Detaillierte Zeitmessungen für Webtests

diva-e Agenda

1. Kurze Vorstellung
2. Motivation
3. Ziel
4. JavaScript APIs
5. Einbindung in Selenium
6. Prometheus
7. Grafana
8. Fazit

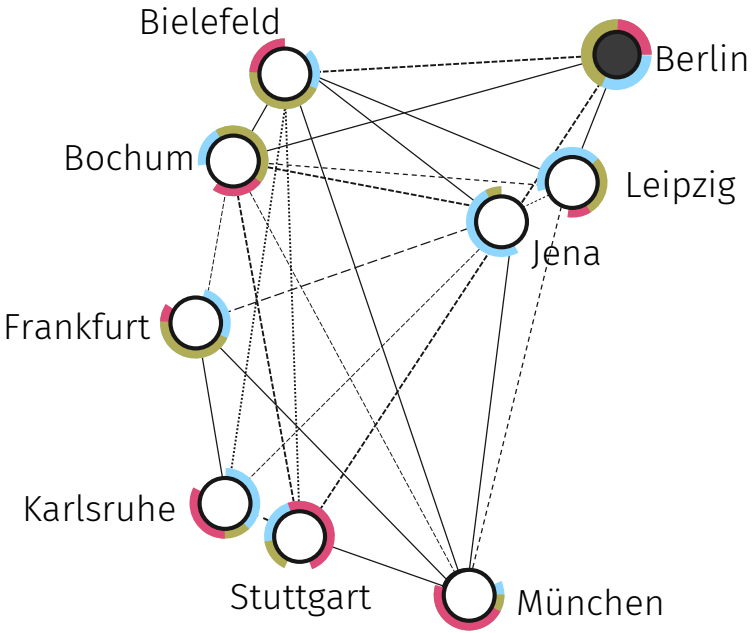
diva-e Referent

Christian Kumpe
Expert Developer

- Informatikstudium am KIT (Universität Karlsruhe)
- Freelancer im Bereich Web und Java
- Seit Mai 2011 bei diva-e in Karlsruhe
- Seit 2002 in der Java-Welt unterwegs

diva-e Unternehmensvorstellung

Digital Value Enterprise



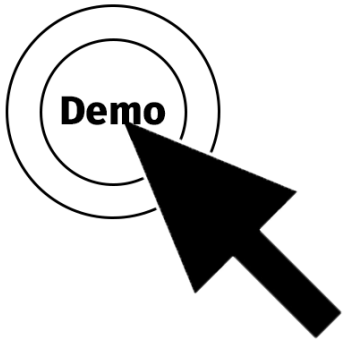
9 Standorte – 100% Know-How

Von E-Commerce über Content- und Digital-Marketing-Services bis hin zur Retail-Kompetenz: Wir sorgen für alle wichtigen E-Business-Disziplinen, vernetzt unter einem Dach.

