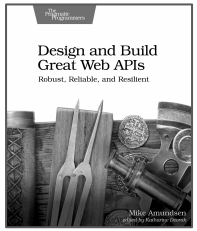
Design and Build Great Web APIs

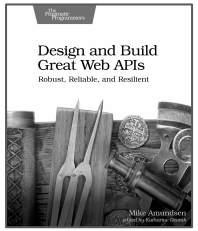
Building

@mamund
Mike Amundsen
training.amundsen.com

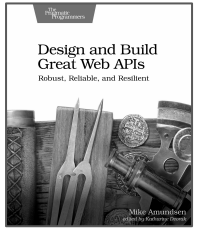


Building

- Sketching your API
  - Quick experiments
- Prototyping you API
  - Detailed exploration
- Building your API
  - Heavy commitment



Sketching your API



sketch

/skeCH/

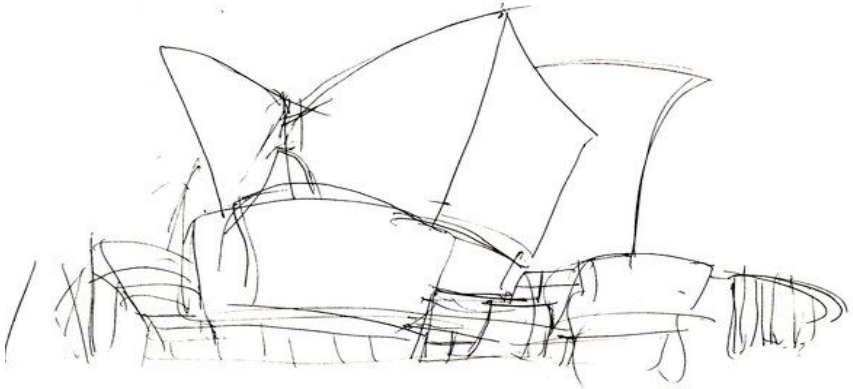
noun

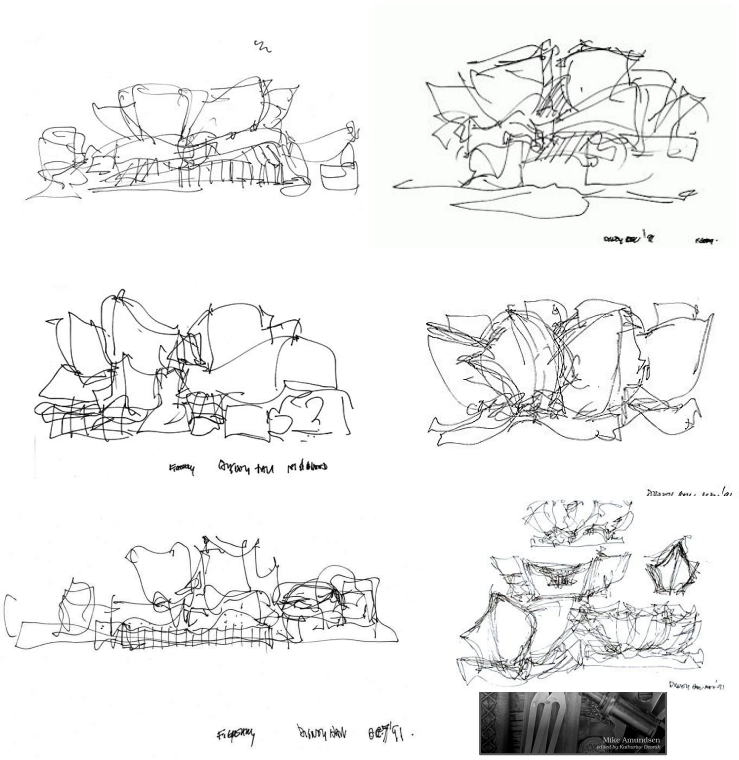
1. a rough or unfinished drawing or painting, often made to assist in making a more finished picture.
   "a charcoal sketch"
   synonyms: (preliminary) drawing, outline; More

