

Event Sourcing

basic introduction

Disclaimer

- I don't consider myself a grand architect
- I just don't like complexity
- I like system that I can reason about
- ... and I am prepared to pay the price for it

My History

- 5 consulting projects for Deutsche Bank during university (learned how banks tick)
- 4.5 year DaWanda, joined with 12 people, left with 200, 2.5 last years DevOps
- 1.5 years building complex Frontend UIs for corporation dashboards

React.js

- Promise: UI is a result of a function call with state...
- $UI = appFn(state)$
- mindblown...

UI State

- but what about state?
- Flux: unidirectional data flow
- components subscribe to changes in stores and just do a conceptually full - re-render (thx VirtualDom)
- what about history?

UI State

- Redux came and stole “time-travel” from Elm
- basic idea: all events added up result in the present/current state
- stores derive their internal state from the event stream

But that is just frontend...

- right?
- I mean, frontend usually deals with small amounts of data, so that is not an issue there
- would something like this work for the backend?

It would.

- it is called event sourcing and there are quite some companies doing it
- historically it was expensive (storage / computation), but nowadays it is very affordable
- it is not a solution for everything, so must be used only when you derive competitive advantage from it

Companies

- because it feels so weird and “over-the-top”-architecture, I like to give you some comfort
- real companies are using it and __surprise__ they like it

Paydirekt - mit Microservices gegen Paypal und Co. - Online-Bezahlplattform

- **Event-Sourcing sorgt für Compliance und leichtes Reporting**
- <http://www.channelpartner.de/a/paydirekt-mit-microservices-gegen-paypal-und-co,3223838,2>
- Anstatt zum Beispiel Stammdaten bei Änderungen zu überschreiben, sprich die alten Daten zu löschen, werden sie aus den gemachten Änderungen zusammengesetzt. Dieses Prinzip wird auch bei den Bewegungsdaten angewendet. Da sämtliche Veränderungen mit Datum und Urheber versehen sind, lässt sich der Zustand des Systems zu jedem gegebenen Zeitpunkt quasi auf Knopfdruck rekonstruieren und jede Veränderung erkennen. "Damit erfüllen wir automatisch sämtliche Dokumentations- und Reporting-Verpflichtungen, die den Kreditinstituten vom Gesetzgeber und von der Bankenaufsicht auferlegt werden", unterstreicht Omann einen der Vorteile.

