modern java

in the clouds

airhacks.live

"It's not work if you like it"
…so I never worked. #java

airhacks.io => online video courses

airhacks.news => stay tuned

airhacks.fm => podcast

adambien.blog

airhacks.live

airhacks.TV

airhacks.live

NEW online, live virtual workshops

"like airhacks.com, without leaving your home"

You don't like live, interactive virtual workshops? Checkout video courses: airhacks.io

Live, Virtual Online Workshops, Winter 2021, [backends]:

CI/CD, Testing, Observability, Resiliency on AWS Cloud, December 9th, 2021

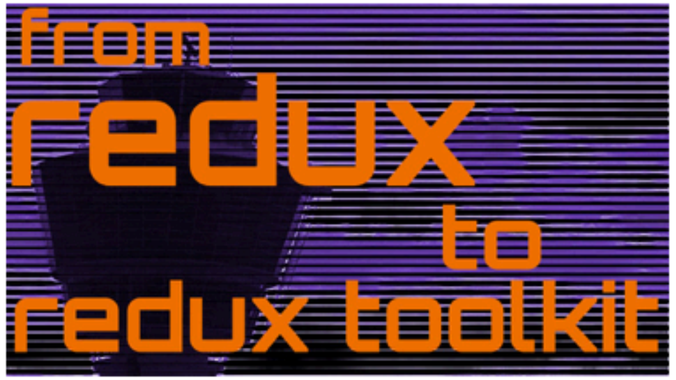
Serverless Java on AWS Cloud, December 16th, 2021

Tickets are also available from: airhacks.eventbrite.com and meetup.com/airhacks

by and with adam-bien.com

>>web workshops

From redux to redux toolkit

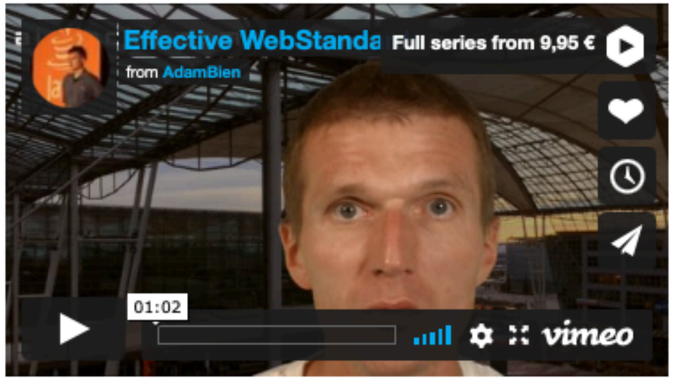


From redux to redux toolkit

Web Components, lit-html and redux

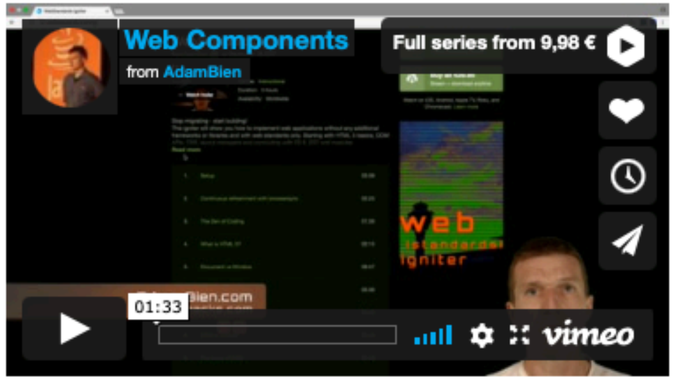
webcomponents-with-redux.training

Effective Web Apps with Web Standards (only)



