

# The Indisputable Structure

...for Backends and Frontends

**"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

# upcoming

## airhacks.live

**NEW** online, live virtual workshops

*"like [airhacks.com](https://airhacks.com), without leaving your home"*

You don't like live, interactive virtual workshops? Checkout video courses: [airhacks.io](https://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](https://airhacks.eventbrite.com) and [meetup.com/airhacks](https://meetup.com/airhacks)

by and with [adam-bien.com](https://adam-bien.com)

airhacks.live

## >>web workshops

From redux to redux toolkit



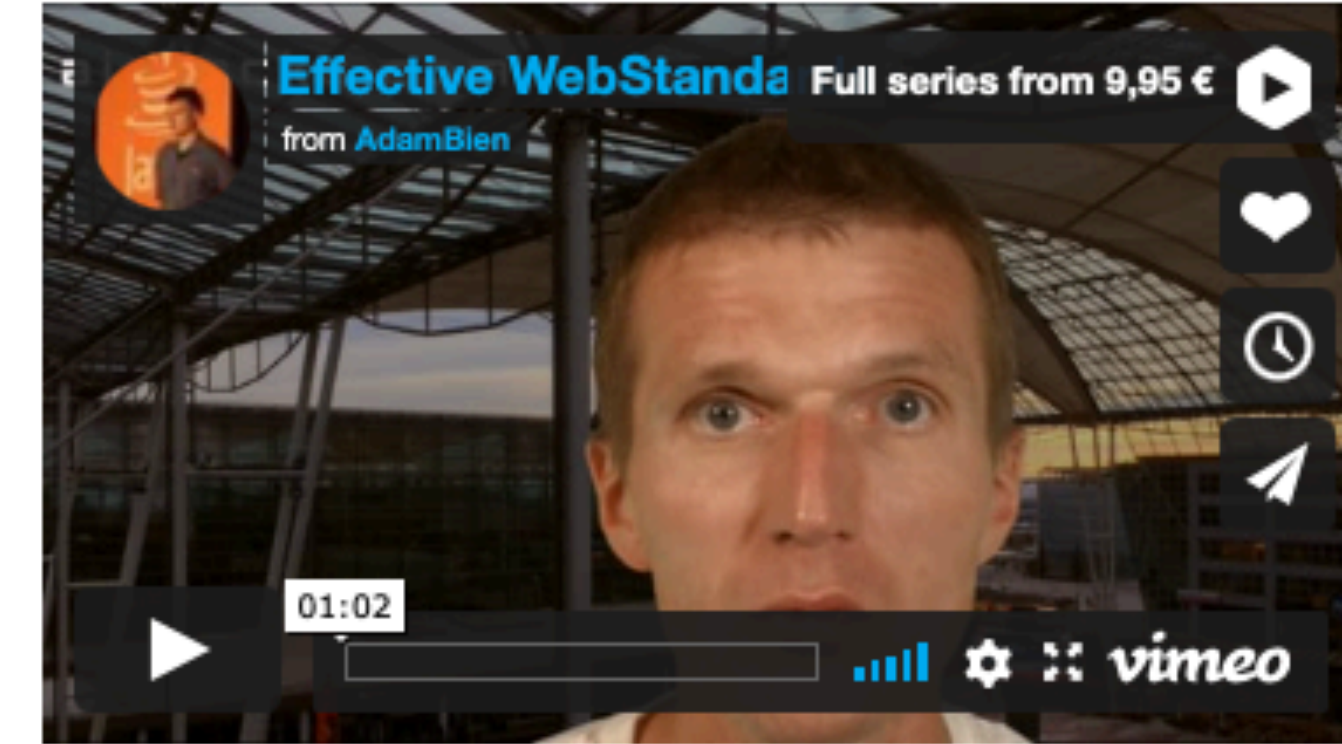
From redux to redux toolkit

Web Components, lit-html and redux



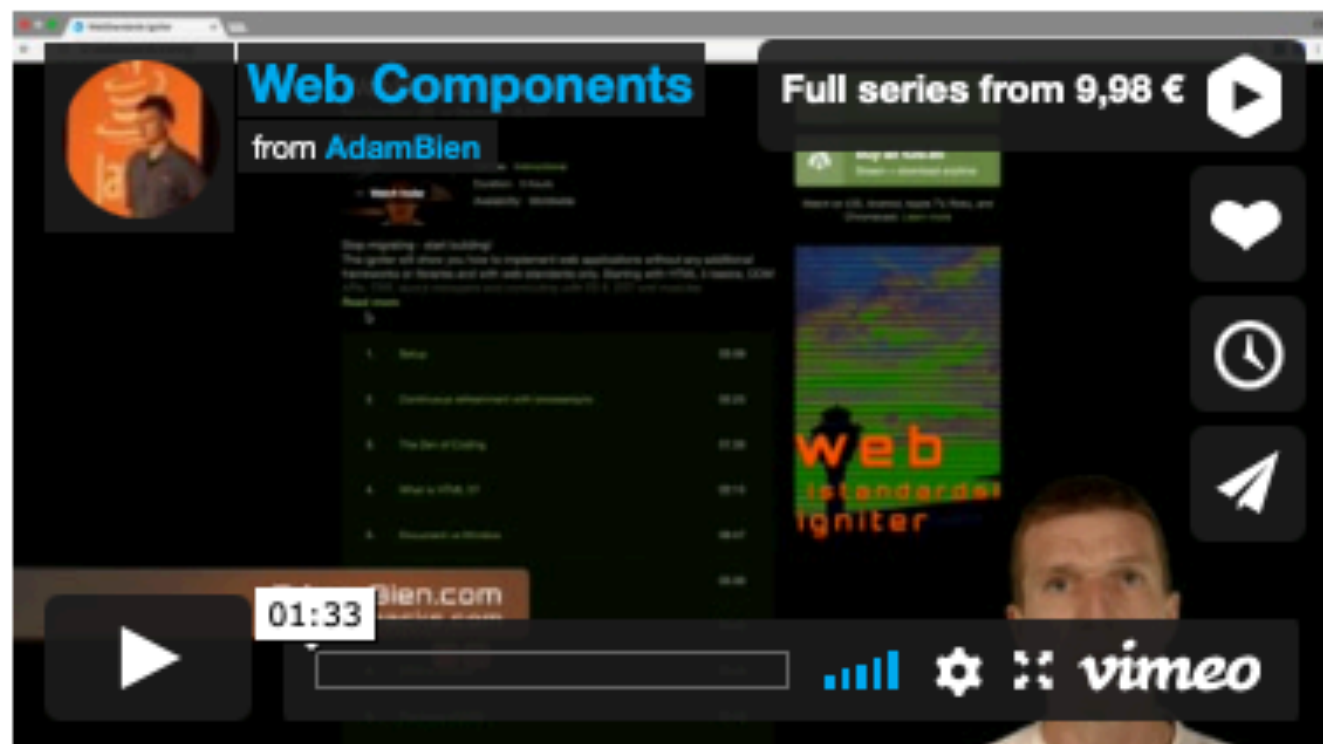
webcomponents-with-redux.training

Effective Web Apps with Web Standards (only)



