

# Performance – From Backend To Frontend

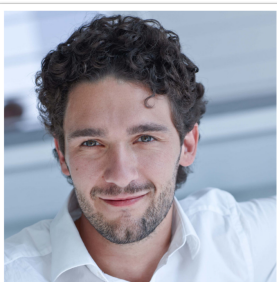
## Oxid Commons 2018

Kore Nordmann (@koredn)  
14th June 2018

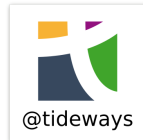


# Hi, I'm Kore (@koredn)

---



Kore Nordmann  
@koredn




# Full Performance In The Mobile Age




# Why Is It Complex?

## Awesome Shop

 2 articles  
42.32 €



### Smartphone

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.



1337,-- €  
5 items in stock

#### Comments



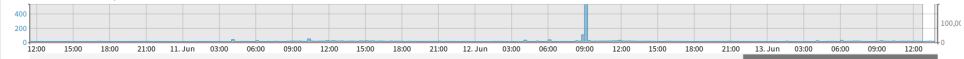
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor.

## What happens on your website?

- ▶ Scenarios in an Online Shop to cover all cases:
  - ▶ Random browser
  - ▶ User registration
  - ▶ Logged in browser
  - ▶ Checkout process
  - ▶ ...
- ▶ Mind additional backend requests by JavaScript frontend enhancements
  - ▶ Page partials load or reload data from backend (services)

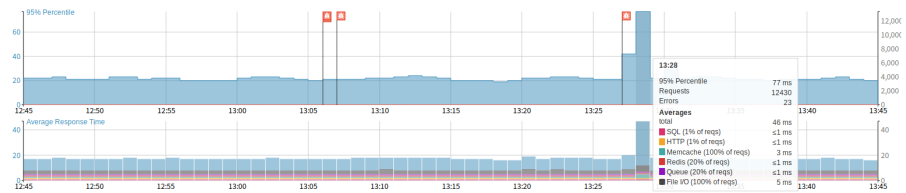
Service: web Environment: production

Select last: 60 minutes 3 hours 12 hours 24 hours



### Response Times for web

22.0 ms 95 Percentile 750k Requests 4 Events 0.05% Error Rate



### Transactions

Name	Requests	Typical Response	Problem Response	Memory	Errors	Impact
XhprofPerformanceBundle.ApiController:recordMultipartAction	449k	14.0 ms	22.5 ms	5.2 MB	0	57%
XhprofProfilerBundle.ApiController:createAction	207k	11.0 ms	19.3 ms	7 MB	0	23%
XhprofProfilerBundle.ApiController:createMultipartAction	45 210	14.0 ms	24.5 ms	34.4 MB	68	6.1%
XhprofPerformanceBundle.ApiController:recordAction	21 802	15.0 ms	32.3 ms	64.9 MB	7	3.3%
XhprofPerformanceBundle.ApiController:graphAction	882	302 ms	628 ms	38.7 MB	0	2.5%
XhprofProfilerBundle.OperationController:criticalAction	861	301 ms	660 ms	18.7 MB	0	2.4%
XhprofAlertBundle.EventApiController:recordAction	652	18.0 ms	28.9 ms	3.7 MB	0	2.1%
XhprofUserBundle.ReplicationController:checkAction	652	20.0 ms	37.9 ms	13 MB	0	1.3%
XhprofPerformanceBundle.ApiController:recordProxyAction	1 655	26.0 ms	237 ms	50 MB	0	1.0%
XhprofPerformanceBundle.ApiController:historyRangeAction	993	39.0 ms	62.0 ms	4.8 MB	0	0.35%
XhprofProfilerBundle.SnapshotController:dataAction	18	1.70 s	1.67 s	26.3 MB	0	0.25%
SQL: XhprofProfilerBundle.EventApiController:replicationAction	348	15.0 ms	257 ms	17.8 MB	0	0.21%
XhprofProfilerBundle.ProfileController:listDataAction	151	66.0 ms	244 ms	4.3 MB	0	0.14%

Complex User Interactions

# Migrate To Frontend Components

---

- ▶ Requirements for user interactions get more complex
- ▶ Can be powered by Angular, React, Vue, ...
- ▶ Components usually still load their data themselves
  - ▶ You could use a state library but this normally requires backend refactoring
- ▶ All data is fetched on each page impression
  - ▶ You could use local storage but this usually requires data structure refactoring

## Benefits

- ▶ Complex user interactions are possible to handle

## Drawbacks

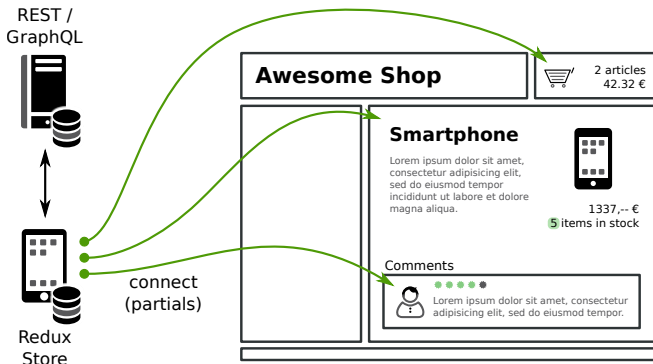
- ▶ Server round trip on click still feels slow to user
- ▶ Storing data local storage requires complex synchronization logic





Now: Single Page Applications (SPA)

# Components in SPAs



# Migrate To Single Page Applications

---

- ▶ More JavaScript means new problems:
  - ▶ Logging of JavaScript errors is now a **must** (Raven.js, Sentry, ...)
  - ▶ Chunking your compiled JavaScript quickly becomes necessary
- ▶ Refactor data structures for in-browser view models
- ▶ Backends “only” deliver static assets and APIs – use HTTP2!

# Backend APIs

---

- ▶ Backend For Frontend (BFF)
  - ▶ Implement (REST) services optimized for your frontend requirements
  - ▶ Best suited for fairly stable frontends with high performance requirements
- ▶ GraphQL / ...
  - ▶ Frontend may query any data, including complex joins
  - ▶ Best suited for quickly changing frontend requirements
  - ▶ “Any query” means that the backend might not be optimized for a certain request type

## Benefits

- ▶ Very high **user perceived** performance
  - ▶ Even a “quick” animation (115ms) is a lot time for a server to fetch and prepare data
- ▶ Easier separation between backend and frontend development

## Drawbacks

- ▶ Requires sensible JavaScript build processes, monitoring, ...
- ▶ Ensure JavaScript performance is well enough on slow devices (old phones)

# Full Performance In The Mobile Age



# Conclusion

---

- ▶ Always monitor server performance – maybe load test them
- ▶ SPAs will increase user perceived performance
- ▶ SPAs also enable modern user interactions



THANK YOU

Rent a quality expert  
[qafoo.com](http://qafoo.com)