irekt



El



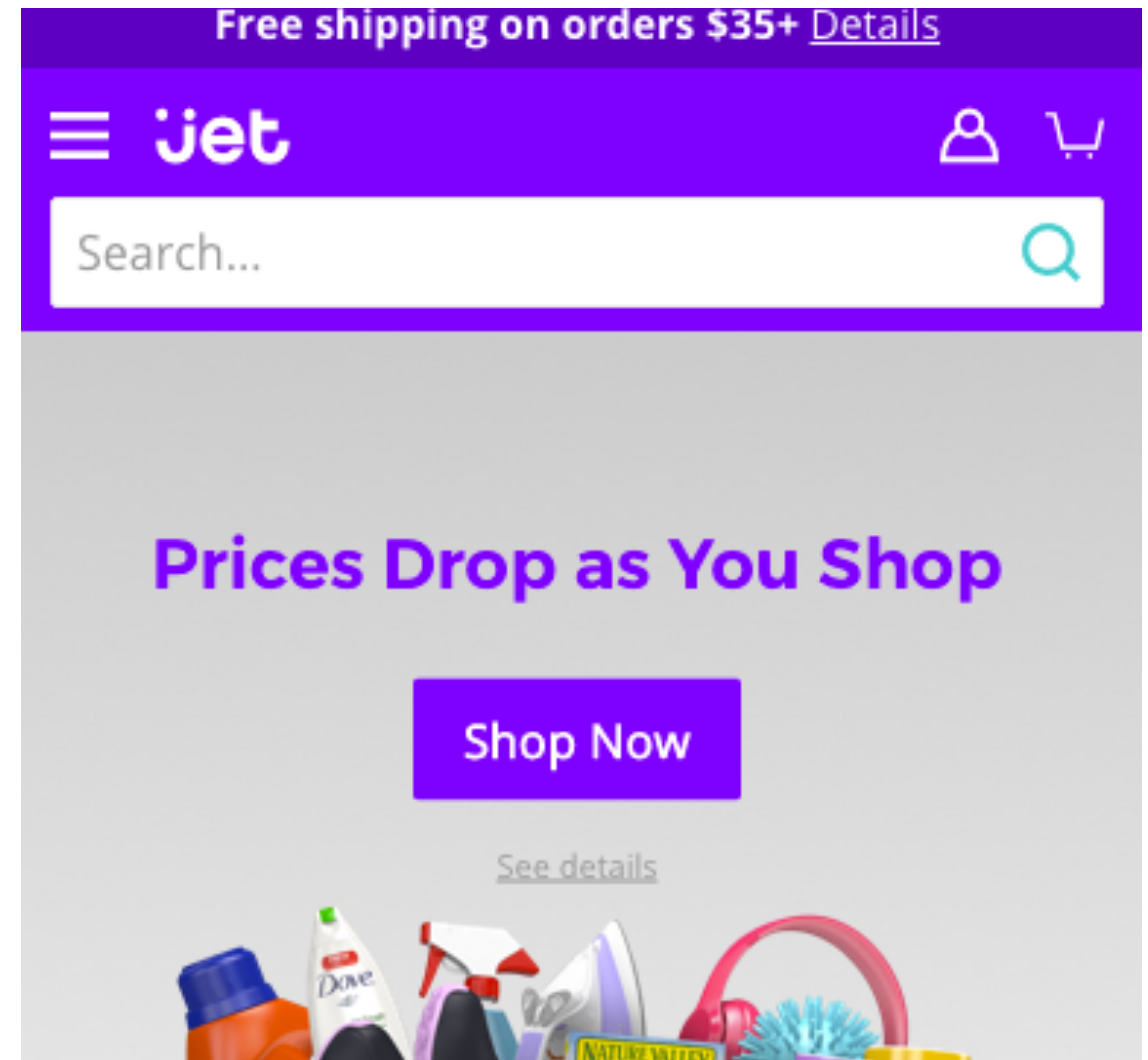
Billy - Free Accounting Software for Small Business




- <https://www.billyapp.com/jobs/>
- Back-end
- We use a microservice-oriented architecture.
- The back-end uses a single REST API (the same API that our main web app at app.billy.com uses.)
- Most of the services run Node.js and we also work with Elixir, Python and PHP.
- All services run in Docker containers within our Mesos/Marathon cluster, which makes our DevOps very easy to work with.
- The services communicate internally using a private API, using asynchronous events.
- We've started working with event sourcing and CQRS, for which we use Event Store.



Jet.com - e-commerce company (now belongs to Walmart)