Sketching APIs

- What is Sketching?
- Sketching APIs with **Blueprint**

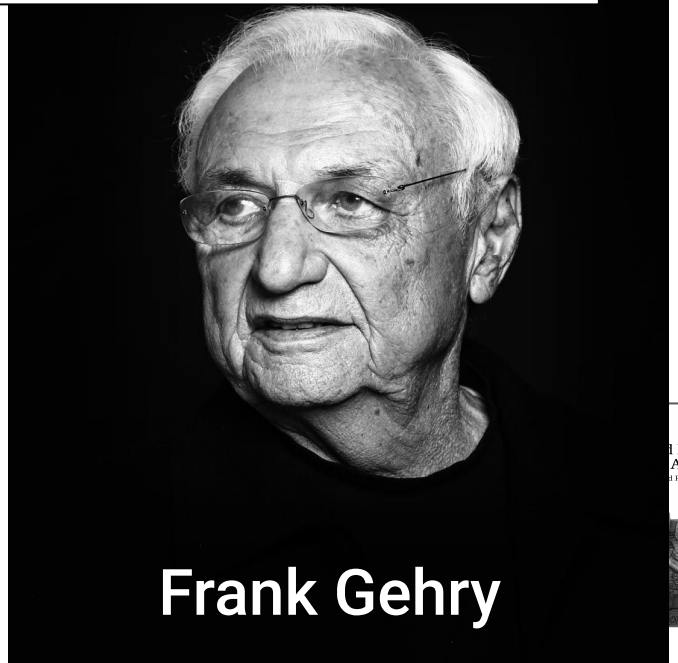
Frank Gehry Sketches

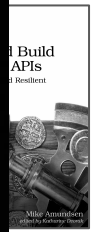




Frank Gehry Sketches

An architect is given a program, budget, place, and schedule. Sometimes the end product rises to art





Sketching APIs

- Sketches are terse, rough drawings
- They give the general idea of a thing but lack important details.
- Usually, one can glean the basics from a sketch but
- Sketches usually are just explorations of ideas, not fully-formed items.

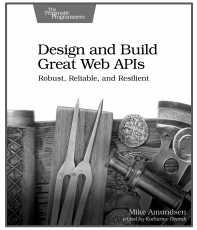
Sketching APIs

- Create a sketch (using **Blueprint**).
- Show it to others (devs, stakeholders) and get their feedback.
- If possible use simple API consumer tools (curl, NodeJS, etc.) to test.
- Continue to modify the simple sketches as needed

Sketching APIs -- Using Blueprint

- Created in 2013 by Jakub Nesetril
- Focused on quickly mocking API request/response
- Based on Markdown
- Sold to Oracle in 2017

apiary



Sketching APIs -- Using Blueprint

- No download needed

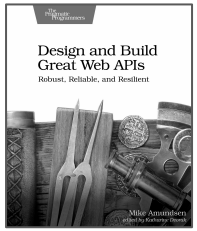
  https://app.apiary.io/onboardingapi/editor

- Documentation

  https://help.apiary.io/tools/apiary-editor/

- Create Account (optional)

  https://login.apiary.io/



Sketching APIs -- Using Blueprint

- Write (or copy/paste) APIB doc into Editor
- Copy/Paste into local doc to save to disk

Onboarding API
Mike Amundsen • onboardingapi

Documentation Inspector Editor Tests

Link this Project to GitHub A Valid document Preview On Save

```
FORMAT: 1A
HOST: http://polls.apiblueprint.org/

# Onboarding API

Polls is a simple API allowing consumers to view polls and vote in t

## Questions Collection [/questions]

### List All Questions [GET]

+ Response 200 (application/json)

        [
            {
                "question": "Favourite programming language?",
```

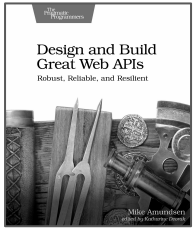
Introduction

Onboarding API

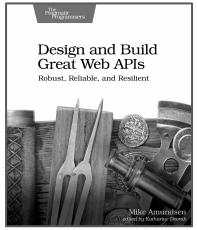
INTRODUCTION

Polls is a simple API allowing consumers to view polls and vote in them

Sketches are made to be thrown away.



