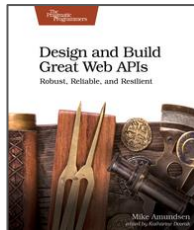


Design and Build Great APIs

@mamund

Mike Amundsen

youtube.com/mamund



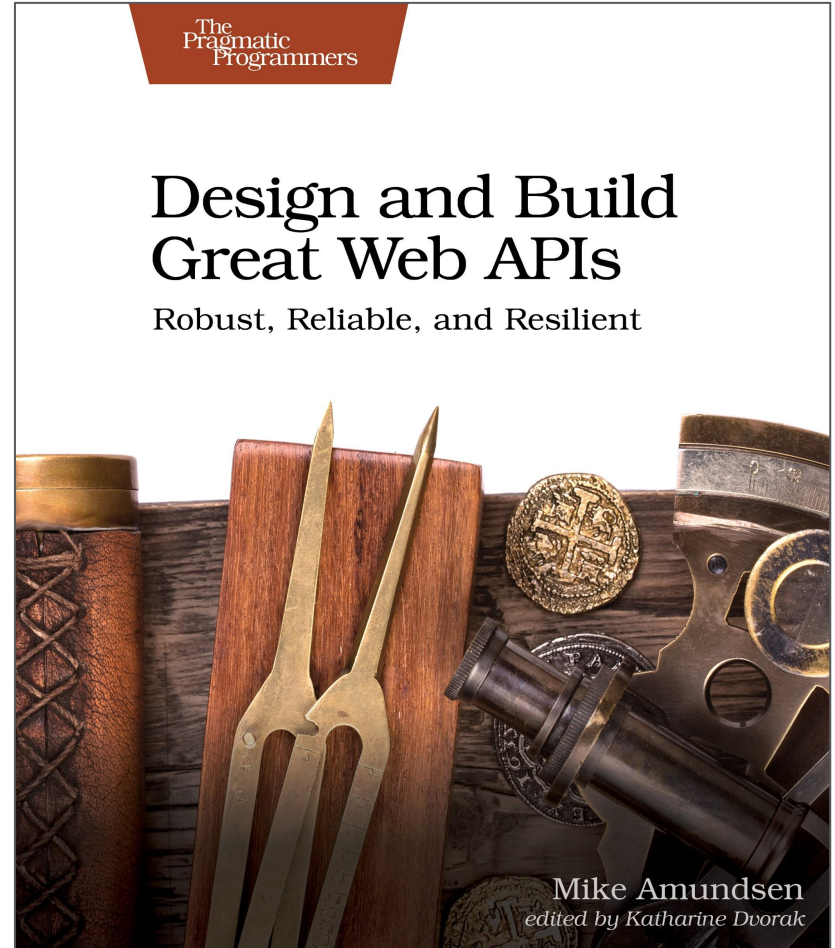
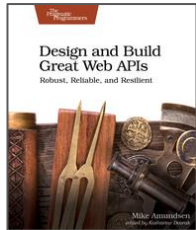


Mike Amundsen
@mamund

g.mamund.com/GreatWebAPIs

"From design to code to test to deployment, unlock hidden business value and release stable and scalable web APIs that meet customer needs and solve important business problems in a consistent and reliable manner."

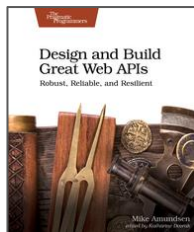
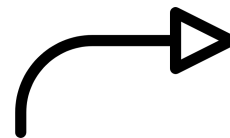
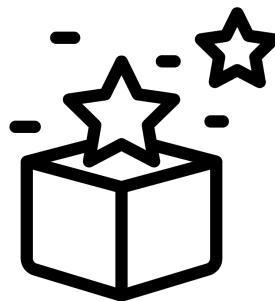
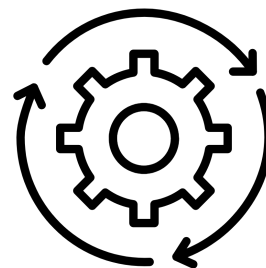
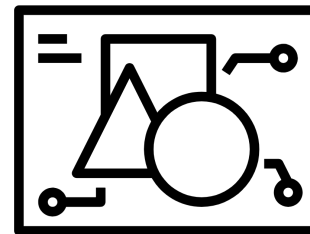
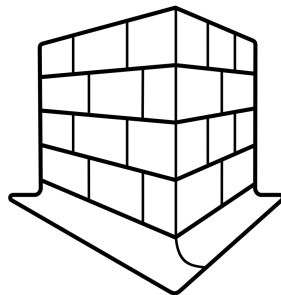
-- Pragmatic Publishers

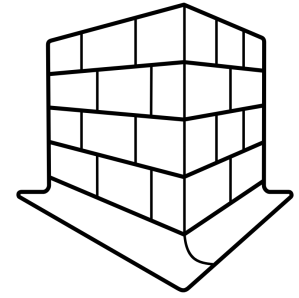


copyright © 2020 by amundsen.com, inc. -- all rights reserved

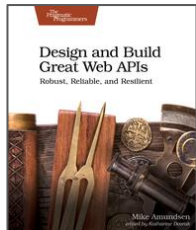
The Big Picture

- Foundations
- Design
- Build
- Release
- And Then...



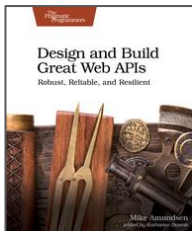
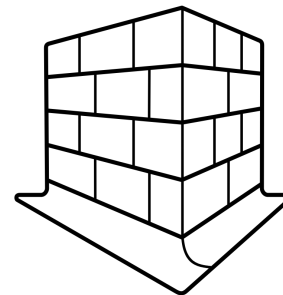


Foundations



Foundations

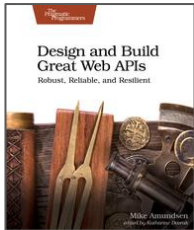
- API First
- HTTP, Web, & REST



What is API-First?

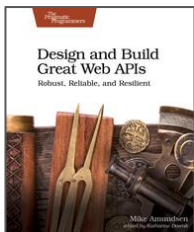
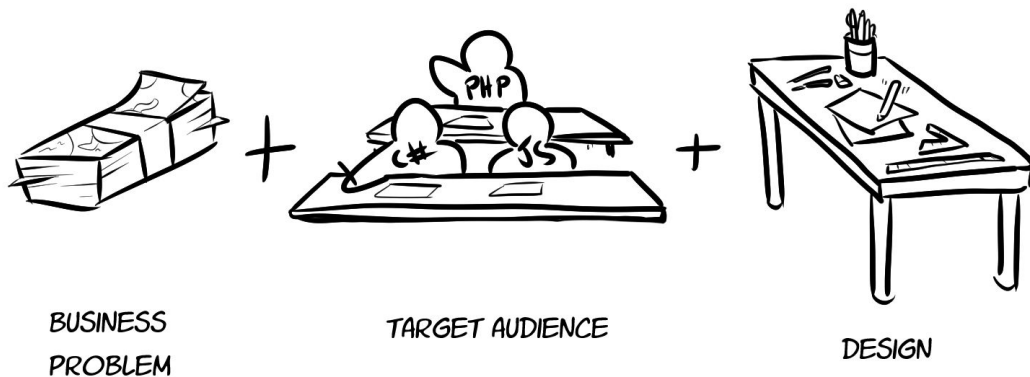
"API-first design means identifying and/or defining key actors and personas, determining what those actors and personas expect to be able to do with APIs"

-- Kas Thomas, 2009



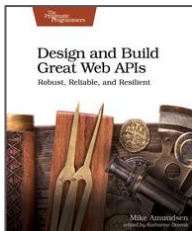
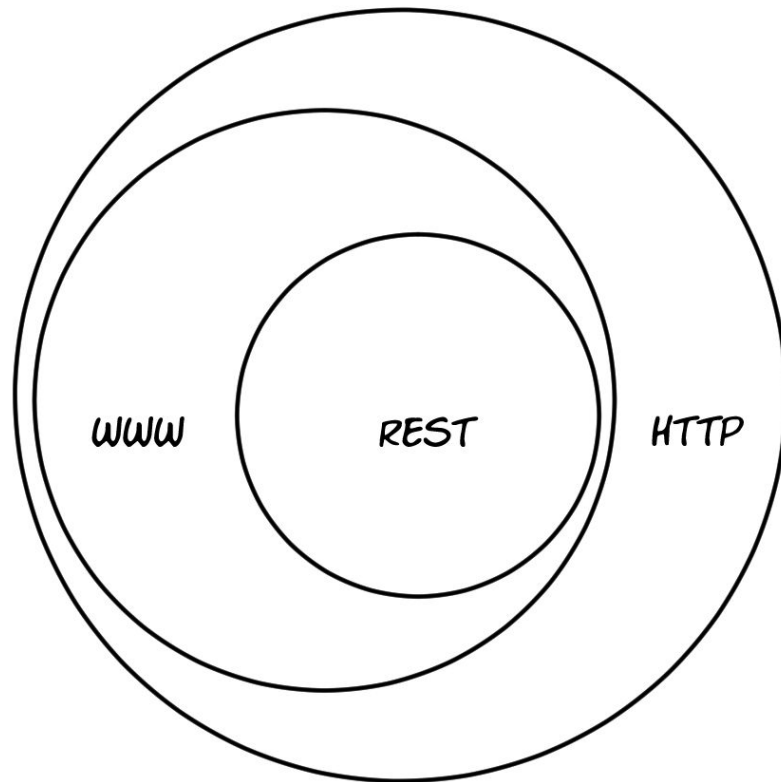
Foundations

