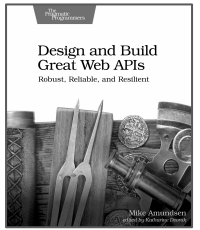
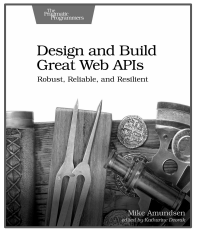
Design and Build Great Web APIs

Summary

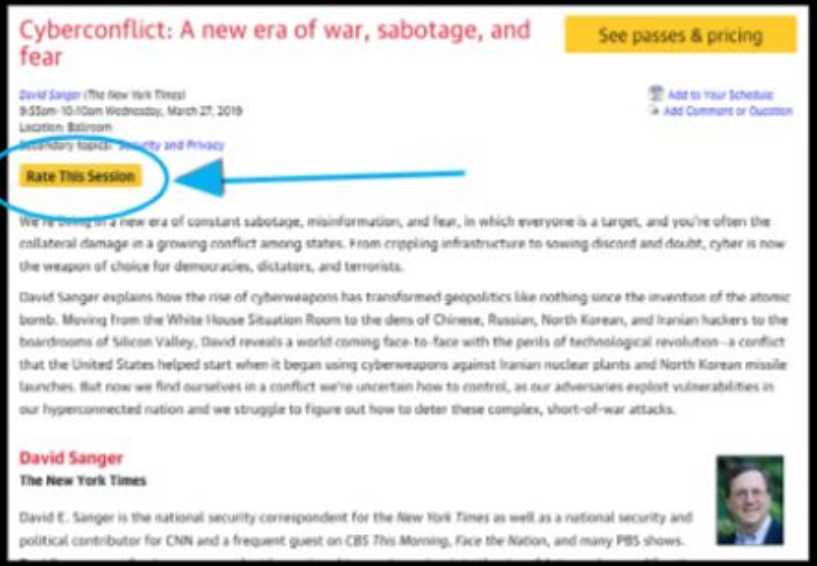
@mamund
Mike Amundsen
training.amundsen.com



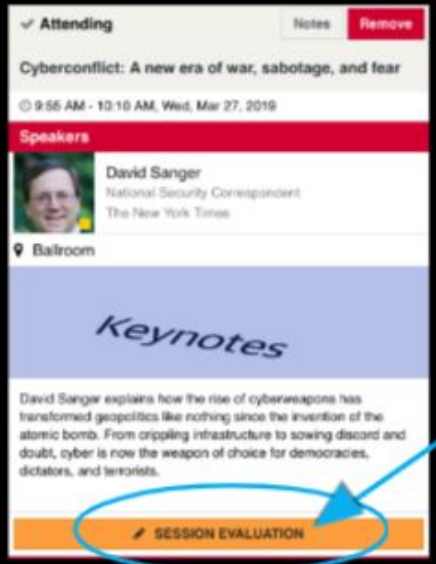
Don't forget your 25% discount & your laptop stickers!



Rate today's session



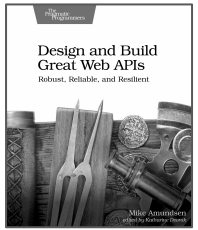
Session page on conference website



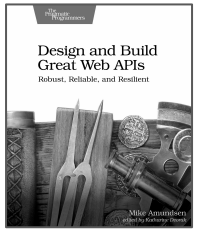
O'Reilly Events App

Design and Build Great Web APIs

- Principles
  - *Install & Setup*
- Designing
  - *Profile Exercise (ALPS yaml file)*
- Building
  - *DARRT Exercise (NodeJS & npm)*
- Releasing
  - *Deployment Exercise (Heroku CLI)*



Principles

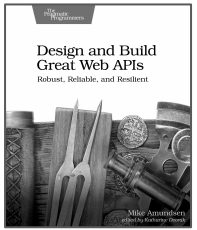

Design and Build
Great Web APIs
Robust, Reliable, and Resilient
Mike Amundsen

Principles : 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."

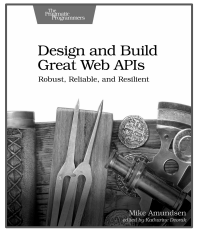
-- Kas Thomas, 2009

http://asserttrue.blogspot.com/2009/04/api-first-design.html



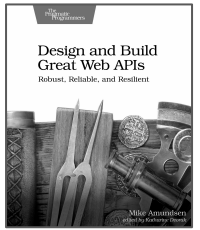
Principles : HTTP, the Web, and REST

- HTTP is a protocol
- The Web is a set of common practices
- REST is a specific style



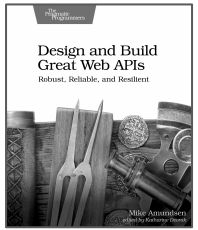
Principles : Interactions

- Norman's Action Lifecycle
- Request - Parse - Wait (RPW Loop)
- Modeling Interactions

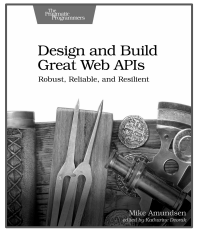


Principles

- API-First
  - Solve your API consumer's problem, not yours
- HTTP, the Web, and REST
  - Rely upon standards, common practices, & style
- Modeling Interactions
  - Use loops to allow API consumers to control interactions



Designing APIs



Document the Story

- Interview each stakeholder
  - Record their references to data, actions, processing, purpose
- Use plain language
  - Speak in the language of th stakeholder, not the developer
- Document each story *separately*
  - Share w/ each stakeholder for validation
- Write composite story to cover all stakeholders
  - Share w/ the group and work out details