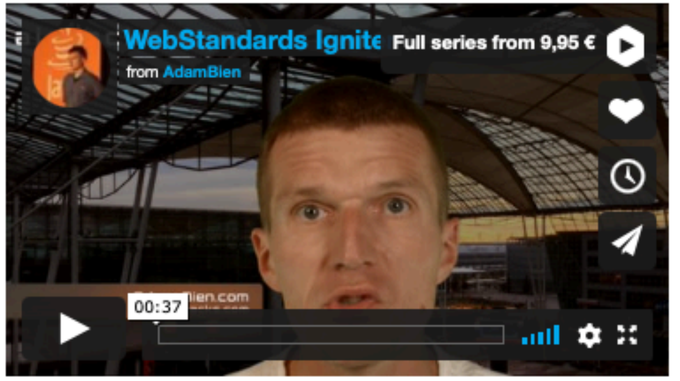
effectiveweb.training

Web Components Igniter



webcomponents.training

WebStandards Igniter



webstandards.training

>>java workshops

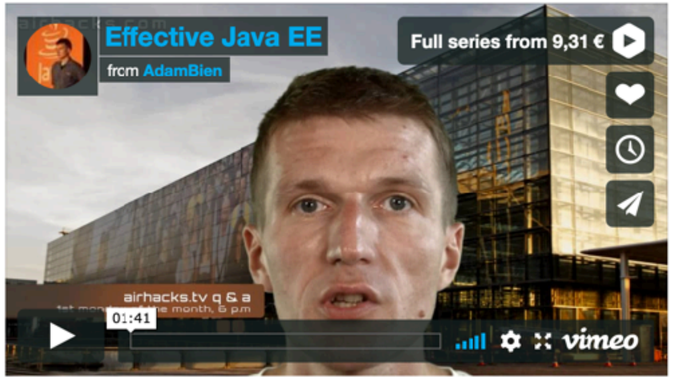
Apps with MicroProfile

microprofile.training

Java EE 7 bootstrap

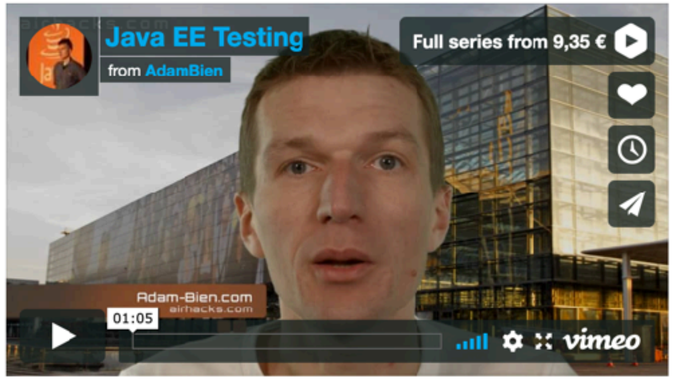
javaeebootstrap.com

Effective Java EE 7



effectivejavaee.com

Java EE 7 Testing



javaeetesting.com

Java EE 7 Microservices

javaeemicro.services

airhacks.live

NEW

coupon code: redux4free

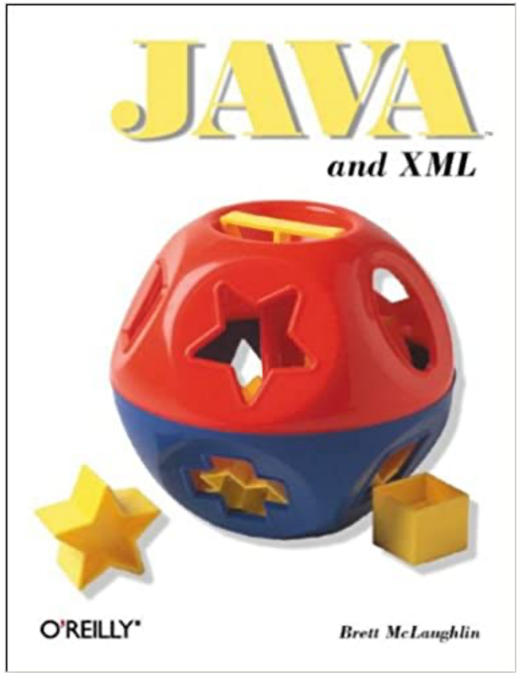
airhacks.live

ask questions, have fun :-)