- API First
 - APIs Solve Business Problems
 - Designing APIs for People
 - Design First
- HTTP, Web, & REST



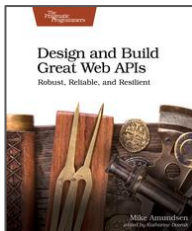
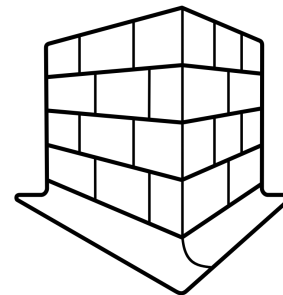
Foundations

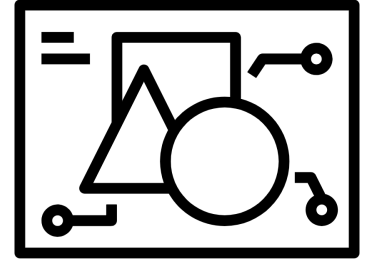
- API First
- HTTP, Web, & REST
 - HTTP is a protocol
 - URLs, Methods, Messages
 - Web is a common practice
 - SoC, links & forms
 - REST is a style
 - properties, requirements, constraints



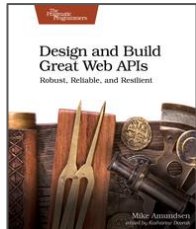
Foundations

- API First
 - Solving business problems for people
- HTTP, Web, & REST
 - Protocol, practice, style



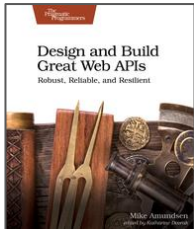
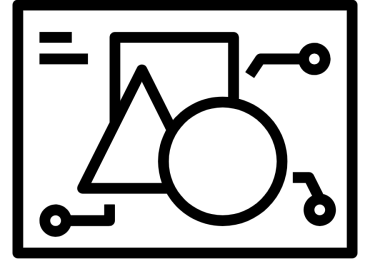


Designing APIs



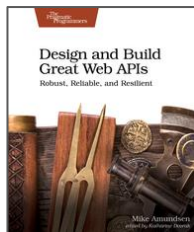
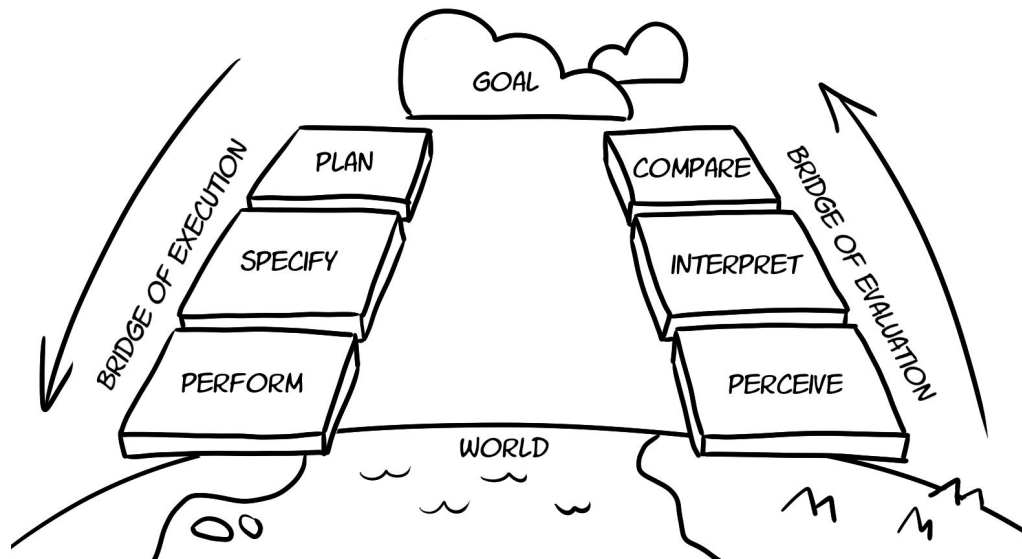
Designing APIs

- Model
- Design
- Describe



API Design

- Model
 - Norman's Lifecycle
 - There are no straight lines
 - The API Story
- Design
- Describe



13 lines (9 sloc) | 1.09 KB

Raw

Blame

History



Company Story at BigCo, Inc.

Purpose

We keep track of companies for BigCo, Inc.

Data

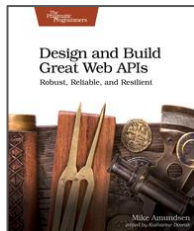
Data we include in a company record includes company name, street address, city, state/province, postal code, country, telephone, and email. Each record has a status value (pending, suspended, active, closed). We also track the date/time the record was created and the last date/time it was updated. We keep copies of the records, even after they have been deleted.

Actions

Typical work on the company records include getting the list of company records, reading a single record, creating, updating, and deleting records. You can also update the status of a single record. Finally, you can get a filtered list of all records (support for filtering by status, by country, by state, and by company name).

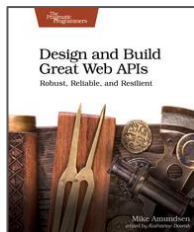
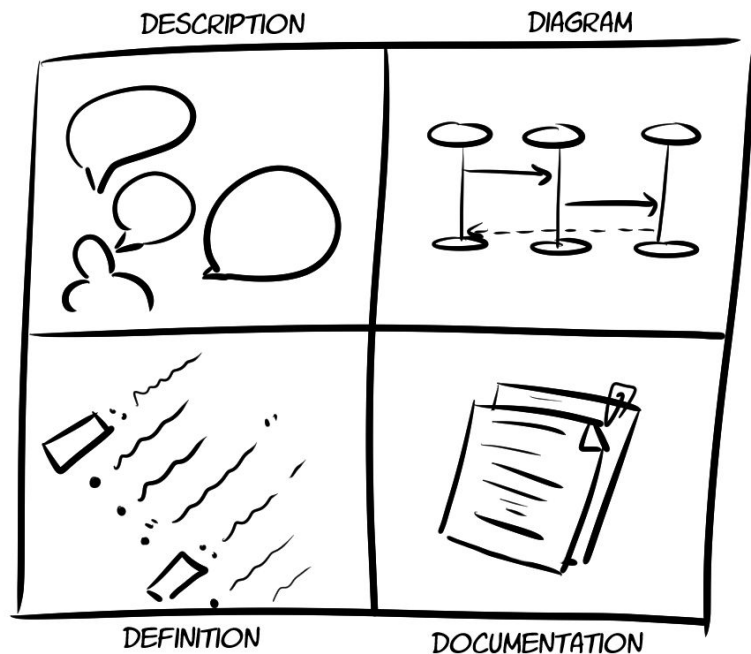
Processing

Each company has a unique identifier in the system. Right now that is a combination of the first four letters of their company name and date the company was added to the system (in the format YYYYMMDD. We add another digit if adding that does not result in a unique identifier in the system.

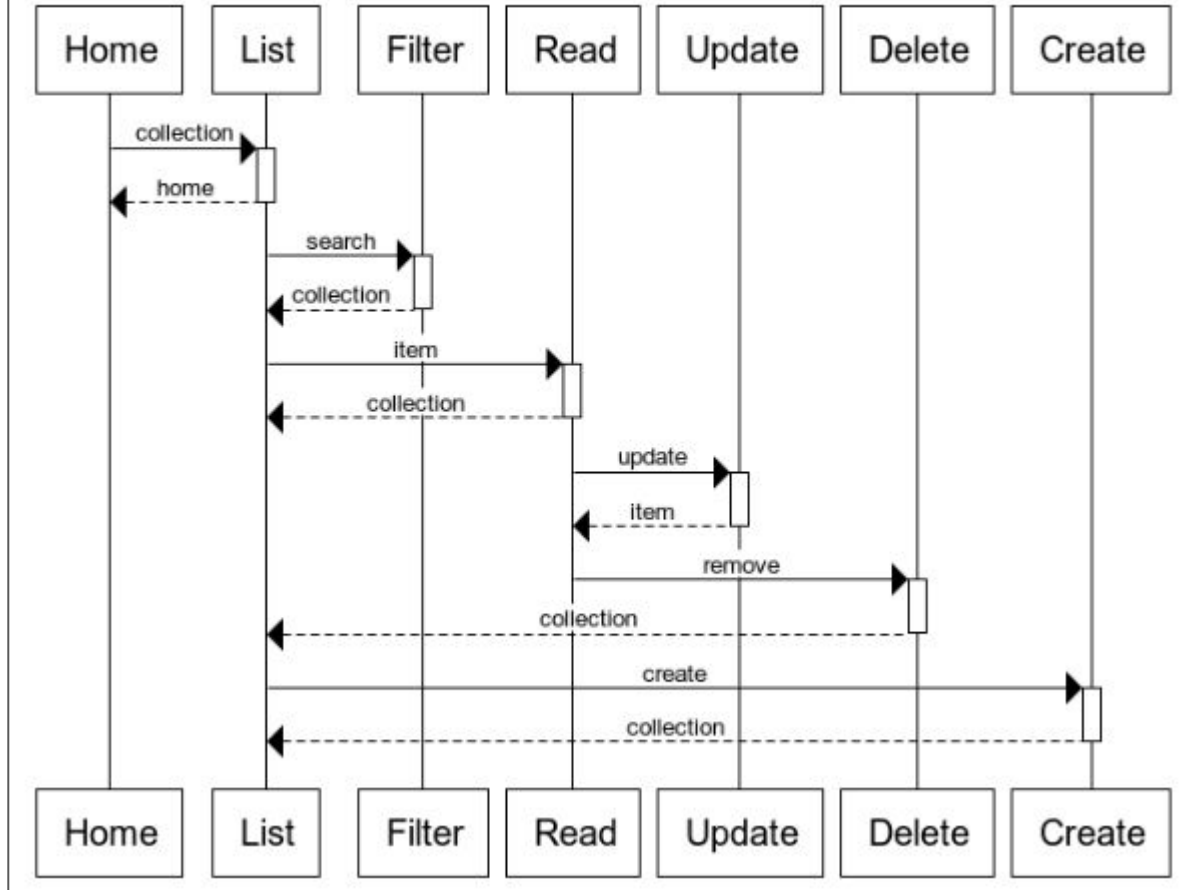


API Design

- Model
- Design
 - Design Thinking
 - Jobs-to-be-Done
 - The API Diagram
- Describe