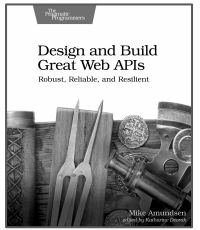
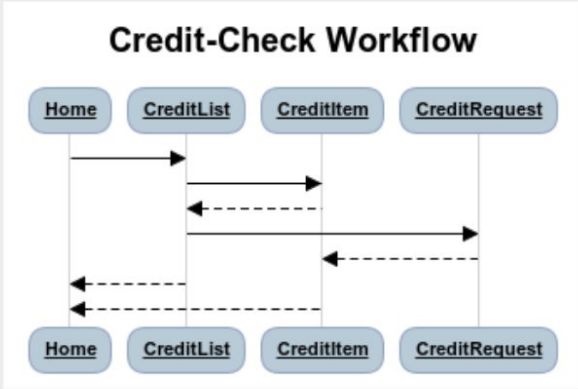


Diagram the Flow : websequencediagrams.com

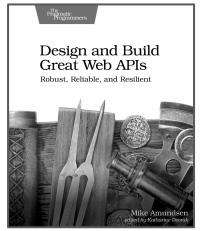
File Edit View Style Help Account

```
1  Title Credit-Check Workflow
2
3  Home -> CreditList:
4  CreditList -> CreditItem:
5  CreditItem --> CreditList:
6  CreditList -> CreditRequest:
7  CreditRequest --> CreditItem:
8  CreditList --> Home:
9  CreditItem --> Home:
10
```



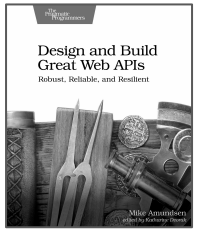
Describe the API : ALPS

- Application-Level Profile Semantics
  - Amundsen-Richardson-Foster (2011)
- Identifies all interface properties
  - Id, familyName, givenName, telephone, etc.
- Identifies all interface actions
  - saveCompany, setStatus, approvePayroll, etc.
- Does not include implementation details
  - URLs, schemas, methods, response codes, etc.

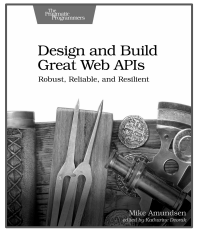


Designing

- Document the Story
  - Purpose, Actions, & Data
- Diagram the Flow
  - Sequence diagrams are not HTTP
- Describe the API
  - Before OpenAPI, AsyncAPI, protobuf and SDL



Building APIs

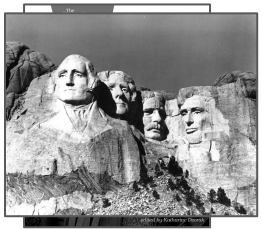


Sketching APIs

- 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.

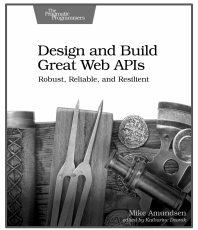


Building APIs

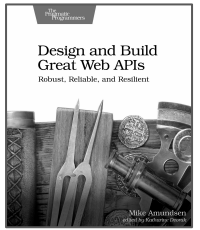
- API builds are the real thing
- Production-ready, access-controlled, resilient, scalable.
- Building the production implementation means
  - Working out all the kinks
  - Supporting all the use-cases identified during the sketch and prototype phases.

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



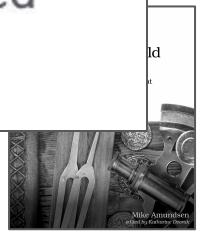
Releasing APIs



Testing APIs : BDD

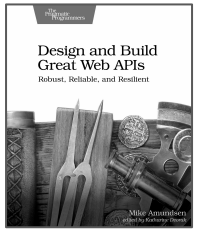
- Behavior-Driven Development (2006)
- Dan North, Thoughtworks
- Outside-in

> Dan North, the developer of the BDD, described it as: "...a second-generation, outside-in, pull-based, multiple-stakeholder, multiple-scale, high-automation, agile methodology. It describes a cycle of interactions with well-defined outputs, resulting in the delivery of working, tested software that matters."



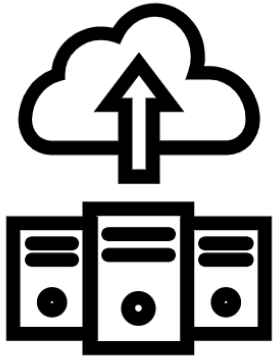
Securing APIs - Security Basics

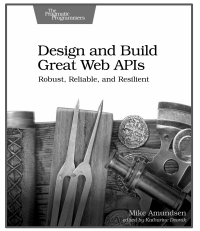
- Authentication (Identity)
- Authorization (Access Control)
- TLS/HTTPS (message encoding)
- Encryption (field-level encoding)



Deploying APIs - Challenges

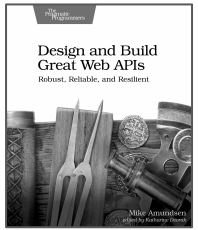
- Deploying your app can be complicated
- Compatibility
  - Hardware
  - OS
  - Platform
  - Framework
  - Dependencies



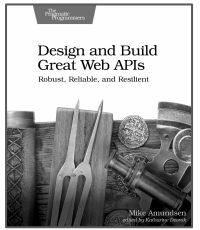


Releasing

- Testing
  - From request lists to BDD
- Securing
  - Identity and Access Control
- Deploying
  - Automation is your friend



Thanks!


Design and Build
Great Web APIs
Robust, Reliable, and Resilient
Mike Amundsen

Design and Build Great Web APIs

Summary

@mamund
Mike Amundsen
training.amundsen.com