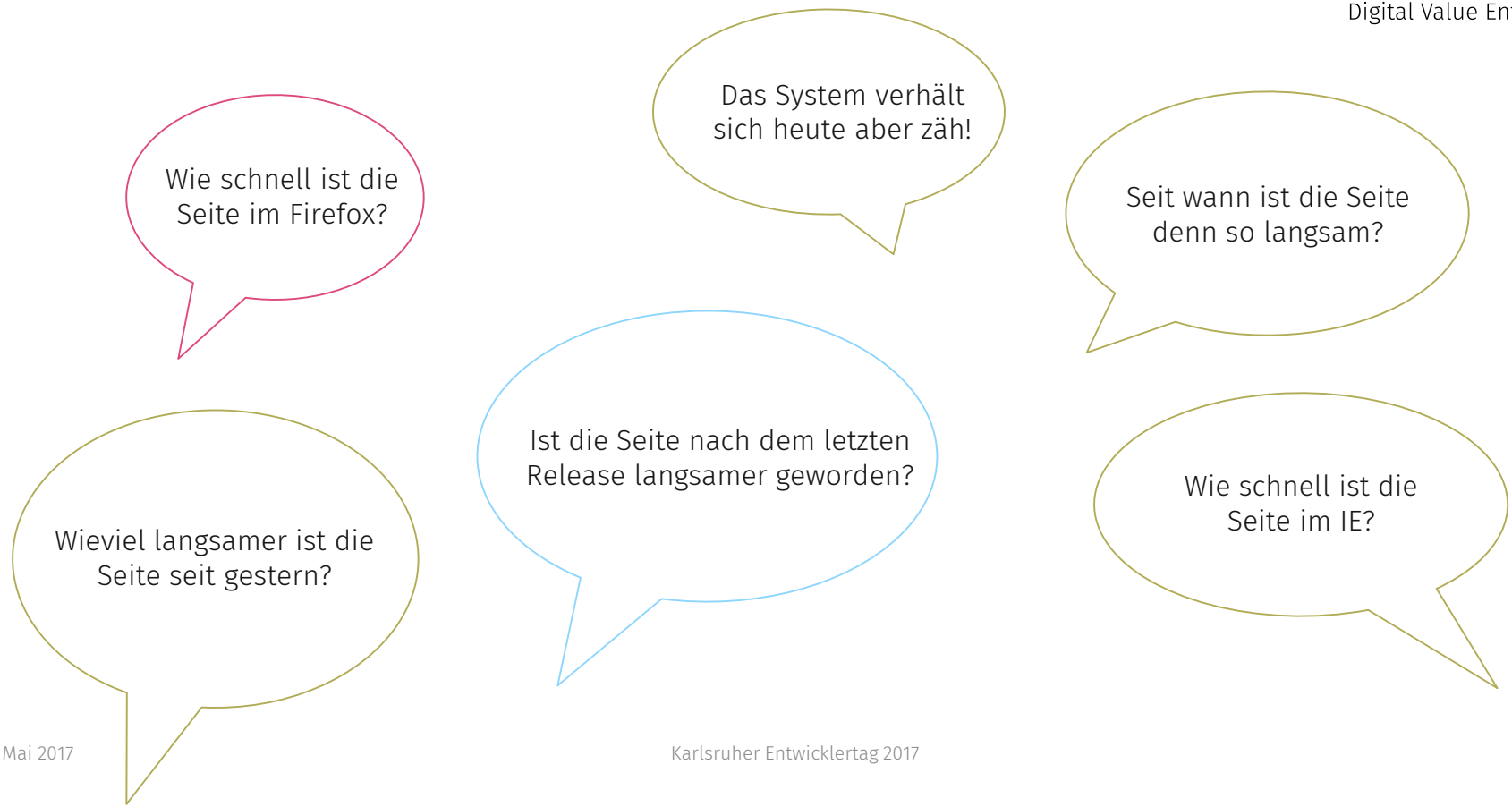
Agiles Projektmanagement

Langjährige Erfahrung im agilen Projektmanagement und arbeiten in allen Projekten mit Scrum.

diva-e Der Demo-Shop



diva-e Motivation



diva-e Eine typische Unterhaltung mit einem Kunden

Die Seite ist heute langsam!

Wie langsam?

Deutlich langsamer als gestern, auf allen Browsern!

Ok, also öffnen Sie die Seite mal in Firefox, drücken Sie F12, wechseln Sie ins Netzwerk-Tab.

???

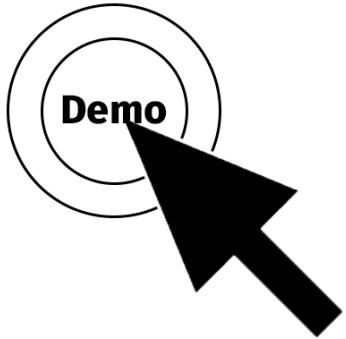
Jetzt laden Sie die Seite neu und lesen mir bitte die Zahlenwerte vor.