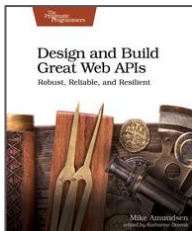
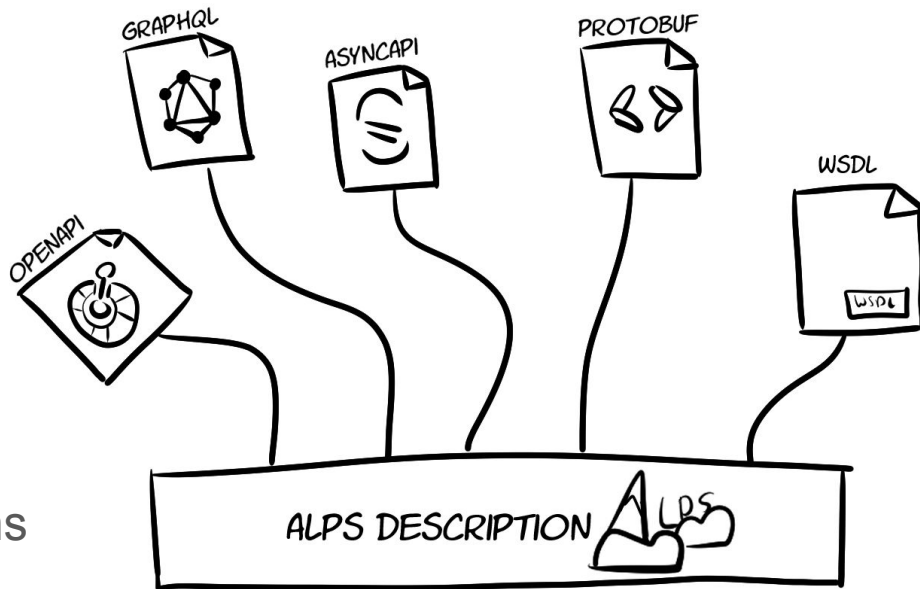


Company API Diagram



API Design

- Model
- Design
- Describe
 - Technology agnostic
 - Details on properties & actions
 - ALPS (2014)
 - <https://github.com/mamund/2020-04-unified-api-design>



91 lines (82 sloc) | 2 KB

Raw

Blame

History

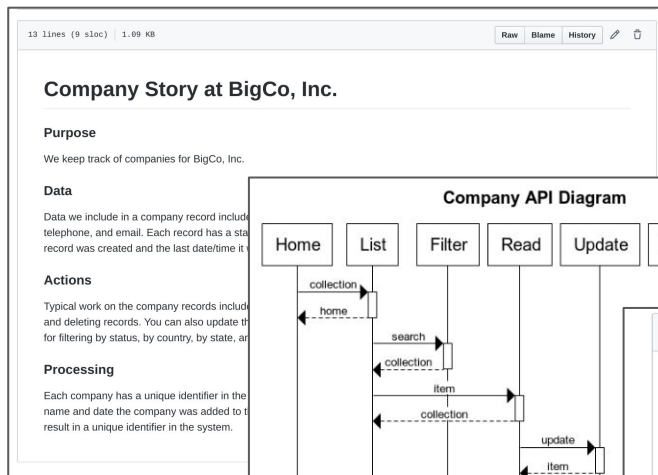
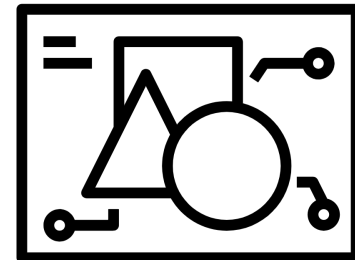


```
1  alps:
2    version: '1.0'
3    description: ALPS document for BigCo Company API
4
5    descriptors:
6      - id: home
7        type: safe
8        returns: '#company'
9
10     - id: listCompanies
11       type: safe
12       returns: '#company'
13
14     - id: filterCompanies
15       type: safe
16       returns: '#company'
17       descriptors:
18         - href: '#companyName'
19         - href: '#country'
20         - href: '#status'
21         - href: '#stateProvince'
22
23     - id: readCompany
24       type: safe
25       returns: '#company'
26       descriptors:
27         - href: '#id'
```

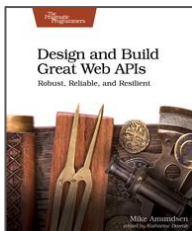
```
63     - id: company
64       type: semantic
65       descriptors:
66         - id: id
67           type: semantic
68         - id: status
69           type: semantic
70           text: 'suspended, pending, active, closed'
71         - id: companyName
72           type: semantic
73         - id: streetAddress
74           type: semantic
75         - id: city
76           type: semantic
77         - id: stateProvince
78           type: semantic
79         - id: postalCode
80           type: semantic
81         - id: country
82           type: semantic
83         - id: telephone
84           type: semantic
85         - id: email
86           type: semantic
```

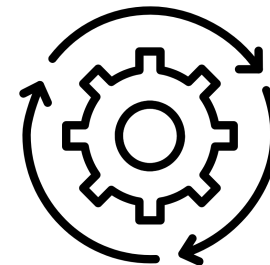
Designing APIs

- Model
 - API Story
- Design
 - API Diagram
- Describe
 - API Description

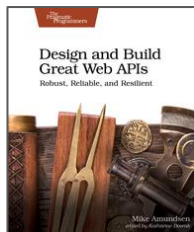


```
91 lines (82 sloc) | 2 KB
1  alps:
2    version: '1.0'
3    description: ALPS document for BigCo Comp
4
5    descriptors:
6      - id: home
7        type: safe
8        returns: '#company'
9
10     - id: listCompanies
11       type: safe
12       returns: '#company'
13
14     - id: filterCompanies
15       type: safe
16       returns: '#company'
17       descriptors:
18         - href: '#companyName'
19         - href: '#country'
20         - href: '#status'
21         - href: '#stateProvince'
22
23     - id: readCompany
24       type: safe
25       returns: '#company'
26       descriptors:
27         - href: '#id'
```



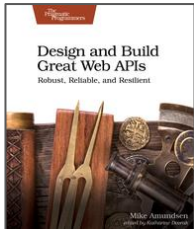
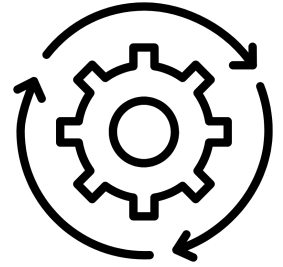


Building APIs



Building APIs

- Sketching
- Prototyping
- Building



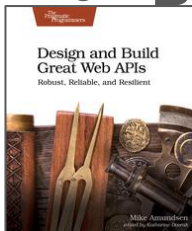
Building APIs

- Sketching

- Frank Gehry (via Ronnie Mitra)
- Experiment w/ possibilities
- Apiary Blueprint (APIB)
- Sketches are meant to be thrown away

- Prototyping

- Building



63 lines (50 sloc) | 1.51 KB

Raw

Blame

History

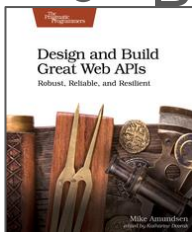


```
1  FORMAT: 1A
2  HOST: http://polls.apiblueprint.org/
3
4  # Credit Check Responses
5
6  Public API profile for BigCo's Credit Check service.
7
8  ## Credit Check History [/list]
9
10 Returns a list of past credit rating records.
11
12 ### List Past Credit Checks [GET]
13
14 + Response 200 (application/json)
15
16     {
17         "creditCheck" : [
18             {
19                 "id" : "123",
20                 "companyName" : "Vandelay Industries",
21                 "dateRequested" : "2009-06-15T13:45:30",
22                 "ratingValue" : "5"
23             },
24             {
25                 "id" : "123",
26                 "companyName" : "Vandelay Industries",
```

```
37         "id" : "789",
38         "companyName" : "Vandelay Industries",
39         "dateRequested" : "2009-06-21T08:35:15",
40         "ratingValue" : "6"
41     }
42 ]
43 }
44
45 ## Credit Check Item [/list/123]
46
47 Returns a list of past credit rating records.
48
49 ### Single Credit Checks [GET]
50
51 + Response 200 (application/json)
52
53     {
54         "creditCheck" : [
55             {
56                 "id" : "123",
57                 "companyName" : "Vandelay Industries",
58                 "dateRequested" : "2009-06-15T13:45:30",
59                 "ratingValue" : "5"
60             }
61         ]
```

