Philipp Rieber

Maßgeschneiderte Testdaten

mit Faker und Alice

@bicpi

@bicpi

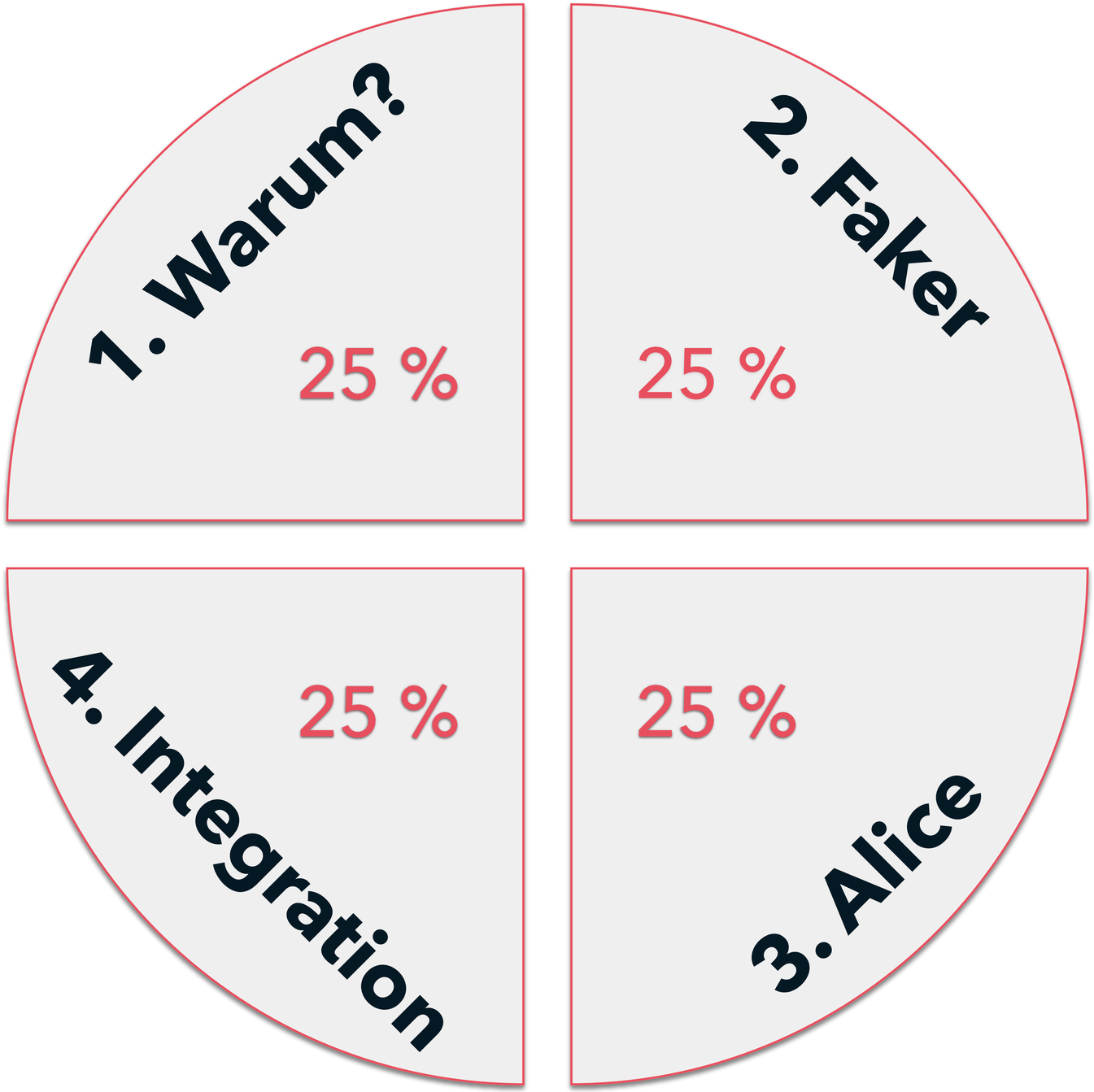
http://philipp-rieber.net

PHP · Symfony · DevOps · Technischer Autor

Ausgabe 3.15 April/Mai
www.phpmagazin.de
S&S
PHPmagazin
PHPmagazin
Deutschland 9,80 €
PHP • JavaScript • Open Web Technologies
Alice und Faker
Maßgeschneiderte Testdaten
im Handumdrehen
EXKLUSIV!
appserver.io
Der Application
Server für PHP!
Exar-
Framework
Aspektorientierung
in PHP
Case Study
Die Real-User-Performance
Google
Google-Dienste im Onlinerechtfokus

Symfony User Group München

@sfugmuc


1. Warum?
25 %
2. Faker
25 %
4. Integration
25 %
25 %
3. Alice

Testdaten, warum?