1,971 kB

Nein, bitte nur die Zeitangaben...

diva-e Eine typische PO-Tätigkeit

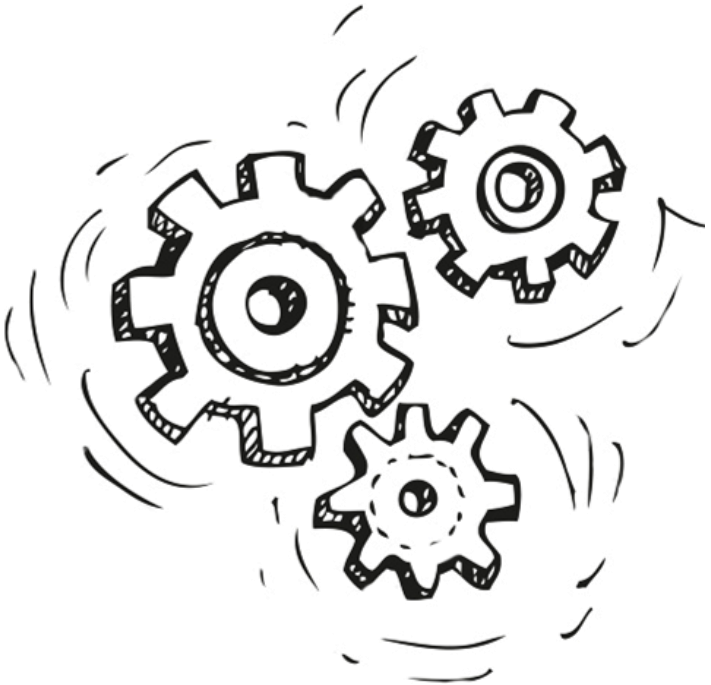
- Firefox öffnen, F12 drücken, in den Netzwerktab wechseln…
- Die Seite öffnen, die Zahlen notieren.
- Oje, vergessen die Caches vorher zu leeren…
- Caches leeren, Seite nochmal öffnen…
- Zahlen notieren.
- Und das ganze jetzt **5 Mal**, weil die Entwickler ja gerne mehrere Messungen hätten!



...aber darauf hat doch wirklich keiner Bock!

Und was machen Entwickler mit Dingen auf die keiner Bock hat?

Automatisieren!



diva-e Ziel

Automatisierte kontinuierliche
und detaillierte Zeitmessung,
während der Weiterentwicklung der Applikation

diva-e Wie komme ich an die Werte?

- Navigation Timing API
  https://www.w3.org/TR/navigation-timing/
- Resource Timing API
  https://www.w3.org/TR/resource-timing/
- User Timing API
  https://www.w3.org/TR/user-timing/

diva-e Navigation Timing API (Teil 1)

- Messwerte findet man unter `performance.timing` als Unix-Millisekunden-Zeitstempel (als `long`):

`navigationStart`: Zeitpunkt nachdem begonnen wurde, das vorherige Dokument zu verlassen.
`unloadEventStart`: Zeitpunkt bevor das unload Ereignis ausgelöst wurde.
`unloadEventEnd`: Zeitpunkt nachdem das vorherige Dokument verlassen wurde.
`redirectStart`: Zeitpunkt der Abholung, die eine Weiterleitung ausgelöst hat.
`redirectEnd`: Zeitpunkt nachdem die letzte Weiterleitungsantwort beendet wurde.
`fetchStart`: Zeitpunkt, an dem die Abholung der Ressource gestartet wurde.
`domainLookupStart`: Zeitpunkt vor dem Domainname Lookup.
`domainLookupEnd`: Zeitpunkt nach dem Domainname Lookup.
`connectStart`: Zeitpunkt vor dem Aufbau der Serververbindung.
`connectEnd`: Zeitpunkt, an dem die Serververbindung beendet wurde.
`secureConnectionStart`: Zeitpunkt als der Handshake für HTTPS begonnen wurde.
…

diva-e Navigation Timing API (Teil 2)

- Messwerte findet man unter `performance.timing` als Unix-Millisekunden-Zeitstempel (als `long`):

…

`requestStart`: Zeitpunkt vor der Serveranfrage.

`responseStart`: Zeitpunkt vor dem Start der Antwort.

`responseEnd`: Zeitpunkt nach dem Ende einer Antwort oder Verbindung.