Building APIs

- Sketching
- Prototyping
 - Borglum's Prototypes
 - Explore the details
 - OpenAPI (OAS)
 - Prototypes are made to be tested
- Building



284 lines (248 sloc) | 6.63 KB

Raw

Blame

History

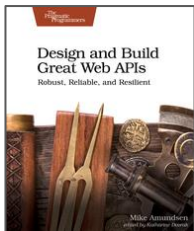
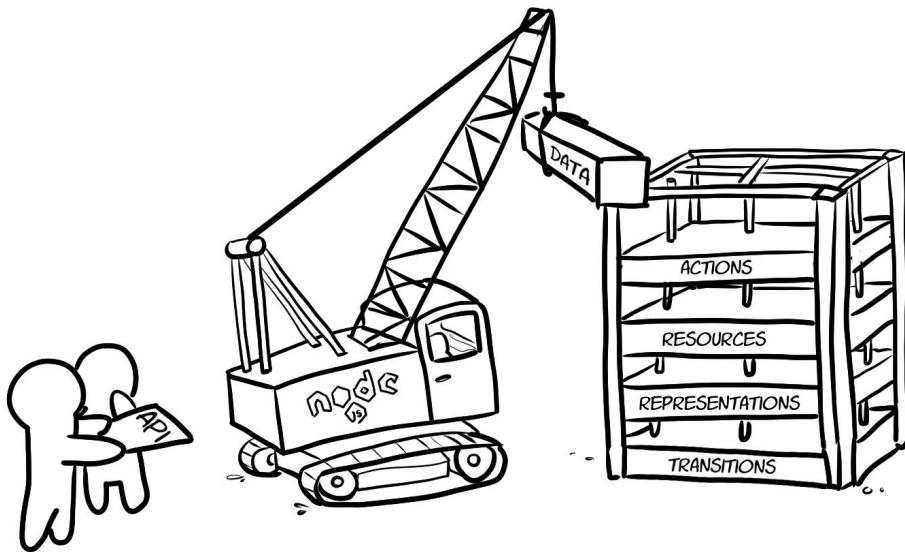


```
1  openapi: 3.0.0
2
3  #####
4  # Credit Check API
5  # 2019-12-22
6  # mamund
7  #####
8
9  ### info section ###
10 info:
11   version: '1.0.0'
12   title: 'Credit Check API'
13   description: 'Credit Check service API for BigCo, Inc.'
14
15 # Added by API Auto Mocking Plugin
16 servers:
17   - description: SwaggerHub API Auto Mocking
18     url: https://virtserver.swaggerhub.com/amundsen/credit-check-api/1
19
20 ### tags ###
21 tags:
22   - name: credit
23     description: "Credit Check API"
24
25 ### paths section ###
26 paths:
```

```
186
187   ### property models ###
188   schemas:
189
190     ### default error ###
191     error:
192       type: object
193       properties:
194         type:
195           type: string
196           example: "Invalid Record"
197         title:
198           type: string
199           example: "One or more missing properties"
200         detail:
201           type: string
202           example: "Review the submitted record for
203
204     ### 200 OK reply (prag+json) ###
205     reply:
206       type: object
207       properties:
208         metadata:
209           type: array
```

Building APIs

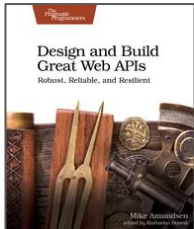
- Sketching
- Prototyping
- Building
 - Repeatable process
 - Convert prototype/design into code
 - DARRT
 - Data, Actions, Resources, Representations, Transitions



Building APIs : DARRT

A simple, repeatable process for publishing API interfaces

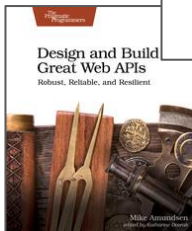
- Data
- Actions
- Resources
- Representations
- Transitions



Building APIs : DARRT : Data

- The state properties to pass in messages
 - `properties`, `requires`, `enums`, `defaults`

```
// this service's message properties
exports.props = [
  'id', 'status', 'dateCreated', 'dateUpdated',
  'companyId', 'companyName', 'streetAddress', 'city', 'stateProvince',
  'postalCode', 'country', 'telephone', 'email',
  'accountId', 'division', 'spendingLimit', 'discountPercentage',
  'activityId', 'activityType', 'dateScheduled', 'notes'
];
```



Building APIs : DARRT : Data

- The state properties to pass in messages
 - properties, requires, enums, defaults

```
// this service's message properties  
// required properties  
exports.reqd = ['id', 'status', 'created', 'dateUpdated',  
  'companyId', 'companyName', 'streetAddress', 'city', 'stateProvince',  
  'postalCode', 'country', 'telephone', 'email',  
  'accountId', 'division', 'spendingLimit', 'discountPercentage',  
  'activityId', 'activityType', 'dateScheduled', 'notes'  
];
```



Building APIs : DARRT : Data

- The state properties to pass in messages
 - properties, requires, enums, defaults

```
// this service's message properties  
// required properties  
exports.reqd = ['id', 'status', 'created', 'dateUpdated',  
  'companyId',  
  'postalCode',  
  'accountId',  
  'activityId'  
];  
// enumerated properties  
exports.enums = [  
  {status:  
    ['pending', 'active', 'suspended', 'closed']  
  },  
  {division:  
    ['DryGoods', 'Hardware', 'Software', 'Grocery', 'Pharmacy', 'Military']  
  },  
  {activityType:  
    ['email', 'inperson', 'phone', 'letter']  
  }  
];
```



Building APIs : DARRT : Data

- The state properties to pass in messages
 - `properties`, `requires`, `enums`, `defs`

```
// this service's message properties
```

```
// required properties
```

```
exports.reqd = ['id', 'status', 'created', 'dateUpdated',
```

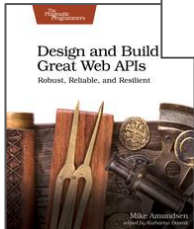
```
'companyId',  
'postalCode',  
'accountId',  
'activityId'
```

```
];
```

```
// enumerated properties
```

```
exports.enums = [  
  {status:  
    ['pending', 'active', 'suspended', ]  
  },  
  {division:  
    ['DryGoods', 'Hardware', 'Software', 'Grocery', 'Pharmacy', 'Military']  
  },  
  {activityType:  
    ['email', 'inperson', 'phone', 'letter']  
  }  
];
```

```
{name:"spendingLimit", value:"10000"},  
{name:"discountPercentage", value:"10"},  
{name:"activityType", value:"email"},  
{name:"status",value:"pending"}  
];
```

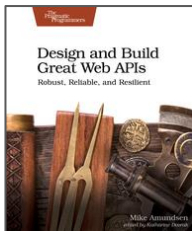


Building APIs : DARRT : Actions

- The actual internal operations for the interface
 - `approvePayroll`, `updateCustomer`, `setStatus`

`building/action-readStatus.js`

```
module.exports.readStatus = function(req,res) {
  return new Promise(function(resolve,reject){
    if(req.params.id && req.params.id!==null) {
      var id = req.params.id;
      var fields="id, status, dateCreated, dateUpdated"
      resolve(
        component(
          {name:'onboarding',action:'item',id:id, fields:fields}
        )
      );
    }
    else {
      reject({error:"missing id"});
    }
  });
}
```

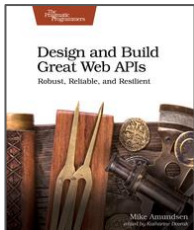


Building APIs : DARRT : Resources

- The public HTTP resources to access the operations

building/resource-list.js

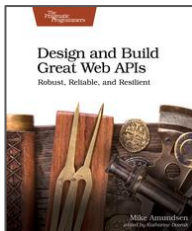
```
// *****  
// public resources for the onboarding service  
// *****  
  
router.get('/', function(req, res){ });  
router.post('/wip/', function(req, res){ });  
router.get('/wip/', function(req, res){ });  
router.get('/wip/filter/', function(req, res){ });  
router.get('/wip/:id', function(req, res){ });  
router.get('/wip/:id/company', function(req, res){ });  
router.put('/wip/:id/company', function(req, res){ });  
router.get('/wip/:id/account', function(req, res){ });  
router.put('/wip/:id/account', function(req, res){ });  
router.get('/wip/:id/activity', function(req, res){ });  
router.put('/wip/:id/activity', function(req, res){ });  
router.get('/wip/:id/status', function(req, res){ });  
router.put('/wip/:id/status', function(req, res){ });
```



Building APIs : DARRT : Representations

- Produce the requested format for resource responses

```
building/app-json-template.js
// plain JSON rerpresentor template
exports.template =
{
  format:"application/json",
  view:
  `
  {
    "<%=type%>":
    [
      <%var x=0;%>
      <%rtn.forEach(function(item){%>
        <%if(x!==0){%>,<%}%>
        {
          <%var y=0;%>
          <%for(var p in item){%>
            <%if(y!==0){%>,<%}%>
            "<%=p%>": "<%=helpers.stateValue(item[p],item,request,item[p])%>"
          <%y=1;%>
          <%}%>
        }
      <%x=1;%>
    <%});%>
  }
  `
}
```

