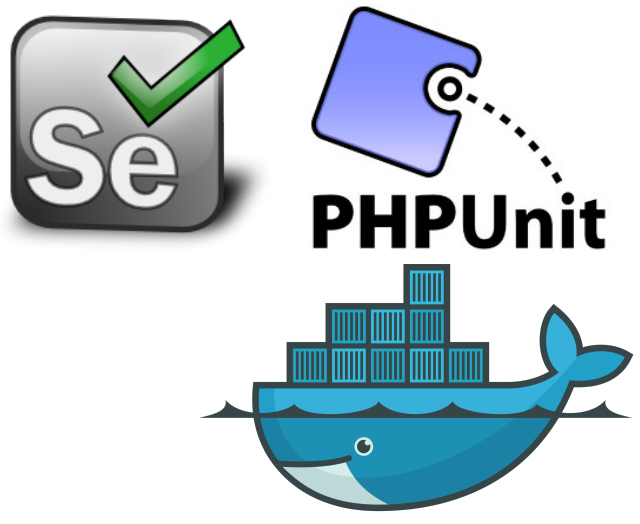
Continuous Web Data Extraction

with



by Robert Koch, e2 communications

for viennaPHP, 2017-02-23

Problem Statement

- extract data that changes daily
- from several login-protected backends
- do analytics / mash-ups / whatsoever

Why Selenium?

- Selenium is established
- real browsers are best for navigation
- could scale via Selenium Grid

Why PHPUnit?

- easy integration
- for each backend use on PHPUnit test class ⟶ extractor
- provides test runner

Why PHPUnit?

- easy integration
- for each backend use on PHPUnit test class ⟶ extractor
- provides test runner

```
vendor/bin/phpunit --group ... --exclude-group ...
```

```
/**
 * @test
 * @group daily
 */
public function extract()
{
    ...
```

Why PHPUnit?

- easy integration
- for each backend use on PHPUnit test class ⟶ extractor
- provides test runner

```
vendor/bin/phpunit --group ... --
exclude-group ...
```

```
/**
 * @test
 * @group daily
 */
public function extract()
{
    ...
```

```
vendor/bin/phpunit --filter <pattern>
```

Why Jenkins?

- Jenkins is already there
- test-runs
  - scheduled
  - on demand
- see logging & execution history

Php <-> Selenium

- `facebook/webdriver`
  - compatible with Selenium server version 2.x and 3.x

- `symfony/dom-crawler`
  - simplifies DOM navigation

Demo Time

1. show test site: https://datatables.net/manual/styling/bootstrap-simple.html
2. run all extractors locally via:
   - `ant run-local`
     - explain `selenium-standalone`
   - `ant run-local -DEXECUTION_FILTER=Plain`
3. boot Jenkins via `docker-compose up`
   - explain docker-setup of 3 containers: `jenkins master`, `php slave` & `selenium standalone`
   - run all extractors on Jenkins
   - connect to Selenium via VNC `127.0.0.1:5900` using password `secret`
   - show screenshots of failing test
4. compare extracting methods
   - Selenium way
   - use HTML source
   - JS for the rescue

Challenges & Findings Part 1

- selectors

Challenges & Findings Part 1

- selectors
- popups

```
protected function saveScreenshot() {
    try {
        $this->driver->takeScreenshot(...);
    } catch (UnexpectedAlertOpenException $e) {
        $this->driver->switchTo()->alert()->dismiss();
        $this->saveScreenshot();
    }
}
```

Challenges & Findings Part 1

- selectors
- popups

```
protected function saveScreenshot() {
    try {
        $this->driver->takeScreenshot(...);
    } catch (UnexpectedAlertOpenException $e) {
        $this->driver->switchTo()->alert()->dismiss();
        $this->saveScreenshot();
    }
}
```

- timeouts - slow backends

```
$quickStatsLink = $this->driver->wait(60, 250)->until(
    WebDriverExpectedCondition::visibilityOfElementLocated(
        WebDriverBy::linkText('Quick Stats')
    )
);
```

```
$this->driver->wait()->until(
    WebDriverExpectedCondition::stalenessOf($resultTable)
);
```

Challenges & Findings Part 2

- date pickers
  - progressively enhanced are great, but …

Challenges & Findings Part 2

- date pickers
  - progressively enhanced are great, but …
  - set via JavaScript

```
$dateString = $dateTime->format('Y-m-d');

$js = <<<JAVASCRIPT
return (function (angular) {
    "use strict";

    var input = document.querySelector('input[ng-model="qf.from"]');
    input.value = "{$dateString}";

    var ngInput = angular.element(input);
    ngInput.triggerHandler('input');

    return input.value;
})(angular);
JAVASCRIPT;

$gotDateString = $this->driver->executeScript($js);
```

What about you?

- Do you like the approach?
- Do you see drawbacks?
- Have you got questions?