`domLoading`: Zeitpunkt bevor die Dokumentverfügbarkeit auf "loading" gesetzt wurde.

`domInteractive`: Zeitpunkt bevor die Dokumentverfügbarkeit auf "interactive" gesetzt wurde.

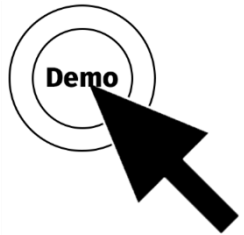
`domContentLoadedEventStart`: Zeitpunkt bevor das DOMContentLoaded Ereignis ausgelöst wurde.

`domContentLoadedEventEnd`: Zeitpunkt nachdem das DOMContentLoaded Ereignis ausgelöst wurde.

`domComplete`: Zeitpunkt bevor das Dokument vollständig verfügbar war.

`loadEventStart`: Zeitpunkt bevor das load Ereignis ausgelöst wurde.

`loadEventEnd`: Zeitpunkt nachdem das load Ereignis beendet wurde.



- Wichtigste Zeitstempel:

  `responseEnd – navigationStart`: HTML komplett geladen.

  `loadEventEnd – navigationStart`: Seite fertig (ohne AJAX).

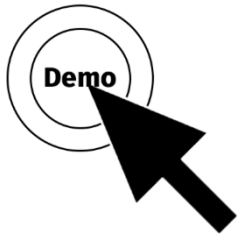
  `ajaxFertig – navigationStart`: Seite wirklich fertig.

- Woher bekommen wir `ajaxFertig`?

diva-e Hooks im verwendeten Framework

- Hook für Angular:

```
angular.getTestability(document.body)
      .whenStable(function(){
            var ajaxFertig = new Date().getTime();
      });
```



- Messwerte findet man unter `performance.getEntries()` als Millisekunden-Zeitstempel, relativ zu `navigationStart`, Nanosekunden-Auflösung (als `double`):

  `name`: Name des Eintrags, meist die URL der Ressource (als `string`).

  `entryType`: Typ des Eintrags, hier „`resource`“ (als `string`).

  `startTime`: Zeitpunkt an dem die Messung für den Eintrag begann.

  `duration`: Dauer des Vorgangs.

  `initiatorType`: Was hat den Request ausgelöst (als `string`)?

  Weitere Felder analog zur Navigation Timing API:

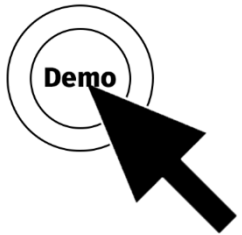
  `redirectStart`, `redirectEnd`, `fetchStart`, `domainLookupStart`, `domainLookupEnd`, `connectStart`, `connectEnd`, `secureConnectionStart`, `requestStart`, `responseStart`, `responseEnd`

- Messwerte findet man unter `performance.getEntries()` als Millisekunden-Zeitstempel, relativ zu `navigationStart`, Nanosekunden-Auflösung (als `double`):
  - `name`: Name des Eintrags, meist die URL der Ressource (als `string`).
  - `entryType`: Typ des Eintrags, hier „`mark`“ (als `string`).
  - `startTime`: Zeitpunkt an dem die Messung für den Eintrag begann.
  - `duration`: Hier immer `0`.

```
performance.mark("start");
// to something
performance.mark("end");
```

- Hook für Angular:

```
angular.getTestability(document.body)
      .whenStable(function(){
            performance.mark("ajaxFertig");
      });
```



- Webtests laufen mit **Selenium**
- **Zeitstempel** werden kontinuierlich erfasst
- Während normalen **Webtests**
- Für ein spezielles **Monitoring**
- Auch komplexe Monitoring und Test-Szenarien lassen sich damit umsetzen.