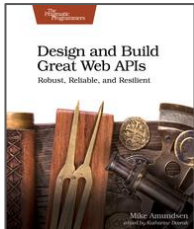


Building APIs : DARRT : Transitions

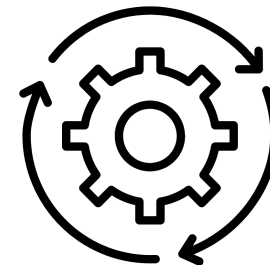
- The list of public actions as expressed in HTTP

```
building/add-account-transition.js
```

```
{
  id:"addAccount_{id}",
  name:"addAccount",
  href:"{fullhost}/wip/{id}/account",
  rel: "item edit-form onboarding",
  tags: "onboarding list item",
  title: "Add Account",
  method: "PUT",
  properties: [
    {name:"accountId",value:"{accountId}"},
    {name:"division",value:"{division}"},
    {name:"spendingLimit",value:"{spendingLimit}"},
    {name:"discountPercentage",value:"{discountPercentage}"}
  ]
}
```



Building APIs



- Sketching

- Experiment, throw away
- APIB

- Prototyping

- Explore, test
- OpenAPI

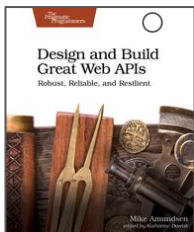
- Building

- Translate, repeat
- NodeJS/DARRT

```
63 lines (50 sloc) | 1.51 KB
1 FORMAT: 1A
2 HOST: http://polls.apibluprint.org/
3
4 # Credit Check Responses
5
6 Public API pro
7
8 ## Credit Check
9
10 Returns a list
11
12 ### List Past
13
14 + Response 200
15
16 {
17   "cre
18   {
19
20
21
22
23   },
24
25   {
26
```

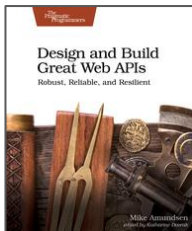
```
284 lines (248 sloc) | 6.63 KB
1 openapi: 3.0.0
2
3 #####
4 # Credit Check
5 # 2019-12-22
6 # mamund
7 #####
8
9 ## info sect
10 info:
11   version: '1.
12   title: 'Cred
13   description:
14
15 # Added by API
16 servers:
17   - descripti
18     url: https
19
20 ### tags ###
21 tags:
22   - name: cred
23     descripti
24
25 ## paths sect
26 paths:
```

```
building/resource-list.js
// *****
// public resources for the onboarding service
// *****
router.get('/', function(req,res){ });
router.post('/wip/', function(req,res){ });
router.get('/wip/', function(req,res){ });
router.get('/wip/filter/', function(req,res){ });
router.get('/wip/:id', function(req,res){ });
router.get('/wip/:id/company', function(req,res){ });
router.put('/wip/:id/company', function(req,res){ });
router.get('/wip/:id/account', function(req,res){ });
router.put('/wip/:id/account', function(req,res){ });
router.get('/wip/:id/activity', function(req,res){ });
router.put('/wip/:id/activity', function(req,res){ });
router.get('/wip/:id/status', function(req,res){ });
router.put('/wip/:id/status', function(req,res){ });
```



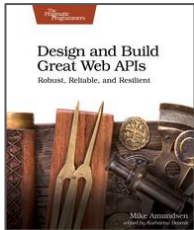


Releasing APIs



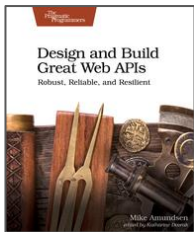
Releasing APIs

- Testing
- Security
- Deployment



Releasing APIs

- Testing
 - Testing Interface, not Code
 - Happy-path (200) & sad-path (400)
 - Simple Request Tests (SRTs)
 - Postman/Newman for BDD
- Security
- Deployment



Postman

File Edit View Help

New Import Runner My Workspace Invite

Filter

Launchpad GET Company Home company-heroku

History Collections APIs

+ New Collection Trash

BigCo Company API
15 requests

- happy_path
 - GET Company Home
 - GET Company List
 - POST Company Create
 - GET Company Read
 - PUT Company Update
 - PATCH Company Status
 - DEL Company Remove
- sad_path
 - GET Company Read (Bad Id)
 - DEL Company Remove (valid)
 - POST Company Create (missing email)
 - POST Company Create (missing comp...)
 - POST Company Create (missing status)

Company Home

GET {{server}}/

Params Authorization Headers (7) Body Pre-request Script

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (9) Test Results (8/8)

All	Passed	Skipped	Failed
PASS	Status is 200		
PASS	Header content-type contains application/forms+json		
PASS	Meta property title contains BigCo Company Records		
PASS	Meta property release contains 1.0.0		
PASS	Meta property author contains Amundsen		

EDIT COLLECTION

Name

BigCo Company API

Description Authorization Pre-request Scripts Tests Variables

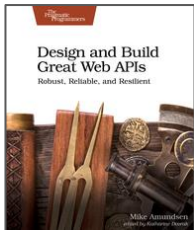
This script will execute before every request in this collection. [Learn more about Postman's execution order.](#)

```

1 // SHARED FUNCTIONS for forms+json
2 // Utilities available for all tests in this collection
3 pm.globals.set('loadUtils', function loadUtils() {
4   let utils = {};
5   let obj = '';
6
7   // set shared object
8   utils.setObject = function(args) {
9     obj = args.object||'';
10  };
11
12  // check status
13  utils.checkStatus = function(value) {
14    pm.test('Status is ' + value, function() {
15      pm.expect(pm.response.code).to.equal(value);
16    });
17  };
18
19  // check header
20  utils.checkHeader = function(args) {
21    pm.test('Header ' + args.name + ' contains ' + args.value, function() {
22      var hdr = pm.response.headers.get(args.name);
23      pm.expect(hdr).to.include(args.value);
24    });
25  };
26
27  // check metadata
28  utils.checkMetaProperty = function(args) {
29    pm.test('Meta property ' + args.name + ' contains ' + args.value, function() {
30      var meta = body[obj].metadata.find(x => x.name === args.name);
31      pm.expect(meta.value).to.include(args.value);
32    });
33  };

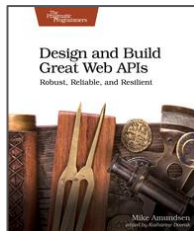
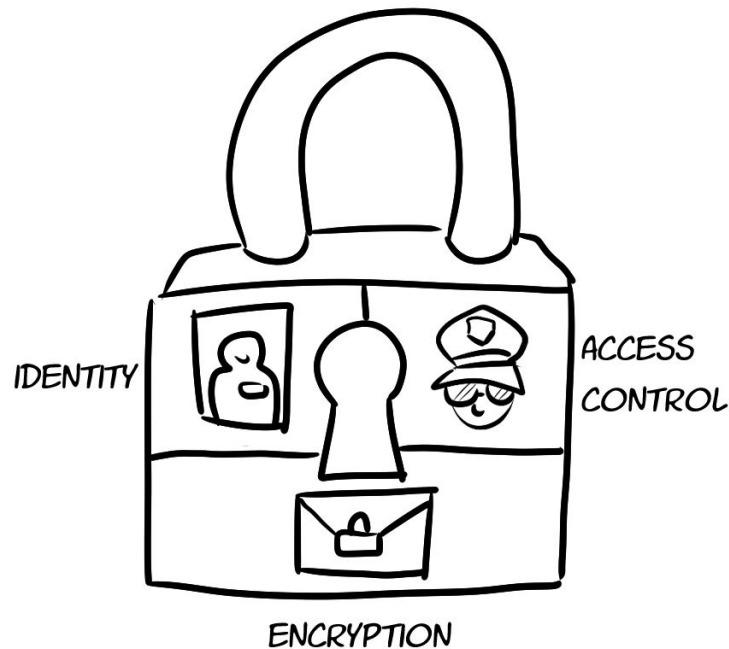
```