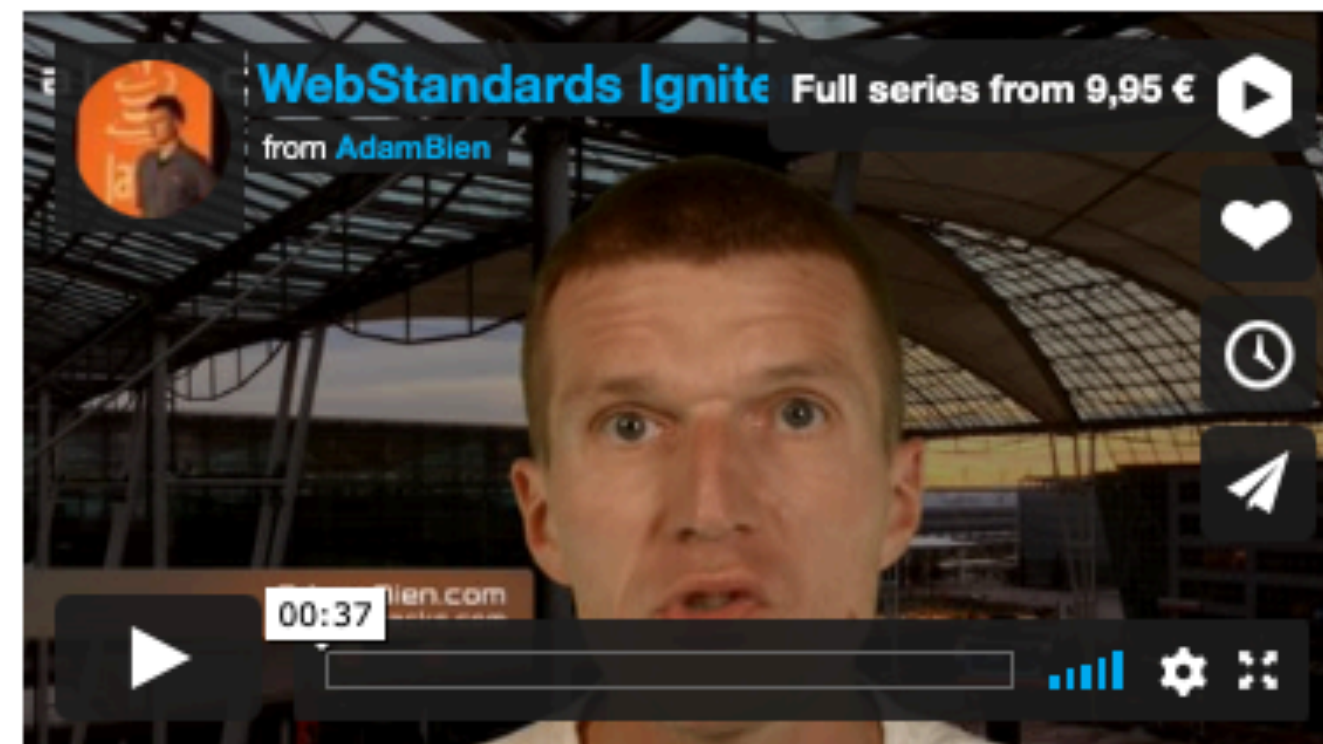
effectiveweb.training

Web Components Igniter



webcomponents.training

WebStandards Igniter



webstandards.training

## >>java workshops

Apps with MicroProfile



[microprofile.training](http://microprofile.training)

Java EE 7 bootstrap



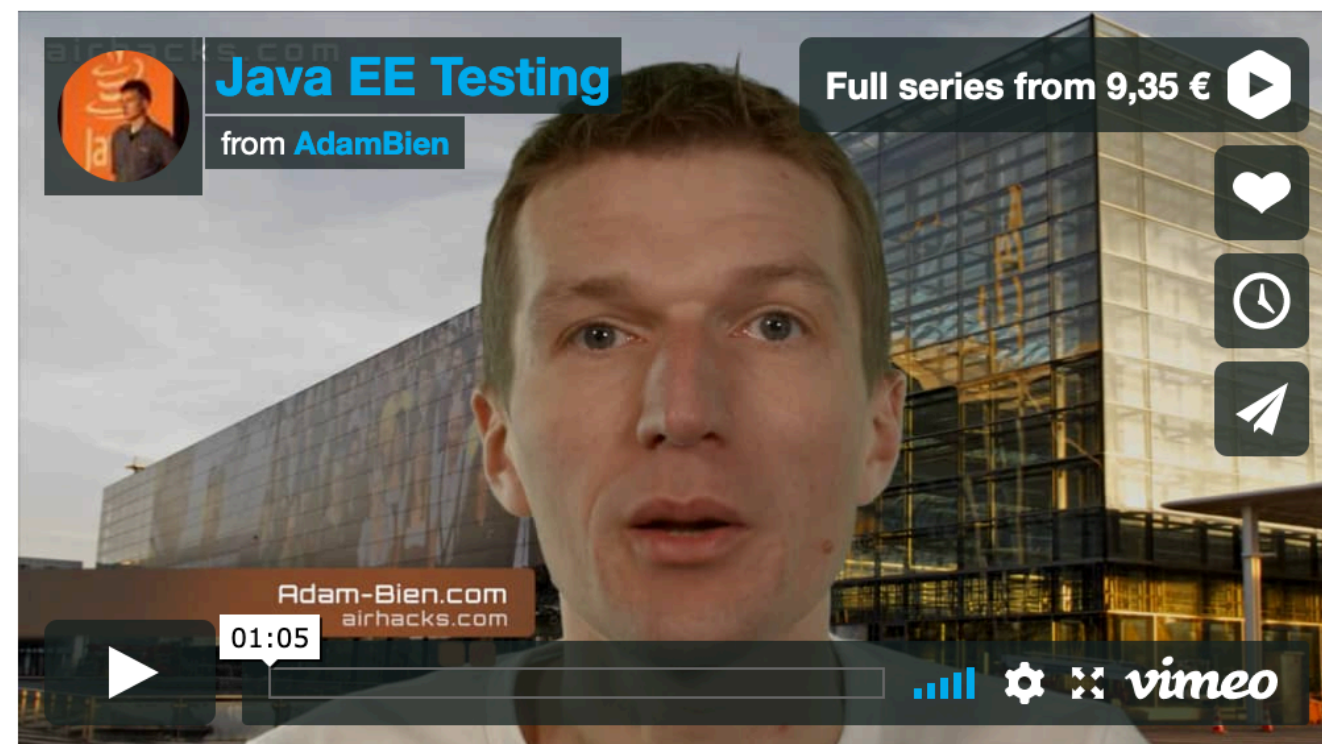
[javaeebootstrap.com](http://javaeebootstrap.com)