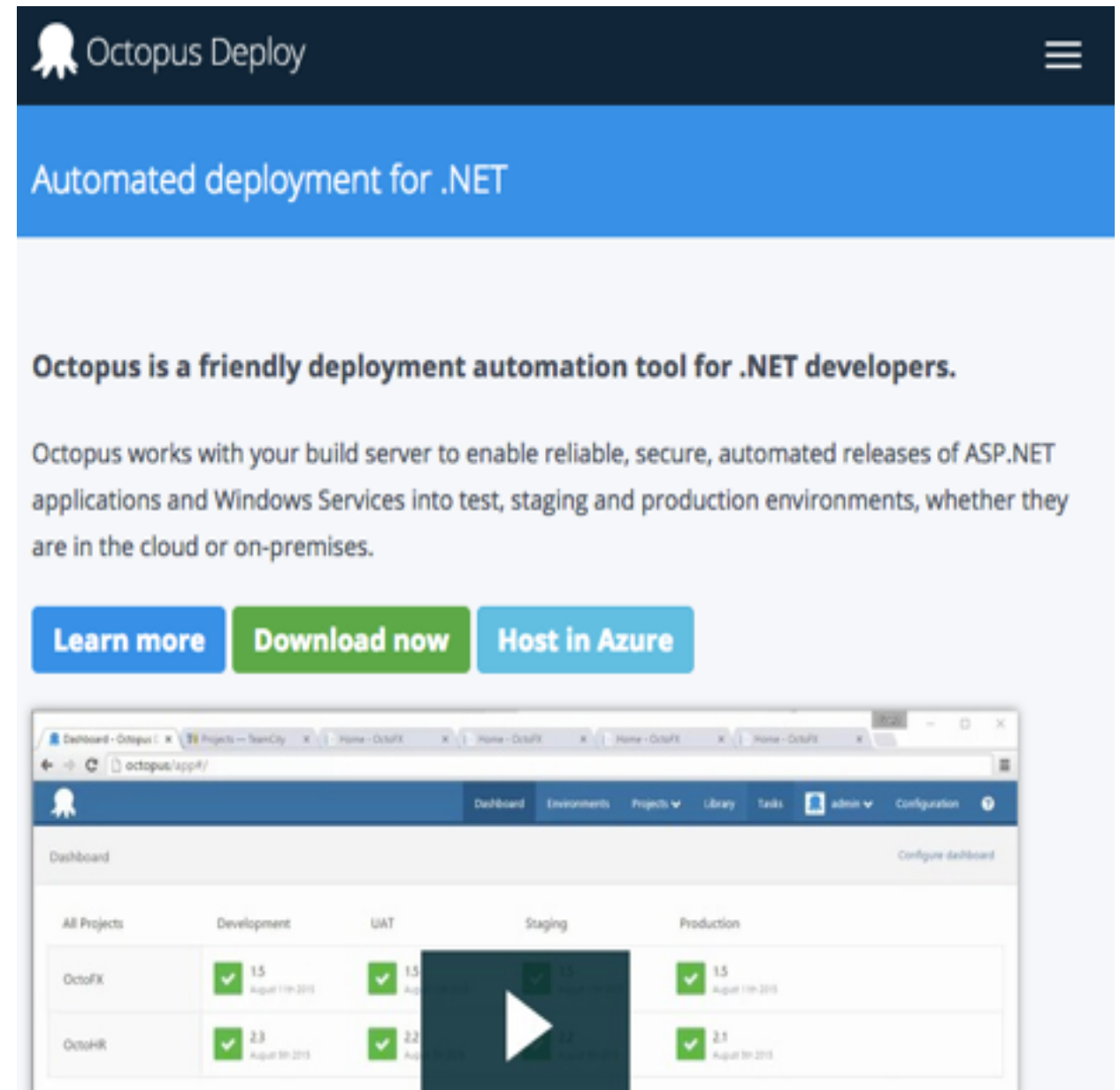
- <https://tech.jet.com/blog/2016/09-07-event-sourcing-awesome/>
- <https://tech.jet.com/blog/2016/01-26-microservices-messaging/>
- <https://vimeo.com/109343720>
- Event sourcing allows us to have a detailed history of the actions that occurred on our systems. It is possible to build the current state of a stream by using fold and its variant, apply. From an auditing perspective it is very powerful in that all transactions contain detailed history. Another benefit to event sourcing is that we can replay the events over a period of time to recreate the state of the system up to that point in time. This means that it is possible to replay a bug in the system, which is very helpful in tracking down errors in distributed systems.



-  Everyday low prices to start
-  Prices drop lower as you shop
-  Fast delivery to your door

Octopus Deploy - Automated deployment for .NET

- <https://octopus.com/blog/the-evolution-of-auto-deployments-and-event-sourcing>
- Our auto-deployment engine is then simplified to a pure event-sourcing architecture, which makes it more isolated and easier to test (and works under HA). It runs every 15 seconds and scans the Event table to see what events have occurred since it last passed. It then analyses and balances these events in terms of positive events (machine was added, came online, was enabled) and negative events (machine was deleted, went offline, was disabled) and can then decide on what releases to auto-deploy.