Selenium

steuert

Firefox

Edge

Chrome

PhantomJS

Internet Explorer

diva-e Selenium Beispiel


```
EventFiringWebDriver driver = new EventFiringWebDriver(new
FirefoxDriver());
driver.register(new AbstractWebDriverEventListener() {
        @Override
        public void afterNavigateTo(String url, WebDriver driver) {
                // extract metrics
        }
});

driver.get("http://localhost:8080/");
// interact with the page
driver.quit();
```

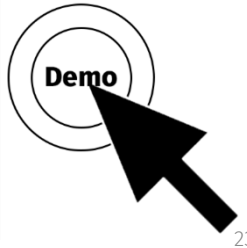
diva-e Prometheus sammelt die Daten

- Prometheus sammelt die Daten als Zeitserien.
- Zeitstempel können mit verschiedenen Tags versehen werden.
- Bei der Abfrage können die Daten verschieden aggregiert werden.
- Wir sammeln in Prometheus auch viele andere (technische) Messwerte: CPU, Heap, …

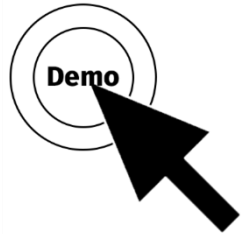
  => Damit lassen sich Schwankungen im Browser gut mit der Auslastung auf Serverseite gegenüber stellen.

diva-e Prometheus beispielhafte Messwerte

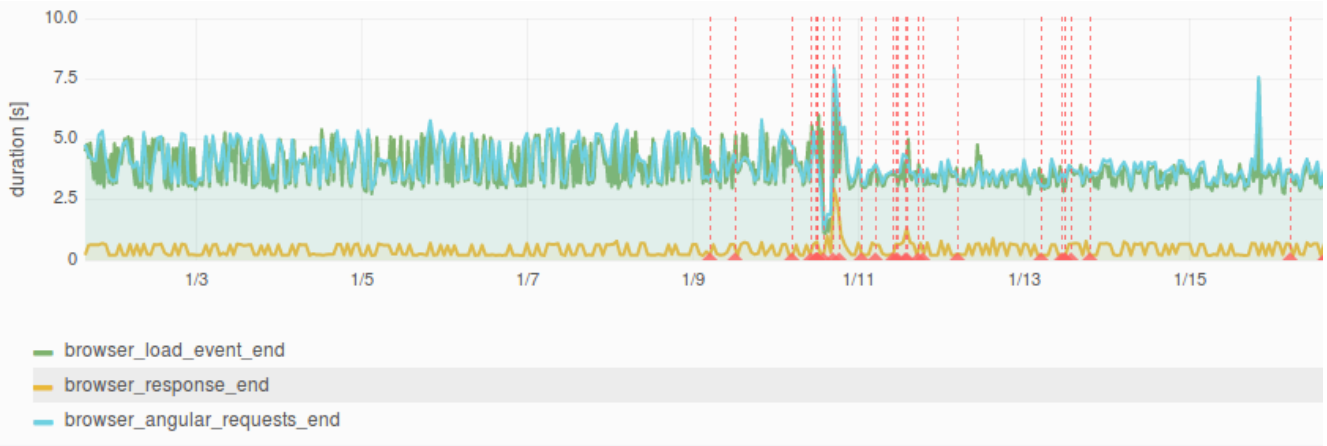

```
browser_response_end{stage="live",browser="firefox",url="http://.../start.html"} 120
browser_load_event_end{stage="live",browser="firefox",url="http://.../start.html"} 517
browser_ajax_fertig{stage="live",browser="firefox",url="http://.../start.html"} 1202
browser_response_end{stage="live",browser="firefox",url="http://.../start.html"} 120
browser_response_end{stage="live",browser="firefox",url="http://.../style.css"} 34
browser_response_end{stage="live",browser="firefox",url="http://.../rest/basket"} 376
```



diva-e Grafana zeigt die Daten an



- Bietet eine Vielzahl von Visualisierungsmöglichkeiten.
- Über Dashboards kann man sich seine persönlichen „Lieblingsgraphen“ zusammen stellen.
- Grafana bietet Authentifizierung und Autorisierung für spezielle Sichten, etwa für den Kunden.

