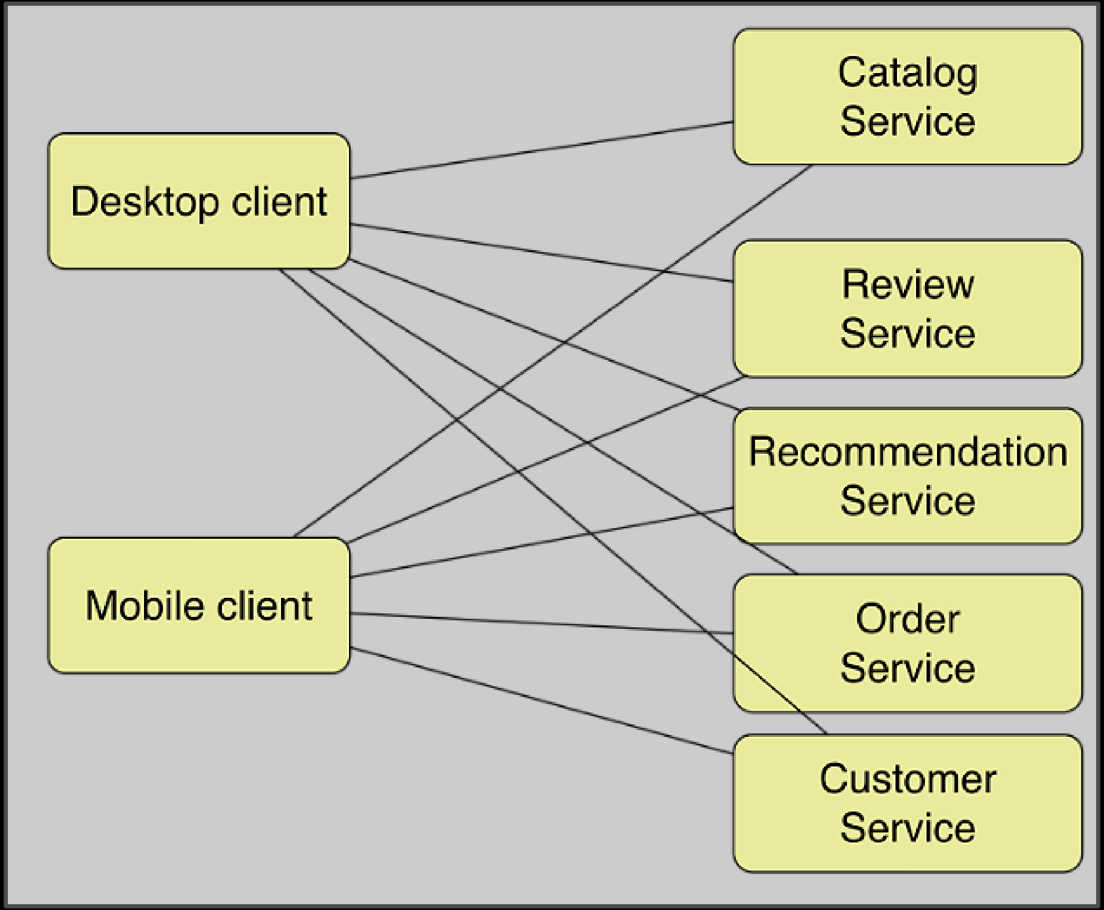
… Can change functions and arguments

Using Hypermedia Templates in responses makes it easier/safer to refactor services



Hypermedia Clients
Reduce Continuous Deployment



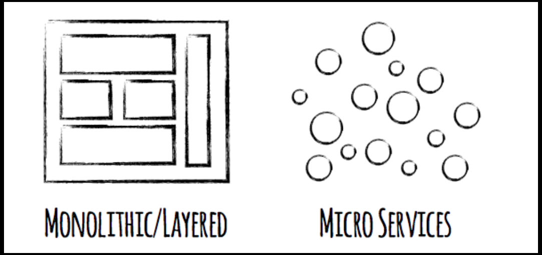


http://www.infoq.com/articles/microservices-intro Chris Richardson

Changes in workflow...

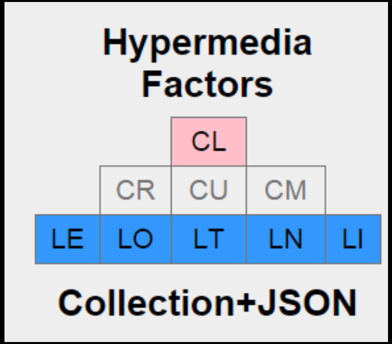
… Can break clients/consumer apps

Using Hypermedia-style apps makes it easier/safer to support new workflows without re-deployment.