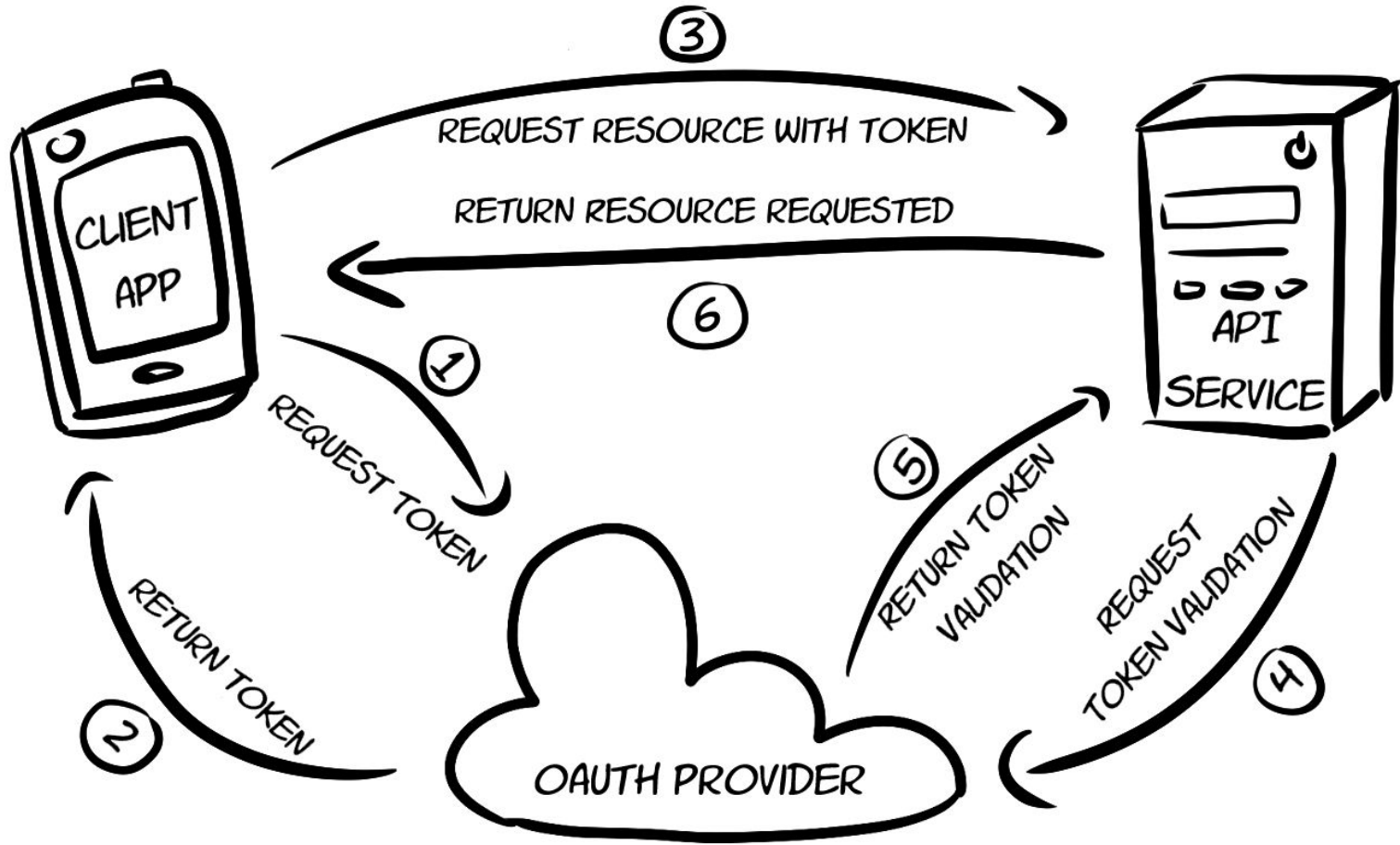
Cancel Update



Releasing APIs

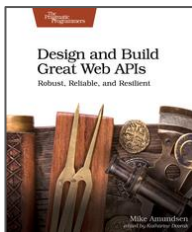
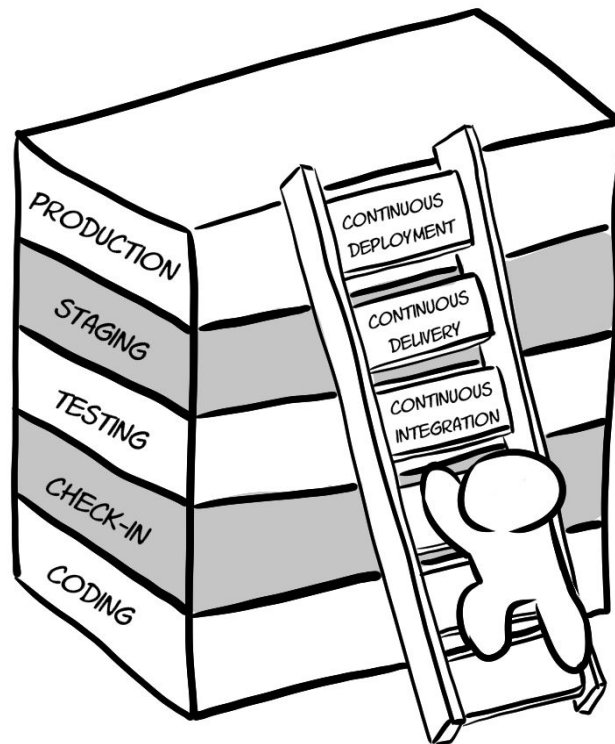
- Testing
- Security
 - Encryption, Identity, Access Control
 - M2M Authorization w/ JWTs
 - Auth0 as provider
- Deployment





Releasing APIs

- Testing
- Security
- Deployment
 - Integration, delivery, deployment
 - Automation improves safety
 - Heroku via git push



```
remote:
remote: -----> Build succeeded!
remote:   └─ ejs@2.3.3
remote:   └─ express@4.13.3
remote:
remote:
remote: -----> Discovering process types
remote:   Procfile declares types -> v
remote:
remote: -----> Compressing...
remote:   Done: 10.2M
remote: -----> Launching...
remote:   Released v4
remote:   https://tranquil-hollows-728
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/tranquil-hollows-
763c36e..f555ce0 master -> master
~/Code/tranquil-hollows-72848
```

```
Running pre-deployment tests...

BigCo Company Tests
=====
Wed May 27 19:55:24 EDT 2020
Running local environment...
Initializing...
rm: cannot remove 'newman/*': No such file or directory
Pulling postman data...

> company@1.0.0 dev /home/mca/projects/api-tool-kit
> nodemon index

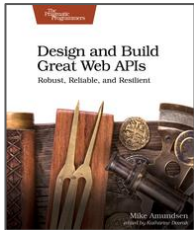
[nodemon] 2.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index index.js`
Running tests...
listening on port 8484!
Time: 1590623727242 : localhost:8484/ : GET : {}
Time: 1590623727583 : localhost:8484/list/ : GET
Time: 1590623727961 : localhost:8484/ : POST : {
om","telephone":"123-456-7890","status":"pending"}
```

```
Deploying to production...
Everything up-to-date
Running post-deployment tests...

BigCo Company Tests
=====
Wed May 27 19:55:33 EDT 2020
Running remote environment...
Initializing...
Pulling postman data...
[nodemon] restarting due to changes...
[nodemon] starting `node index index.js`
listening on port 8484!
[nodemon] restarting due to changes...
Running tests...
[nodemon] starting `node index index.js`
listening on port 8484!
One or more tests failed!

Test run completed and saved to test-output.txt.
*** POST-DEPLOY TESTS FAILED - job cancelled. ***

mca@penguin:~/projects/api-tool-kit/company$
```



Releasing APIs



- Testing
 - Test interface and behavior
 - Postman/Newman
- Security
 - APIs use client_credentials w/ JWTs
 - Auth0 as a provider
- Deployment
 - Automate for safety
 - Heroku git push

The screenshot shows the Postman interface with a collection named 'BigCo Company API'. A test for the 'GET /company/home' endpoint is selected. The test body contains a request for a resource with a token. Overlaid on the screenshot is a diagram of the OAuth2 flow:

- 1. CLIENT APP requests a token from OAUTH.
- 2. OAUTH returns a token to the CLIENT APP.
- 3. CLIENT APP requests a resource with the token from the server.
- 4. The server returns the resource to the CLIENT APP.

Below the diagram is a terminal window showing the deployment process:

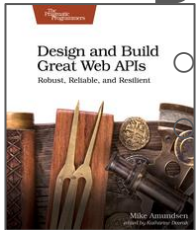
```
Deploying to production...
Everything up-to-date
Running post-deployment tests...

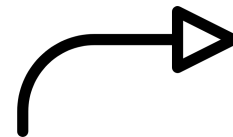
BigCo Company Tests
=====
Wed May 27 19:55:33 EDT 2020
Running remote environment...
Initializing...
Pulling postman data...
[nodeemon] restarting due to changes...
[nodeemon] starting `node index index.js`
listening on port 8484!
[nodeemon] restarting due to changes...
Running tests...
[nodeemon] starting `node index index.js`
listening on port 8484!
One or more tests failed!

Test run completed and saved to test-output.txt.

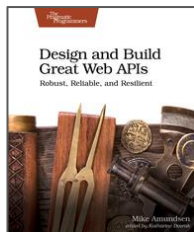
*** POST-DEPLOY TESTS FAILED - job cancelled. ***

mca@penguin:~/projects/api-tool-kit/company$
```





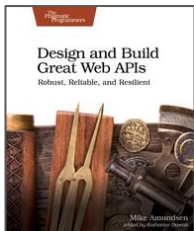
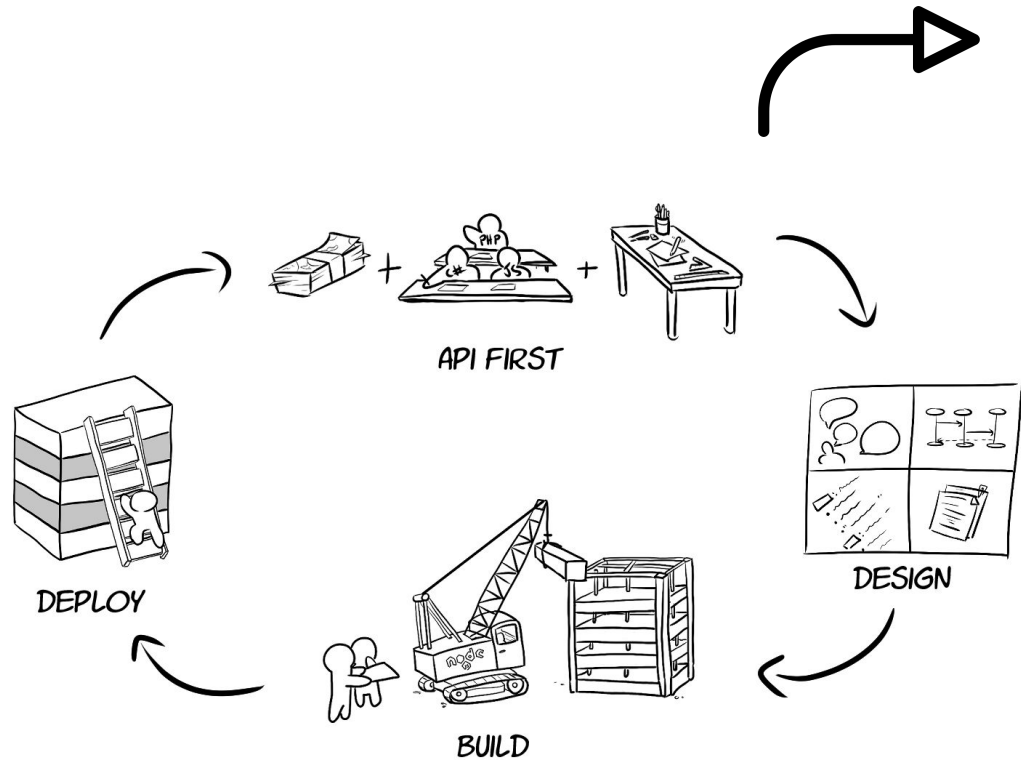
And Then...