Prototyping your API

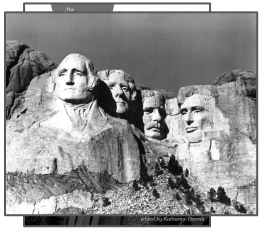


pro·to·type

/ˈprōdəˌtīp/

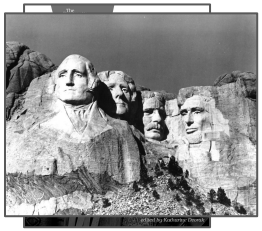
noun

1. a first, typical or preliminary model of something, especially a machine, from which other forms are developed or copied.
   "the firm is testing a prototype of the weapon"

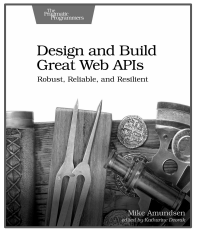


Prototype APIs

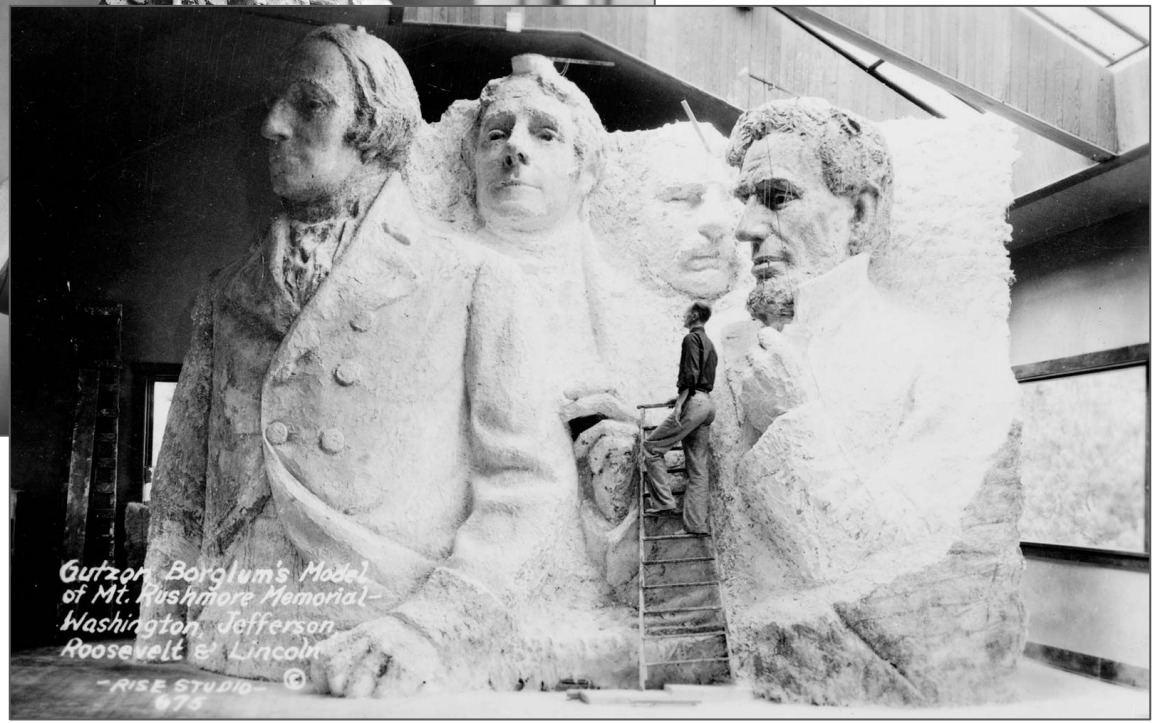
- What is Prototyping?
- Prototyping with **OpenAPI**