Summary

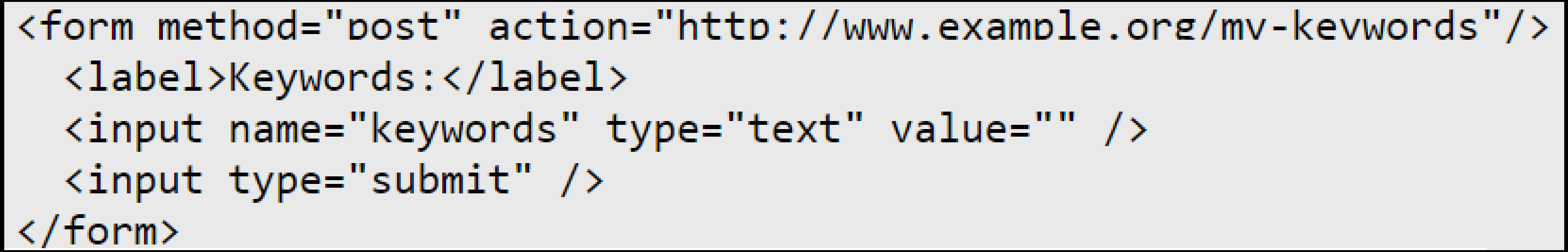
Hypermedia controls provide inline descriptions of your application's actions.



"Hypermedia as the engine of application state."

Hypermedia Constraint

Use Hypermedia



```
<form method="post" action="http://www.example.org/my-keywords"/>
  <label>Keywords:</label>
  <input name="keywords" type="text" value="" />
  <input type="submit" />
</form>
```

to execute application actions.

Client w/o Hypermedia (Review)

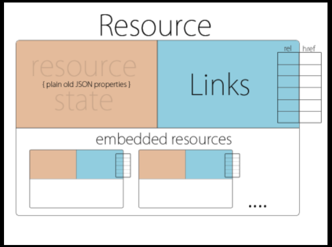
- Get JSON objects/collections from server
- Put Read/Write semantics in the client Code

- Changing URLs == new version
- Adding new URLs == new version
- Adding new features == new version

Client with Hypermedia Links (Review)

- Get the server to emit URLs
- Get the client to use the server's URLs

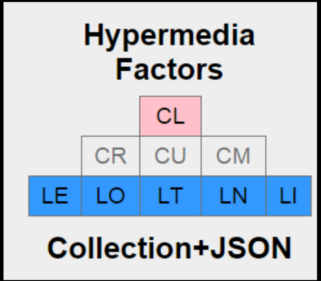
- Changing URLs is "free"
- Adding new URLs is "free"
- Adding new payload operations === new version
- Relocating services is "free"



Client with Hypermedia Templates (Review)

- Get the server to emit Templates
- Get the client to use the server's Templates

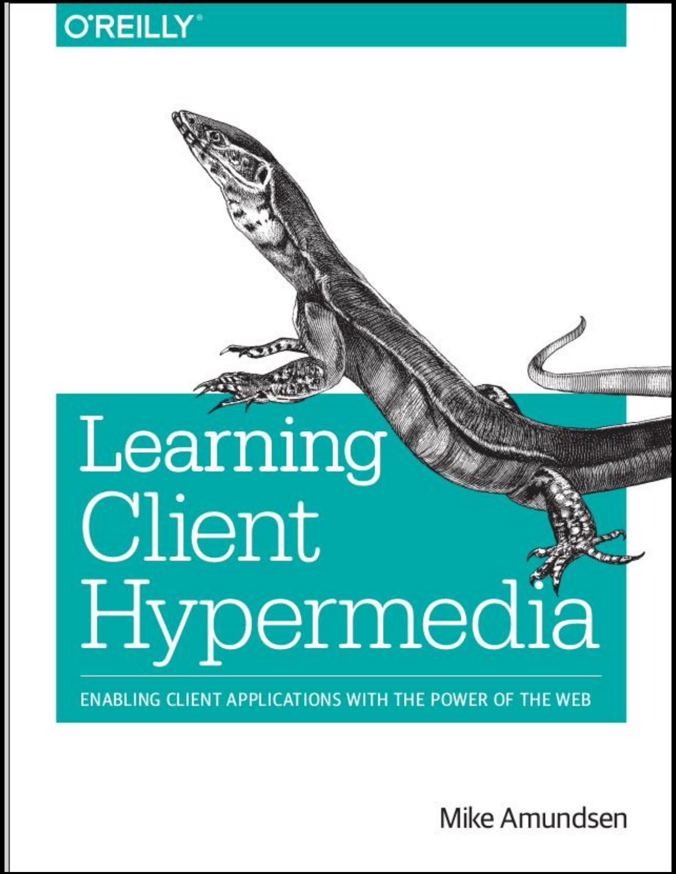
- Changing Templates are "free"
- Adding new Templates is "free"
- Adding new payload operations is "free"
- Refactoring at Runtime MAY be "free"



No Versions.

No Re-deploy.

Just Hypermedia!



O'REILLY

Learning Client Hypermedia

ENABLING CLIENT APPLICATIONS WITH THE POWER OF THE WEB

Mike Amundsen

@LCHBook (twitter & github)

Learning Client Hypermedia from the Ground Up

https://github.com/apiacademy/ndcoslo2015

Mike Amundsen,
API Academy / CA
@mamund