copyright © 2020 by amundsen.com, inc. -- all rights reserved

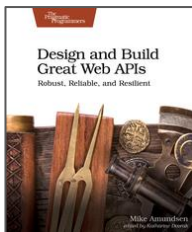
Modifying APIs

- Update Principles
- Design Updates
- Test Updates
- Release Updates
- API Shutdown



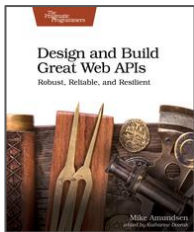
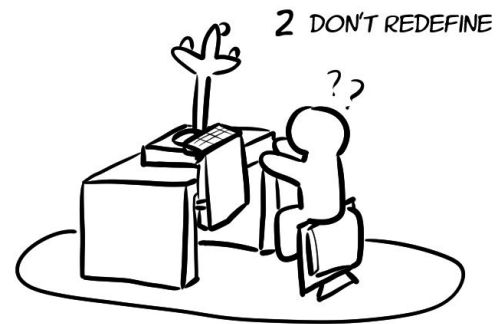
Modifying APIs

- Update Principles
 - First, do no harm
 - Fork your API
 - Know when to say "No"
- Design Updates
- Test Updates
- Release Updates
- API Shutdown



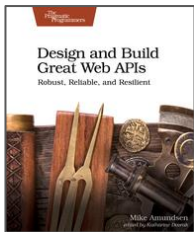
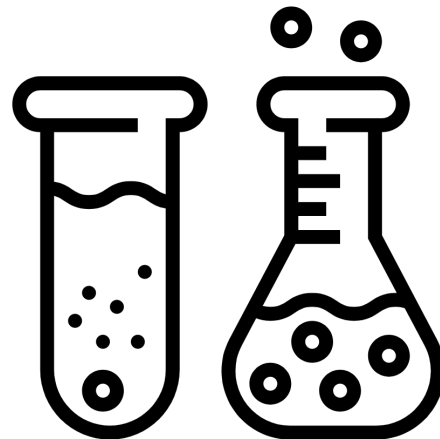
Modifying APIs

- Update Principles
- Design Updates
 - Take nothing away
 - Don't redefine
 - Additions are optional
- Test Updates
- Release Updates
- API Shutdown



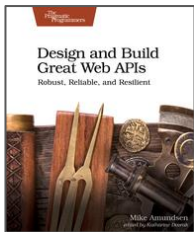
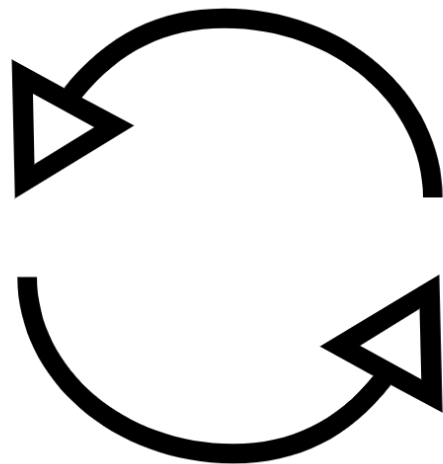
Modifying APIs

- Update Principles
- Design Updates
- Test Updates
 - Use all existing tests
 - Add new tests for each release
- Release Updates
- API Shutdown



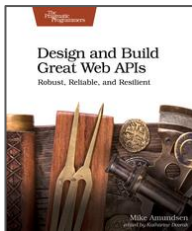
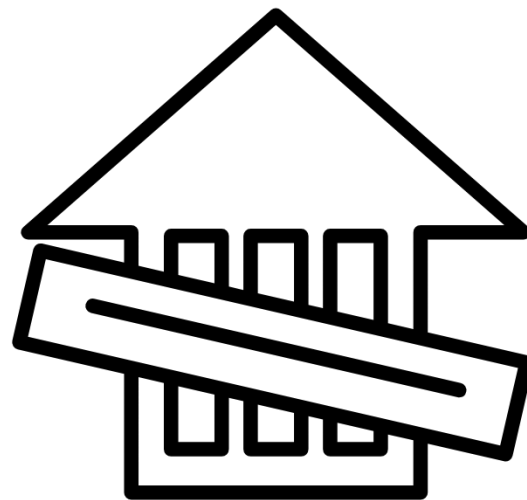
Modifying APIs

- Update Principles
- Design Updates
- Test Updates
- Release Updates
 - Reversibility/Re-entry First
 - Side-by-Side Releases
 - Overwriting Releases
- API Shutdown



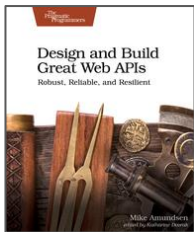
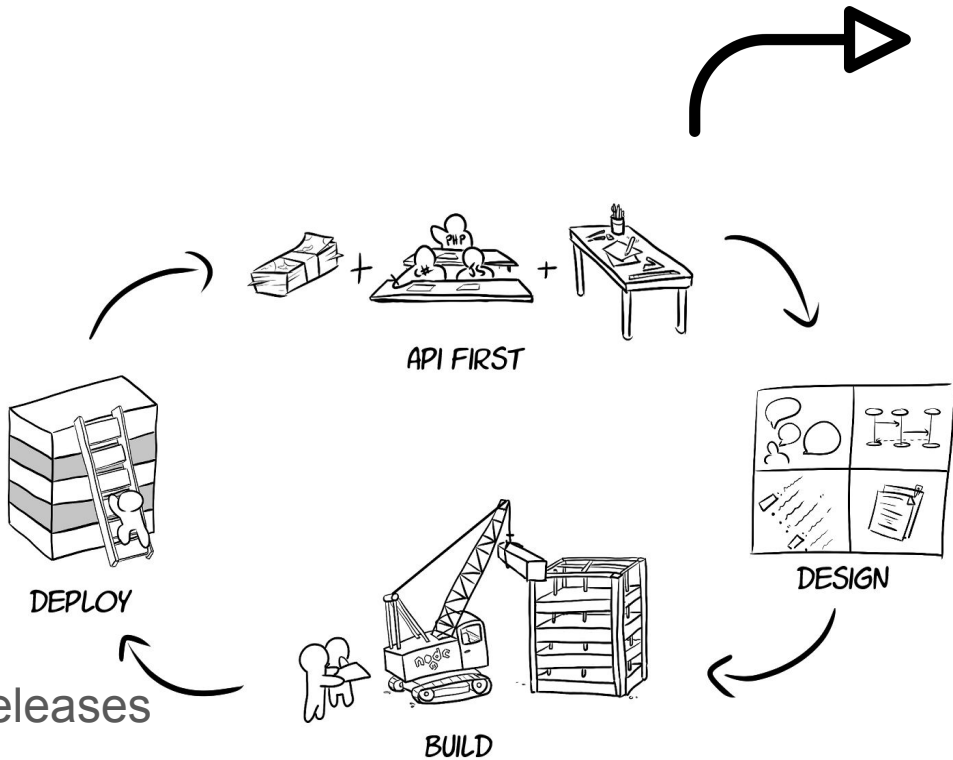
Modifying APIs

- Update Principles
- Design Updates
- Test Updates
- Release Updates
- API Shutdown
 - Place Code in Public Domain
 - Open Source the Interface
 - Recoverable Data
 - Mark the API 410 Gone (w/ a pointer)

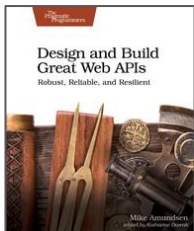


Modifying APIs

- Update Principles
 - First, Do no harm
- Design Updates
 - No breaking changes
- Test Updates
 - Use all the old tests
- Release Updates
 - Favor Side-by-Side Releases
- API Shutdown
 - Responsible shutdowns



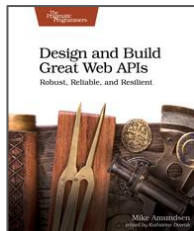
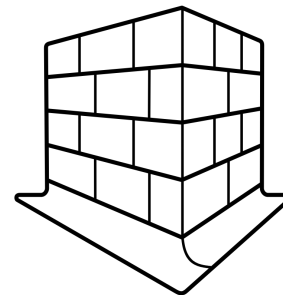
So....



copyright © 2020 by amundsen.com, inc. -- all rights reserved

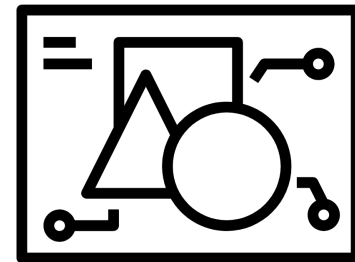
Foundations

- API First
 - Solving business problems for people
- HTTP, Web, & REST
 - Protocol, practice, style



Designing APIs

- Model
 - API Story
- Design
 - API Diagram
- Describe
 - API Description



13 Lines (9 sloc) | 1.09 KB

Raw Blame History

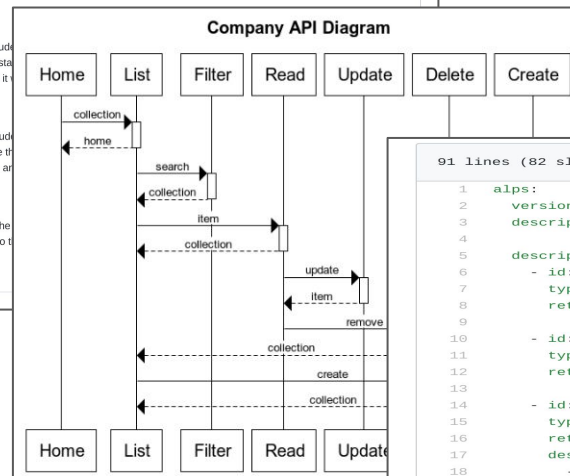
Company Story at BigCo, Inc.