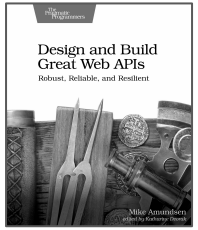


Borglum's Prototypes

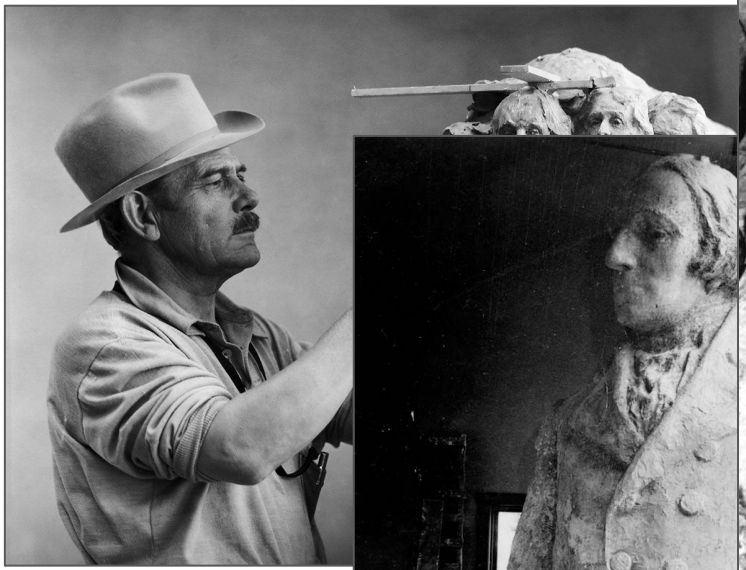


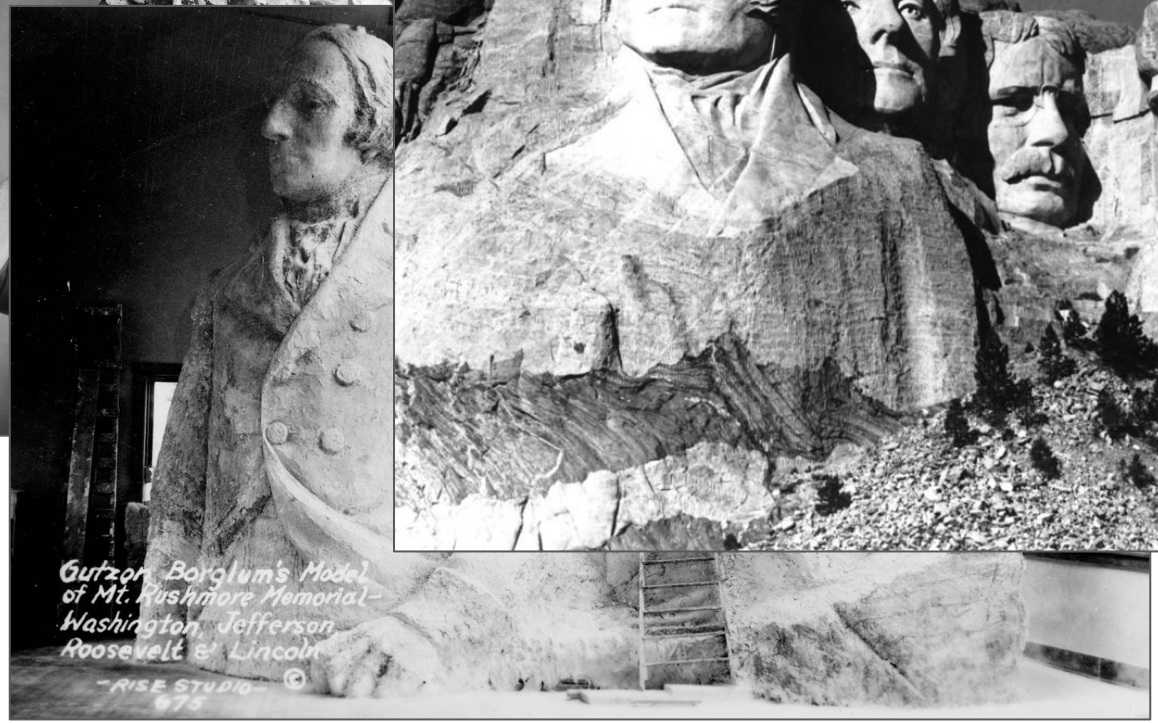
Borglum's Prototypes

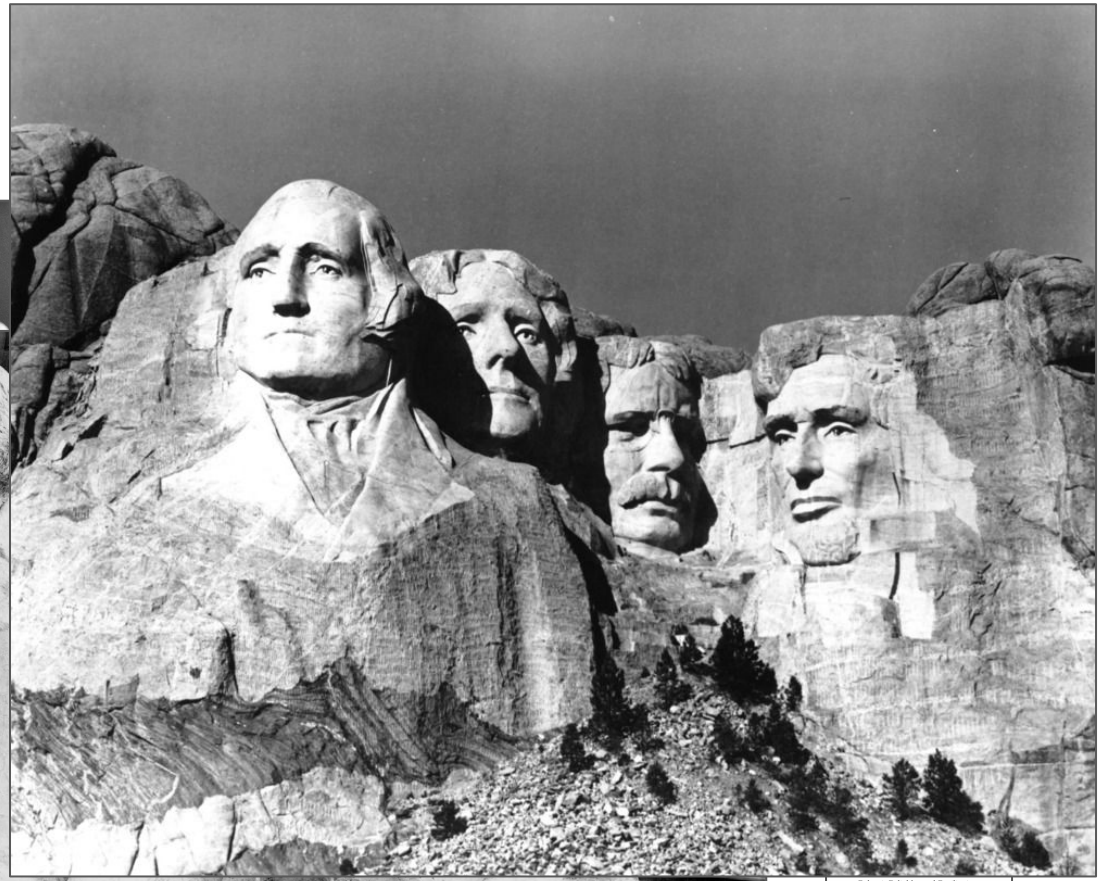


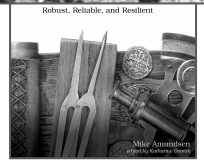


Borglum's Prototypes



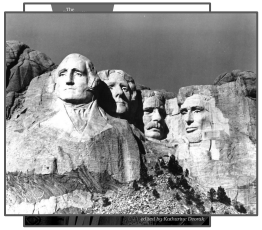






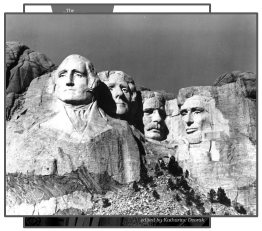
Prototyping APIs

- Prototypes look like the real thing, but are not. They’re "fakes."
- They let you work up something with all the details of a real API, but without the actual functionality behind it.
- They’re an inexpensive way to work out the details
- Use them to discover challenges before you go into production.