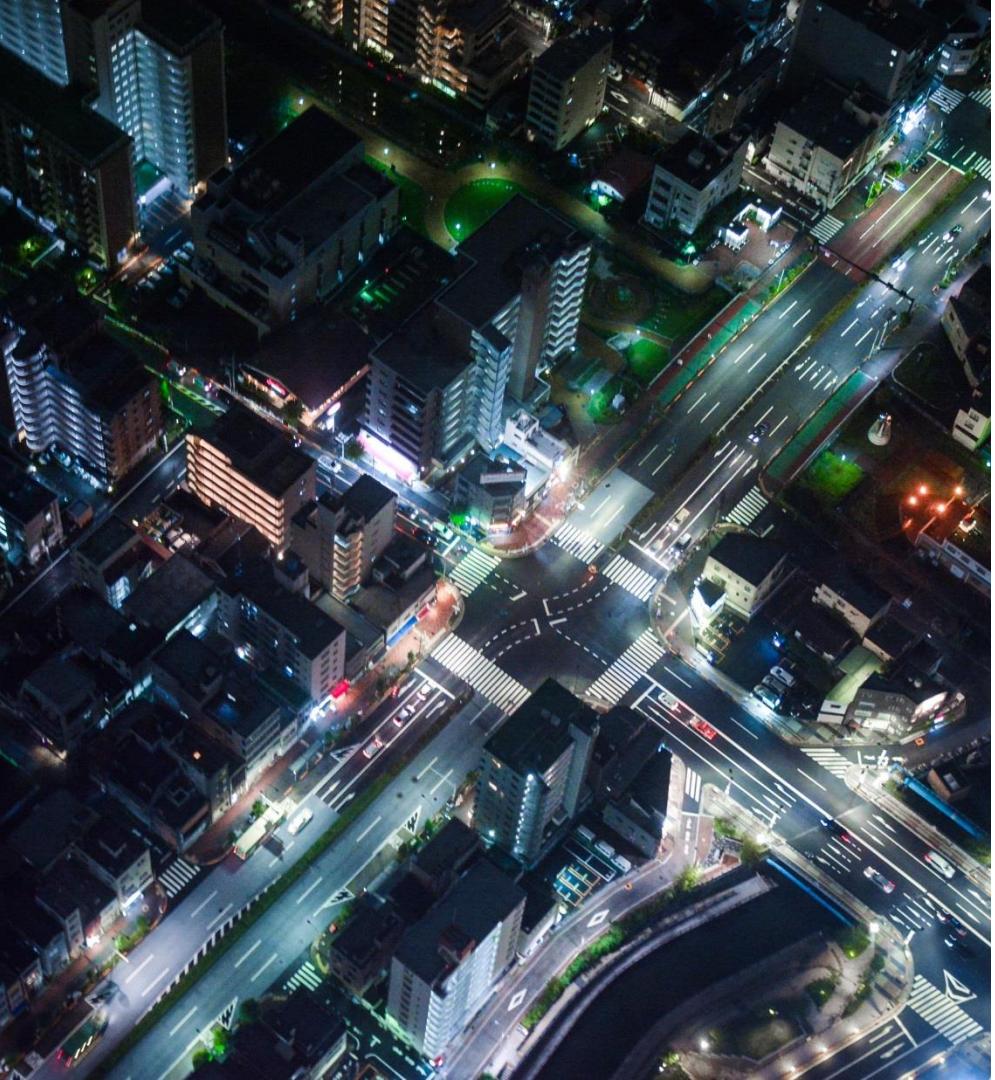


diva-e Fazit

- Kontinuierliche Performancemessungen im Browser haben sich im Projekt bewährt.
- Die gemessenen Werte stimmten recht gut mit den vom „normalen Benutzer“ berichteten Verhalten überein.
- Dem Kunden gefallen die schönen Grafen im Grafana ☺

- Man kann sich aber in der Vielzahl der Messwerte schnell verlieren!

Fragen?



Vielen Dank für Ihre Aufmerksamkeit.

Bis zum nächsten Mal

Adresse
diva-e Digital Value Enterprise GmbH
Office Karlsruhe

Ihr Kontakt
Christian Kumpe
Expert Developer

T +49 721 92060 710
christian.kumpe@diva-e.com
www.diva-e.com