enabled camera is more fun, but fully optional

brief history of serverside Java

- Sun Java Web Server (before XML, DevOps)
- early J2EE (lots of XML, shared deployments)
- late J2EE (lots of generated XML, shared deployments)
- Java EE / Jakarta EE (CoC, shared deployments, no XML)
- MicroProfile (no XML, optional configuration, cloud features)
- Java, MicroProfile, Jakarta EE + Clouds (no XML, lots of yaml or JSON)
- Late Jakarta EE / MicroProfile (no XML, lots of generated YAML or JSON, or IaC)



shared deployments

Sun E10K

E10K Server - Enterprise 10000 Server - Sun Microsystems

Sun Enterprise E10000 Server. This is a complete server with all system boards, power supplies, fan trays etc. It was installed at AOL and was in good working order until it was deinstalled. The CPU's and memory have been removed, but this is a great machine for parts. If you are suporting E10K servers, this is a perfect spare.Click here for technical information on the Sun Fire E10K server

- 30-Day Warranty on all hardware
- Great for spare parts. Complete list of installed options is available upon request.
- Can ship today
- in stock. Call for availability on quantities over 1

Delta Marketing Group, Inc. since 1986

We would also be happy to take your order or answer any questions over the phone
480-367-6698

For Sales dial 1

or
Michael Hankerson
extension 101

Manufactured By: Sun Microsystems

Print Product Description | Email a Friend | Sign-up for Monthly Specials | Update My Profile

List Price:$1,200,000.00
Qty 1 qty:
Add To Cart
$1,995.00

deployments

deployments

- on premise
- bare metal
- virtualization
- public clouds
- private clouds
- containers