Prototyping APIs

- Select a likely API sketch
- Create a prototype of it (using **OpenAPI**).
- Show it to others (devs, stakeholders) and get their feedback.
- If possible, use production-level API consumer tools to test.
- Continue to modify the prototypes as needed



Prototype APIs -- Using Swagger

- No download needed

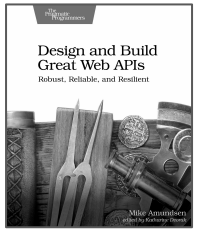
  https://editor.swagger.io/

- Documentation

  https://swagger.io/docs/

- Create an account (optional)

  https://app.swaggerhub.com



Prototype APIs -- Using Swagger

- Copy/Paste into Swagger Editor

OPENAPI INITIATIVE

Swagger Editor
Supported by SMARTBEAR

File ▾ Edit ▾ Generate Server ▾ Generate Client ▾

```yaml
swagger: '2.0'
info:
  title: Onboarding API
  version: ''
  description: Polls is a simple API allowing consumers to view
    polls and vote in them.
host: polls.apiblueprint.org
basePath: /
schemes:
  - http
paths:
  /questions:
    get:
      responses:
        '200':
          description: OK
          headers: {}
          examples:
            application/json:
              - question: Favourite programming language?
                published_at: '2015-08-05T08:40:51.620Z'
```

Onboarding API

[Base URL: polls.apiblueprint.org/]

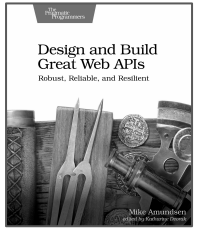
Polls is a simple API allowing consumers to view polls and vote in them.

Schemes

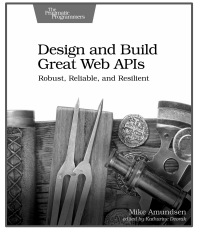
HTTP

Authorize

Prototypes are made to be tested.



Building your API

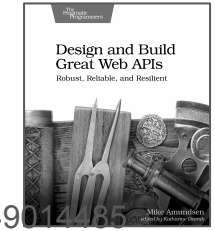


build

/bild/

verb

1. construct (something, typically something large) by putting parts or material together over a period of time.
   "the factory was built in 1936"
   synonyms: construct, erect, put up, assemble; More

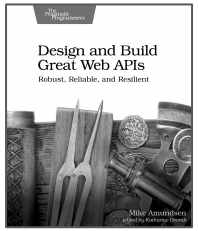


Building APIs

- API builds are the real thing
- Production-ready, access-controlled, resilient, scalable.
- Building the production implementation means
  - Working out all the kinks
  - Supporting all the use-cases identified during the sketch and prototype phases.

Building APIs : DARRT

- Simple process for publishing running interfaces
- Data
- Actions
- Resources
- Representations
- Transitions



Building APIs : DARRT : Data

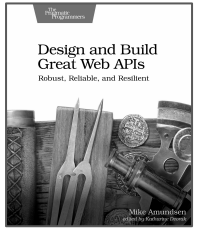
- The state properties to pass in messages
  - properties, requireds, enums, defaults

```
// 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'
];
```



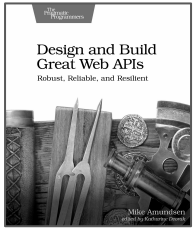
Building APIs : DARRT : Data

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

```
// this service's message properties

reated','dateUpdated',
'companyId','companyName','streetAddress','city','stateProvince',
'postalCode','country','telephone','email',
'accountId','division','spendingLimit','discountPercentage',
'activityId','activityType','dateScheduled','notes'
];
```

```
// required properties
exports.reqd = ['id','status'];
```



Building APIs : DARRT : Data

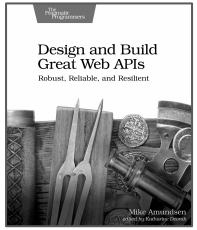
- The state properties to pass in messages
  - `properties, requireds, enums, defaults`

```
// this service's message properties

reated','dateUpdated',
'companyId',
'postalCode'
'accountId',
'activityId'
];
```

```
// required properties
exports.reqd = ['id','status'];
```

```
// 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, requireds, enums, defaults`

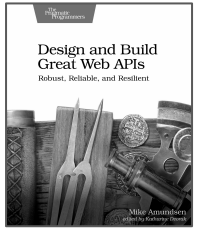
```
// this service's message properties