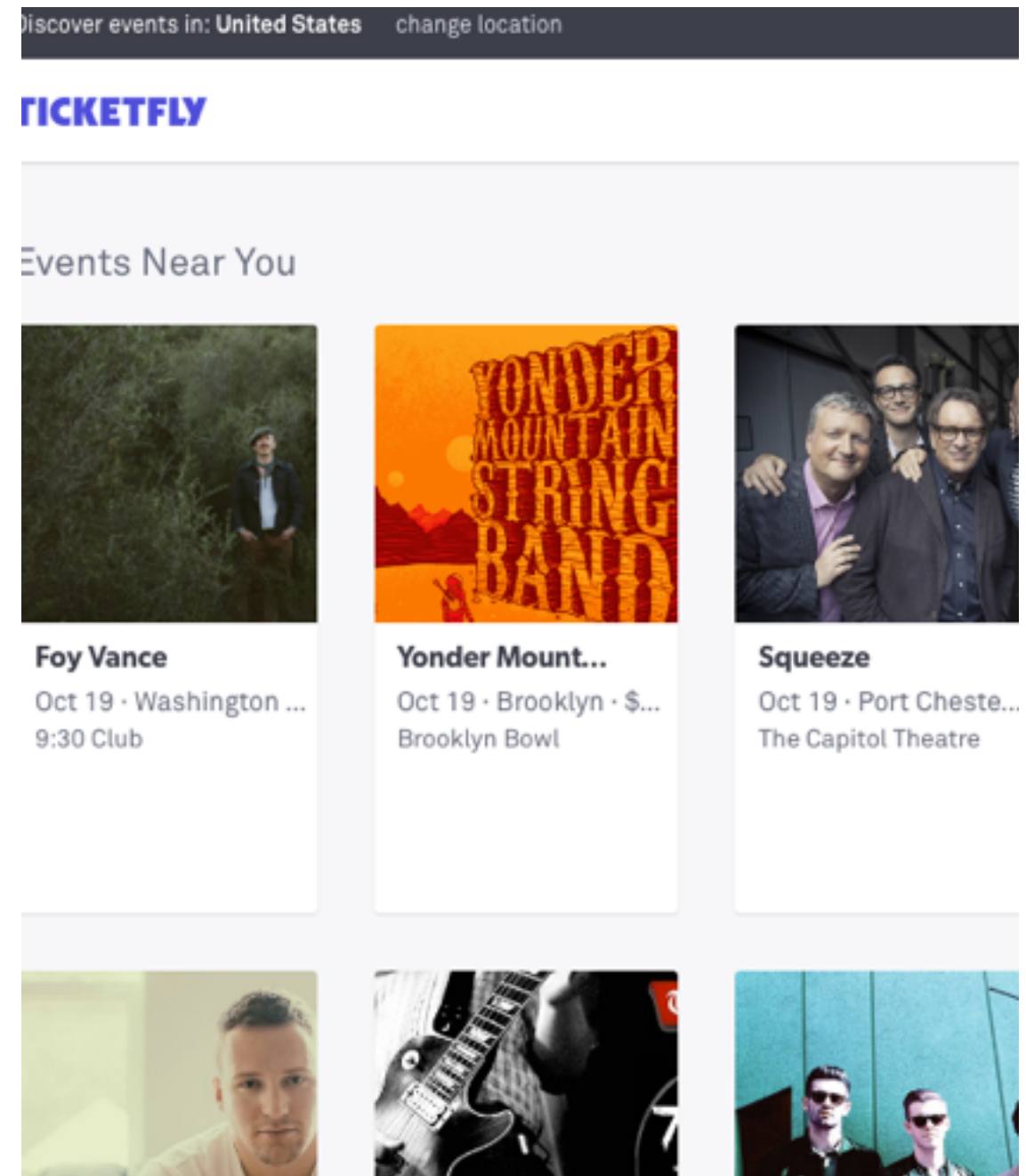
Paycasso - Conquers Speed Gap In Three-Factor Authentication With Reactive Stack

- - <https://www.lightbend.com/resources/case-studies-and-stories/paycasso-conquers-speed-gap-in-three-factor-authentications-with-reactive-stack>
- Apache Cassandra - Paycasso's journals and snapshots are stored in Apache Cassandra, a highly-scalable, decentralized NoSQL database. The journals and snapshots are an essential component in CQRS/ES systems, and they provide the backing for elastic cluster scaling.



Ticketfly - integrated ticketing and digital marketing platform for event promoters

- <https://www.lightbend.com/resources/case-studies-and-stories/lightbend-helps-ticket-sales-soar-at-ticketfly>
- The overall solution lay in a CQRS Event Sourcing architecture that would thereby allow Andrew to bring the business logic up into the application layer as opposed to pushing it down into the database. He modeled everything much of the system as finite state machines (FSM); seats, indexes, check out processes, etc.



Yammer

- <https://berlinbuzzwords.de/session/event-sourcing-yammer>
- Event Sourcing in Yammer
- Only slightly over a year ago, Yammer's entire system was either based on synchronous calls or Ruby workers and RabbitMQ. For a while, we have been moving performance-critical components out of the Ruby on Rails monolith toward Dropwizard-based services. This has served us well, but with increased reliability requirements, the pressure to simplify and decouple our system's architecture also increased.
- Over the course of the last year, we first created a prototype implementation of Event Sourcing and, after validating the idea, have been moving it to a managed Azure Event Hubs based solution.
- We are going to cover not only how to migrate to a new technology, but also look at how to change organizational thinking from a synchronous world to one of event streams. We will tell the war stories of our migration from a self-hosted Kafka cluster to a solution based in the cloud.