Useful Java 8+ (17) Features

Standard APIs for boring plumbing

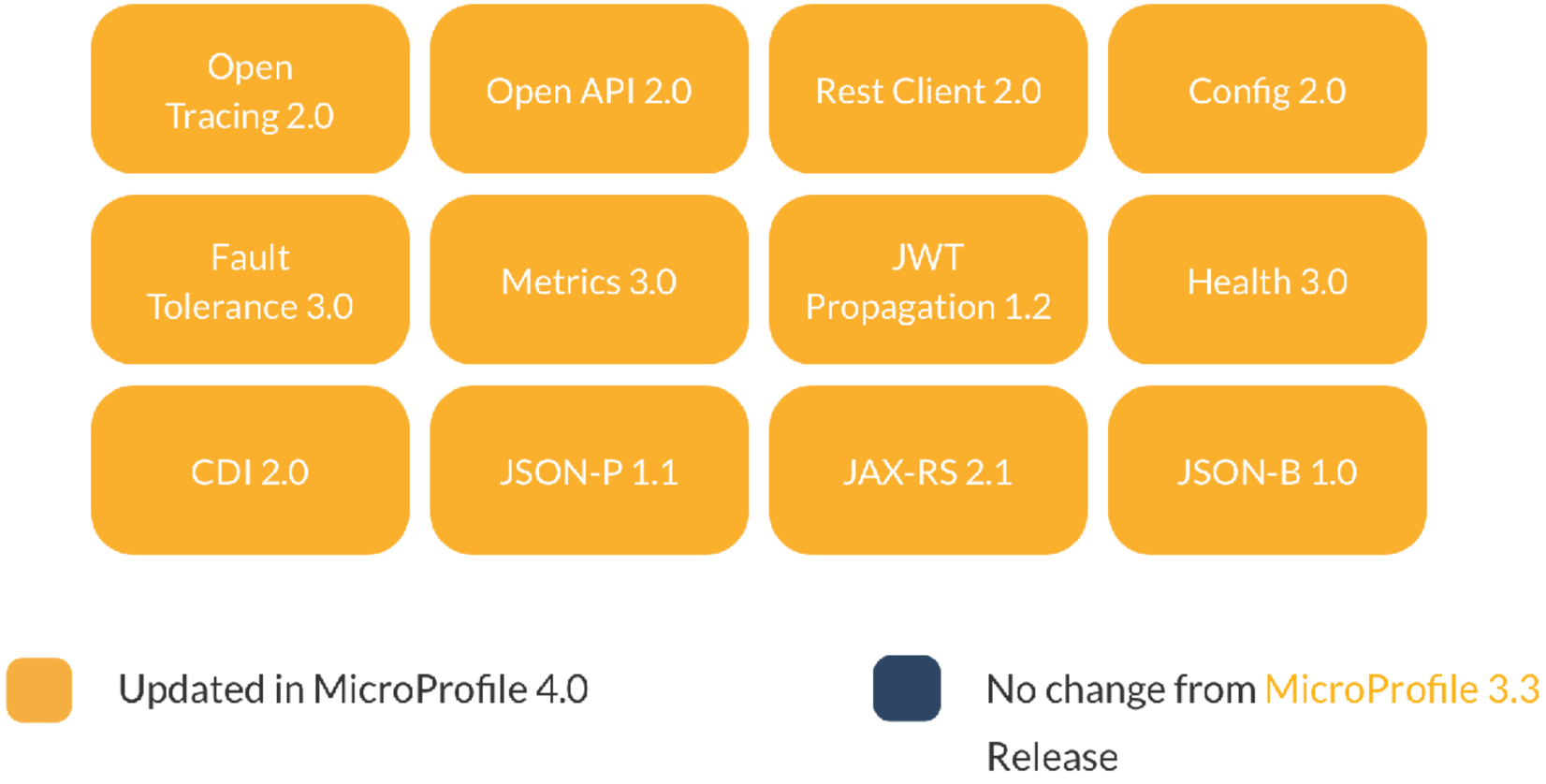
Modern backend applications

Killer use cases for GraalVM

Useful GraalVM features

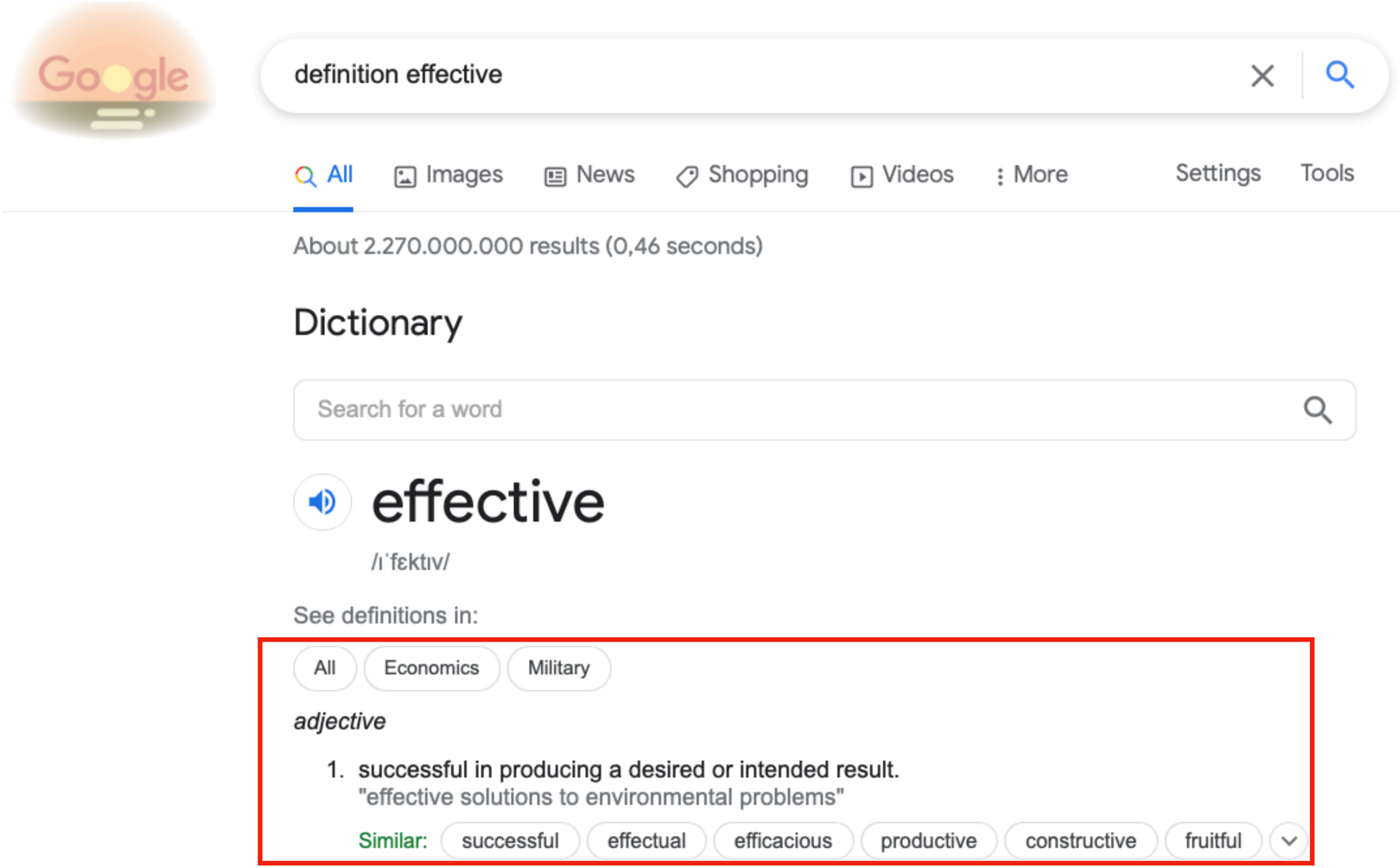
Microprofile && Jakarta EE

MicroProfile 4.0



effective java architectures

what is a “effective”?



The larger the company, the less the “desired outcome” is defined.

the ultimate internal structure

Boundary Control Entity

- ancient pattern / icons available for all tools
- top level packages are named after business responsibilities
- boundary: the perimeter / the facade
- control: procedural logic
- entity: data and domain logic
- works with frontends and backends

architecture

- data and business logic location
- combining or separating
- type safety vs. flexibility
- self-containment vs. reuse
- build or buy
- vendor neutrality vs. secret sauce

microservices

AWS history

During 6 years at Amazon he witnessed the transformation of the company from a bookseller to the almost $1B, Infrastructure as a Service (IaaS) API, cloud computing leader. As Yegge's recalls that one day Jeff Bezos issued a mandate, sometime back around 2002 (give or take a year):

- All teams will henceforth expose their data and functionality through service interfaces.
- Teams must communicate with each other through these interfaces.
- There will be no other form of inter-process communication allowed: no direct linking, no direct reads of another team's data store, no shared-memory model, no back-doors whatsoever. The only communication allowed is via service interface calls over the network.
- It doesn't matter what technology they use.
- All service interfaces, without exception, must be designed from the ground up to be externalizable. That is to say, the team must plan and design to be able to expose the interface to developers in the outside world. No exceptions.

The mandate closed with:

> ***Anyone who doesn't do this will be fired. Thank you; have a nice day!***

Everyone got to work and over the next couple of years, Amazon transformed itself, internally into a service-oriented architecture (SOA), learning a tremendous amount along the way.

Think about what Bezos was asking! Every team within Amazon had to interact using web services. If you were human resources and you needed some numbers from marketing, you had to get them using an API. He was asking every team to decouple, define what resources they had, and make them available through an API. Every team within your company essential becomes a partner of the other.

https://apievangelist.com/2012/01/12/the-secret-to-amazons-success-internal-apis/

“In short, the microservice architectural style is an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API. These services are built around business capabilities and independently deployable by fully automated deployment machinery. There is a bare minimum of centralized management of these services, which may be written in different programming languages and use different data storage technologies.”

http://martinfowler.com/articles/microservices.html

Microservices is a software architecture design pattern, in which complex applications are composed of small, independent processes communicating with each other using language-agnostic APIs. These services are small, highly decoupled and focus on doing a small task.

http://en.wikipedia.org/wiki/Microservices

micro services

“Microservice architecture – a variant of the service-oriented architecture (SOA) structural style – arranges an application as a collection of loosely coupled services. In a microservices architecture, services are fine-grained and the protocols are lightweight.”

https://en.wikipedia.org/wiki/Microservices

a refactored monolith

breaking the data-silo is the challenge

a single application represented by communicating processes

…and now lets talk about clouds

Thank You!