// required properties
exports.reqd = ['id','status'];
```

```
reated','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"}
];
```

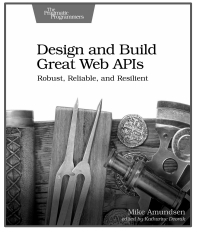


Building APIs : DARRT : Actions

- The actual operations for the interface
  - appovePayroll, 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"});
    }
  });
}
```



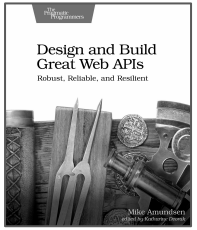
Building APIs : DARRT : Resources

- The 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){ });
```

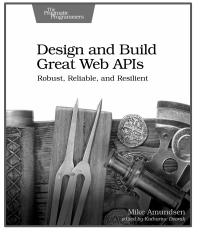


Building APIs : DARRT : Representations

- The format/media-type of 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;%>
    <%});%>
   ]
  }
  `
}
```

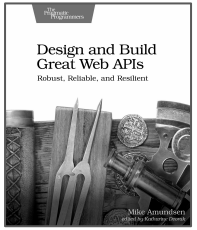


Building APIs : DARRT : Transitions

- The public expression of actions

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}"}
  ]
}
```



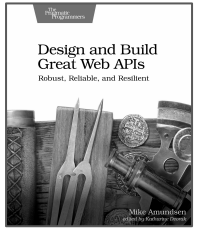
Building APIs : Putting it all together

- Use **`nodemon`** when testing your service locally

building/test-nodemon.txt

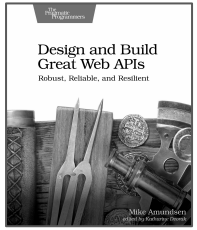
```
> onboarding@1.0.0 dev /building/all-together/onboarding
> nodemon index

[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`
listening on port 8080!
```



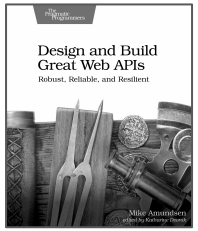
Production APIs are made last.

Building Exercise

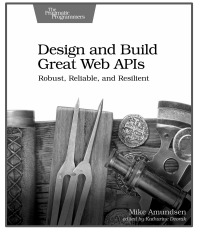


Building Exercise

- Get `https://github.com/mamund/api-starter` project
- Work the DARRT model
  - darrt/data.js
  - darrt/actions.js
  - darrt/resources
  - (darrt/representations)
  - darrt/transitions
- Run with **`nodemon`**



Summary

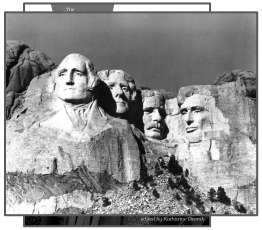

The Pragmatic Programmers
Design and Build Great Web APIs
Robust, Reliable, and Resilient
Mike Amundsen
edited by Katharine Dvorak

Sketching APIs

- Sketches are terse, rough drawings
- They give the general idea of a thing but lack important details.
- Usually, one can glean the basics from a sketch but
- Sketches usually are just explorations of ideas, not fully-formed items.

Prototyping APIs

- Prototypes look like the real thing, but are not. They’re "fakes."
- They let you work up something with all the details of a real API, but without the actual functionality behind it.
- They’re an inexpensive way to work out the details
- Use them to discover challenges before you go into production.

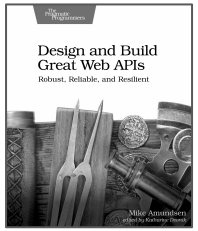


Building APIs

- API builds are the real thing
- Production-ready, access-controlled, resilient, scalable.
- Building the production implementation means
  - Working out all the kinks
  - Supporting all the use-cases identified during the sketch and prototype phases.

Building

- Sketching your API
  - Sketches are made to be thrown away
- Prototyping you API
  - Prototypes are made to be tested
- Building your API
  - Builds are forever



Design and Build Great Web APIs

Building

@mamund
Mike Amundsen
training.amundsen.com