Effective Java EE 7



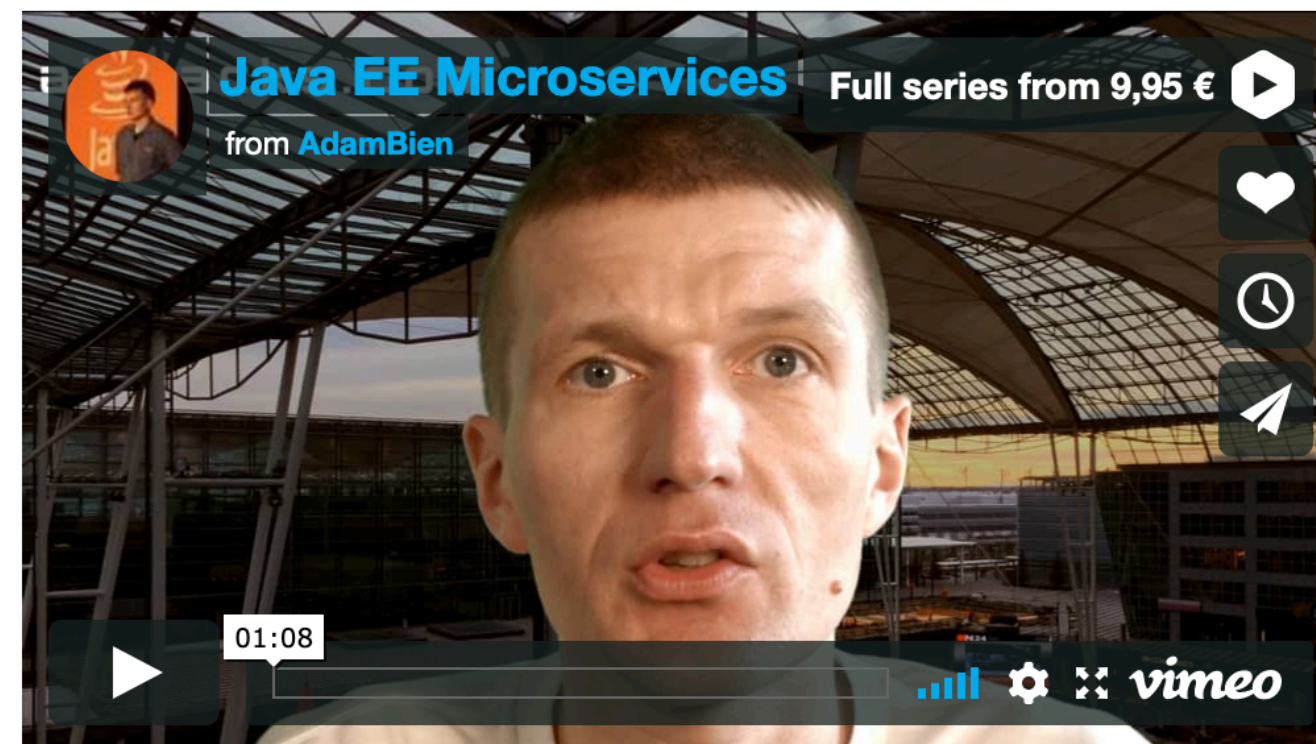
[effectivejavaee.com](http://effectivejavaee.com)

Java EE 7 Testing



[javaeetesting.com](http://javaeetesting.com)

Java EE 7 Microservices



[javaeemicro.services](http://javaeemicro.services)

**don't make me think**



# Information Cascades

“Information cascades occur when external information obtained from previous participants in an event overrides one's own private signal, irrespective of the correctness of the former over the latter.”