Halten Sie Ihr Team auf dem Laufenden

Organisieren Sie die Teamkommunikation zentral mit Yammer-Gruppen. Erstellen Sie ein einheitliches Ziel für Nachrichten, Dateien und Updates, wo jeder

OptioPay - Job offer

- <http://www.golangprojects.com/golang-go-job-baa-Senior-Backend-Developer-Golang-Berlin-OptioPay-GmbH.html>
- Your Role
- To create and maintain services written in Golang which will be connected through a central message and event bus (Kafka)
- Extend **our RESTful API** and data model which is based on the **event sourcing paradigm**
- To ensure your code is accurate, stable and comprehensive by conducting reviews as well as writing unit and acceptance tests
- To help us reach more customers by integrating the OptioPay platform with banks, PSPs and other financial systems
- To engage with continuous improvement of our services. This will include reviews of business processes, infrastructure and hardware as well as enabling frontend developers to collect more data



Capital Match

- <https://www.capital-match.com/>
- <https://abailly.github.io/posts/event-source.html>
- Arnaud Bailly - Life Beyond Relational Database in Haskell - The case for Event Sourcing
- <http://videos.ncrafts.io/video/168197756>
- ARNAUD BAILLY-LIFE BEYOND RELATIONAL DATABASE, EVENT SOURCING IN HASKELL



Refer a SME or an Investor and Get \$\$\$ >

CM Singapore Risks | Register | Log in

Loans funded to date: S\$19,987,285

Short-term business financing made simple!