Purpose
We keep track of companies for BigCo, Inc.

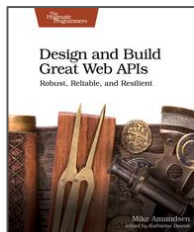
Data
Data we include in a company record include telephone, and email. Each record has a start date the record was created and the last date/time it was updated.

Actions
Typical work on the company records include adding new records, updating existing records, and deleting records. You can also update the status of a record for filtering by status, by country, by state, and by province.

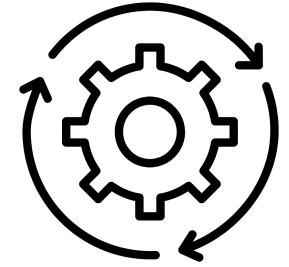
Processing
Each company has a unique identifier in the system. The identifier is the name and date the company was added to the system.



```
91 lines (82 sloc) | 2 KB
1  alps:
2  version: '1.0'
3  description: ALPS document for BigCo Comp
4
5  descriptors:
6  - id: home
7  type: safe
8  returns: '#company'
9
10 - id: listCompanies
11 type: safe
12 returns: '#company'
13
14 - id: filterCompanies
15 type: safe
16 returns: '#company'
17 descriptors:
18 - href: '#companyName'
19 - href: '#country'
20 - href: '#status'
21 - href: '#stateProvince'
22
23 - id: readCompany
24 type: safe
25 returns: '#company'
26 descriptors:
27 - href: '#id'
```



Building APIs



- Sketching

- Explore, throw away
- APIB

- Prototyping

- Details, test
- OpenAPI

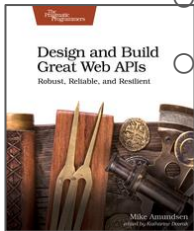
- Building

○ Translate, repeat
○ NodeJS/DARRT

```
63 lines (50 sloc) | 1.51 KB
1 FORMAT: 1A
2 HOST: http://polls.apiblueprint.org/
3
4 # Credit Check Responses
5
6 Public API pro
7
8 ## Credit Check
9
10 Returns a list
11
12 ### List Past
13
14 + Response 200
15
16 {
17   "cred
18   {
19
20
21
22
23   },
24   {
25
26
```

```
284 lines (248 sloc) | 6.63 KB
1 openapi: 3.0.0
2
3 #####
4 # Credit Check
5 # 2019-12-22
6 # mamund
7 #####
8
9 ## info secti
10 info:
11   version: '1.
12   title: 'Cred
13   description:
14
15 # Added by API
16 servers:
17   - descripti
18     url: https
19
20 ### tags ###
21 tags:
22   - name: cred
23     descripti
24
25 ## paths sect
26 paths:
```

```
building/resource-list.js
// *****
// public resources for the onboarding service
// *****
router.get('/', function(req,res){ });
router.post('/wip/', function(req,res){ });
router.get('/wip/', function(req,res){ });
router.get('/wip/filter/', function(req,res){ });
router.get('/wip/:id', function(req,res){ });
router.get('/wip/:id/company', function(req,res){ });
router.put('/wip/:id/company', function(req,res){ });
router.get('/wip/:id/account', function(req,res){ });
router.put('/wip/:id/account', function(req,res){ });
router.get('/wip/:id/activity', function(req,res){ });
router.put('/wip/:id/activity', function(req,res){ });
router.get('/wip/:id/status', function(req,res){ });
router.put('/wip/:id/status', function(req,res){ });
```

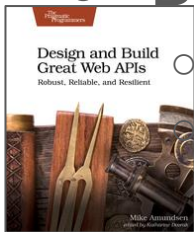


Releasing APIs



- Testing
 - Test interface and behavior
 - Postman/Newman
- Security
 - APIs use client_credentials w/ JWTs
 - Auth0 as a provider
- Deployment

Automate for safety
Heroku git push



The diagram illustrates the OAuth2 flow:

1. CLIENT APP requests a token from the OAUTH provider.
2. OAUTH provider returns the token to the CLIENT APP.
3. CLIENT APP uses the token to request a resource from the server.
4. Server returns the resource to the CLIENT APP.

```
graph TD
    CA[CLIENT APP] -- "REQUEST TOKEN" --> O[OAUTH]
    O -- "RETURN TOKEN" --> CA
    CA -- "REQUEST RESOURCE WITH TOKEN" --> S[SERVER]
    S -- "RETURN RESOURCE" --> CA
```

Terminal output:

```
Deploying to production...
Everything up-to-date
Running post-deployment tests...

BigCo Company Tests
=====
Wed May 27 19:55:33 EDT 2020
Running remote environment...
Initializing...
Pulling postman data...
[nodemon] restarting due to changes...
[nodemon] starting `node index index.js`
listening on port 8484!
[nodemon] restarting due to changes...
Running tests...
[nodemon] starting `node index index.js`
listening on port 8484!
One or more tests failed!

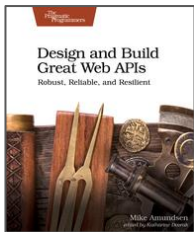
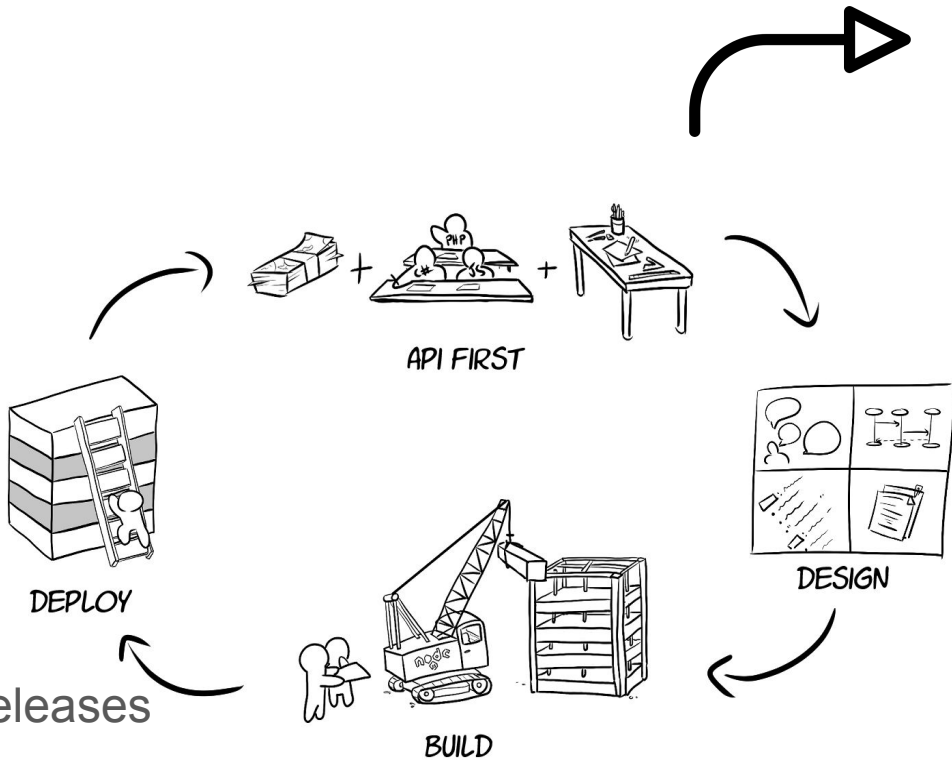
Test run completed and saved to test-output.txt.

*** POST-DEPLOY TESTS FAILED - job cancelled. ***

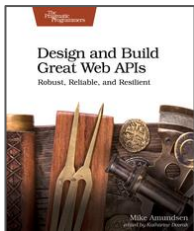
mca@penguin:~/projects/api-tool-kit/company$
```

Modifying APIs

- Update Principles
 - First, Do no harm
- Design Updates
 - No breaking changes
- Test Updates
 - Use all the old tests
- Release Updates
 - Favor Side-by-Side Releases
- API Shutdown
 - Responsible shutdowns



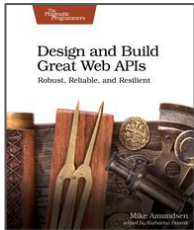
And, don't forget ...



What is API-First?

*"API-first design means **identifying** and/or defining key actors and personas, **determining** what those actors and personas expect to be able to do with APIs"*

-- Kas Thomas, 2009



Design and Build Great APIs

<http://g.mamund.com/2020-interface>

@mamund

Mike Amundsen

[youtube.com/mamund](https://www.youtube.com/mamund)