[https://en.wikipedia.org/wiki/Information\\_cascade](https://en.wikipedia.org/wiki/Information_cascade)

# Cargo Cult Programming

“Cargo cult programming is a style of computer programming characterized by the ritual inclusion of code or program structures that serve no real purpose. Cargo cult programming is symptomatic of a programmer not understanding either a bug they were attempting to solve or the apparent solution (compare shotgun debugging, deep magic)”

[https://en.wikipedia.org/wiki/Cargo\\_cult\\_programming](https://en.wikipedia.org/wiki/Cargo_cult_programming)

# Parkinson's Law of Triviality

“Law of triviality is C. Northcote Parkinson's 1957 argument that people within an organization commonly or typically give disproportionate weight to trivial issues.”

“...whose job was to approve the plans for a nuclear power plant spending the majority of its time on discussions about relatively minor but easy-to-grasp issues, such as what materials to use for the staff bicycle shed, while neglecting the proposed design of the plant itself, which is far more important and a far more difficult and complex task. “

[https://en.wikipedia.org/wiki/Law\\_of\\_triviality](https://en.wikipedia.org/wiki/Law_of_triviality)

# Decision Fatigue

“In decision making and psychology, decision fatigue refers to the deteriorating quality of decisions made by an individual after a long session of decision making. It is now understood as one of the causes of irrational trade-offs in decision making.”

[https://en.wikipedia.org/wiki/Decision\\_fatigue](https://en.wikipedia.org/wiki/Decision_fatigue)

# GRASP

- High Cohesion, Minimal Coupling
- Protected Variations
- Information Expert
- Polymorphism
- Pure Fabrication

[https://en.wikipedia.org/wiki/GRASP\\_\(object-oriented\\_design\)](https://en.wikipedia.org/wiki/GRASP_(object-oriented_design))

# Package Principles

“In computer programming, package principles are a way of organizing classes in larger systems to make them more organized and manageable. They aid in understanding which classes should go into which packages (package cohesion) and how these packages should relate with one another (package coupling).”

[https://en.wikipedia.org/wiki/Package\\_principles](https://en.wikipedia.org/wiki/Package_principles)


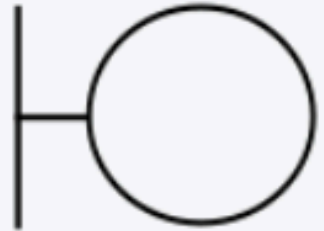

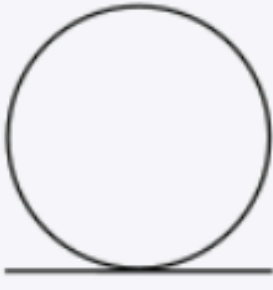
# Entity-Control-Boundary

The ECB pattern organises the responsibilities of classes according to their role in the use-case realization:

- an entity represents long-lived information relevant for the stakeholders (i.e. mostly derived from domain objects, usually persistent);
- a boundary encapsulates interaction with external actors (users or external systems);
- a control ensures the processing required for the execution of a use-case and its business logic, and coordinates, sequences controls other objects involved in the use-case.

<https://en.wikipedia.org/wiki/Entity-control-boundary>

# BCE is beautiful :-)

Representation		Relation with			
Role	Symbol	Actor	Boundary	Control	Entity
Actor		Yes	Yes	No	No
Boundary		Yes	Part/whole	Yes	No
Control		No	Yes	Yes	Yes
Entity		No	No	Yes	Yes



# Conway's Law

Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure.

[https://en.wikipedia.org/wiki/Conway%27s\\_law](https://en.wikipedia.org/wiki/Conway%27s_law)

# Examples

- monoliths
- microservices
- stream processors
- FaaS / Azure Functions / AWS Lambda
- SPA / PWAs
- CLI

# Code-Walkthroughs

- <https://github.com/AdamBien/lightfish>
- <https://github.com/AdamBien/blogpad>
- <https://github.com/AdamBien/mockend>
- <https://github.com/AdamBien/aws-lambda-cdk-plain>
- <https://github.com/AdamBien/bce.design>
- <https://github.com/AdamBien/webcomponents-with-redux.training>
- <https://github.com/AdamBien/jwttenizr>
- <https://github.com/AdamBien/wad>

**Thank YOU!**