Get the cash you need to grow your business or SME

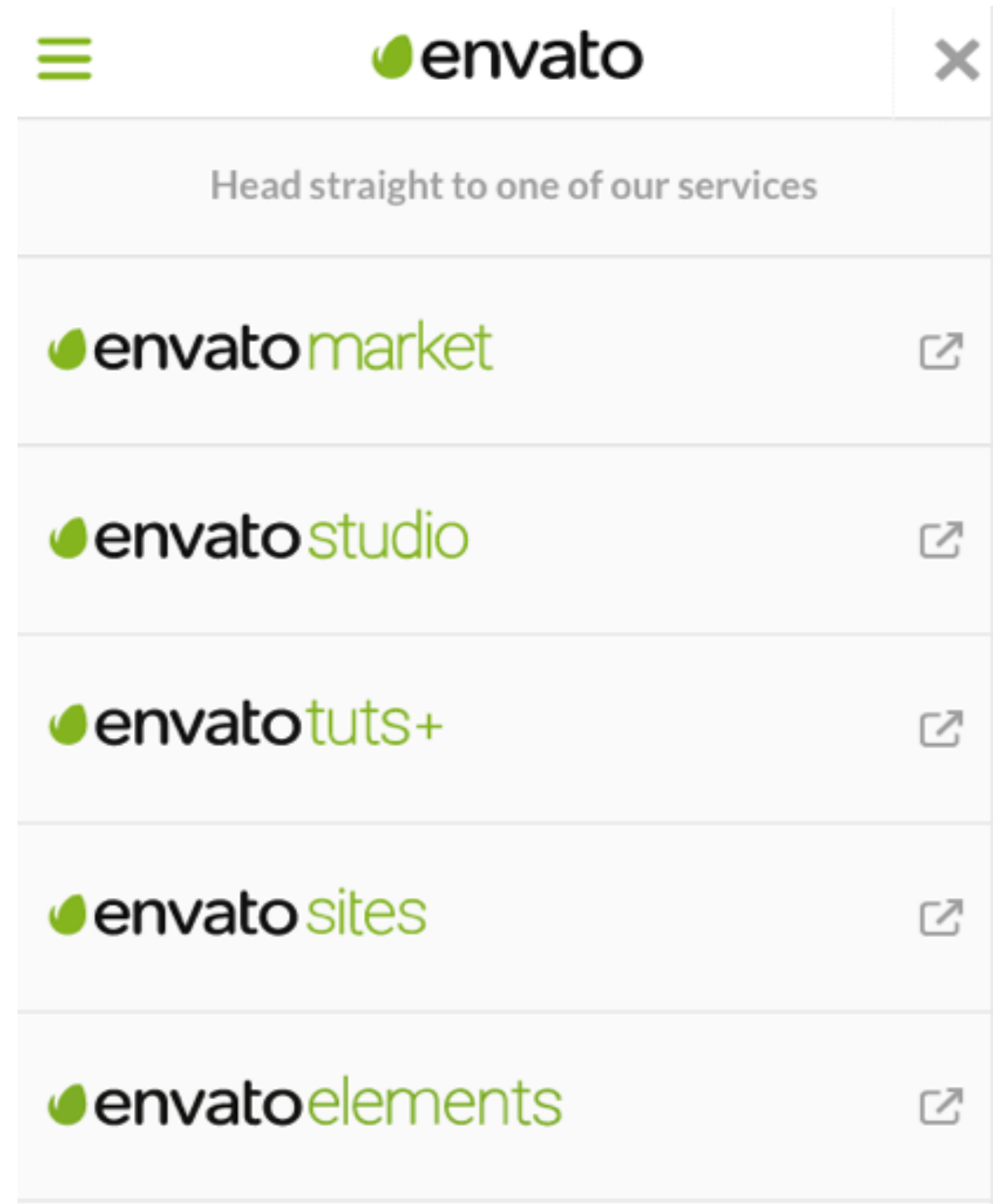
Get a Business Loan

Invest Your Money

Get Business Financing

Envato - Top digital assets and services

- UPDATE Your View on DELETE: The Benefits of Event Sourcing
- https://resources.sei.cmu.edu/asset_files/Presentation/2016_017_001_454714.pdf
- https://www.youtube.com/watch?v=_TeMYF_JjNg
- <https://www.infoq.com/presentations/event-sourcing>



Envato slides

- because many people already invested time explaining those concepts, I decided to cheat and continue with their slides
- Yeah, I am a lazy bastard :P

Future

- after spending some time reading case studies on Lightbend I am a bit unsure about next steps
- Lightbend
- <https://www.lightbend.com/resources/case-studies-and-stories>

FAQ:

- Q: cqrs vs cqs?
- CQS puts commands and queries in different methods within a type.
- CQRS puts commands and queries on different objects.

FAQ

- Data Inconsistency
- <https://jazzy.id.au/2016/10/08/cqrs-increases-consistency.html>
- So by employing CQRS, we are able to get back some of the consistency guarantees that we lost when we moved to microservices. We don't have a globally consistent system, but we can guarantee an eventually consistent system.
- CQRS is a necessary evil
- CQRS is complex, far more complex than handling both the command and query responsibilities of data in single operations on a strongly consistent database. However, in a microservices world, we don't have the luxury of relying on a single strongly consistent database. In that world, inconsistency is a given. If someone says using CQRS in microservices means you lose consistency - they have failed to acknowledge that they lost consistency the moment they started using microservices, it was not CQRS that lost them that consistency. Rather, CQRS is a very powerful tool that allows us to address the inherent inconsistency of microservices to give us eventual consistency instead of terminal inconsistency.

event sourcing - keep the cost of change lower for longer

- <https://www.infoq.com/presentations/event-sourcing>
- <http://www.agileaustralia.com.au/2016/presentations/Agile-Australia-2016-James%20Ross-Sebastian-von-Conrad.pdf>

Event Sourcing, or Why ActiveRecord Must Die

- <https://rubyconf.eventer.com/rubyconf-australia-2016-1489/>
- https://resources.sei.cmu.edu/asset_files/Presentation/2016_017_001_454714.pdf
- https://www.youtube.com/watch?v=_TeMYF_JjNg - UPDATE Your View on DELETE: The Benefits of Event Sourcing

Event Sourcing in practice (from Johannes Seitz):

- <https://ookami86.github.io/event-sourcing-in-practice/#considering-event-sourcing/03-mining-event-streams.md>
- <https://github.com/Ookami86/event-sourcing-in-practice>

Life Beyond the Illusion of Present by Jonas Boner

- #MUSTWATCH!
- <https://www.youtube.com/watch?v=Nhz5jMXS8gE>
- <http://www.slideshare.net/jboner/life-beyond-the-illusion-of-present>

Opening Keynote: Greg Young

- Stop Over-Engineering

- <https://www.youtube.com/watch?v=GRr4xeMn1uU>