Testdaten haben mehr Zuwendung verdient

> Fixtures are used to load a controlled set of data into a database. This data can be used for testing or could be the initial data required for the application to run smoothly.

– Symfony Doku

http://symfony.com/doc/current/bundles/DoctrineFixturesBundle

Testdaten für …

… Initialisierung einer Applikation

… automatisierte Tests

```php
class User
{
    public $name;

    public $email;

    public $birthday;

    public $city;
}
```

```php
class User
{
    public $firstName;
    public $lastName;
    public $email;
    public $birthday;
    public $address;
    public $zip;
    public $city;
    public $country;
    public $avatar;
    public $homepage;
    public $timezone;
    public $description;
    public $language;
    public $joinedAt;
    public $isActive;
}
```

```php
class User
{
    public $name;

    public $email;

    public $birthday;

    public $city;
}
```

Name	E-Mail	Geburtstag	Stadt
?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?
?	?	?	?

Name	E-Mail	Geburtstag	Stadt
Hans Wurst	hans@wurst.de	17.09.1982	Berlin
Max Müller	example@example.org	17.09.1966	buxtehude
sdfsdf	werwer@asdaasd.de	01.01.1970	blabla
qwe sdfsd	example@example.org	01.01.1970	www
asd add	example@example.org	01.01.1970	kkk

Name	E-Mail	Geburtstag	Stadt
Name 1	user-1@example.org	01.01.1970	City 1
Name 2	user-2@example.org	01.01.1970	City 2
Name 3	user-3@example.org	01.01.1970	City 3
Name 4	user-4@example.org	01.01.1970	City 4
Name 5	user-5@example.org	01.01.1970	City 5

Name	E-Mail	Geburtstag	Stadt
Astrid Buchholz	astrid.buchholz@web.de	06.01.1984	Böblingen
Korbinian Stiebitz	k.striebitz@carter.biz	-	-
Ruth Paffrath	paffrath.ruth@gmail.com	-	Köln
Antonio Mälzer	a.maelzer@yahoo.com	03.07.1985	Hettstedt
Sophia Scheibe	tiana23@lakin.com	16.01.1994	Eutin

Testdaten

Müssen nur möglichst echt ausschauen

Unterstützung in Symfony?

> To populate the database with some initial data, we could create a PHP script, or execute some SQL statements with the mysql program.
>
> But as the need is quite common, there is a better way with symfony: create YAML files in the `data/fixtures/` directory and use the `doctrine:data-load` task to load them into the database.

– symfony Doku

> To populate the database with some initial data, we could create a PHP script, or execute some SQL statements with the mysql program.
>
> But as the need is quite common, there is a better way with symfony: create YAML files in the `data/fixtures/` directory and use the `doctrine:data-load` task to load them into the database.
>
> *– symfony Doku*

http://symfony.com/legacy/doc/jobeet/1_4/en/03?orm=Doctrine#chapter_03_the_initial_data

```yaml
# data/fixtures/User.yml
User:
    user_tim:
        name: Tim Buktu
        email: ...
    user_klara:
        name: Klara Fall
        email: ...
```

```
# data/fixtures/User.yml
User:
    user_tim:
        name: Tim Buktu
        email: ...
    user_klara:
        name: Klara Fall
        email: ...
<?php for ($i = 1; $i < 10; $i++): ?>
    user_<?php echo $i; ?>:
        name: Name <?php echo $i; ?>
        email: ...
<?php endfor; ?>
```

```
$ php symfony doctrine:data-load
```

OK, und Symfony2?

> Symfony has no built in way to manage fixtures but Doctrine2 has a library to help you write fixtures.

– Symfony Doku

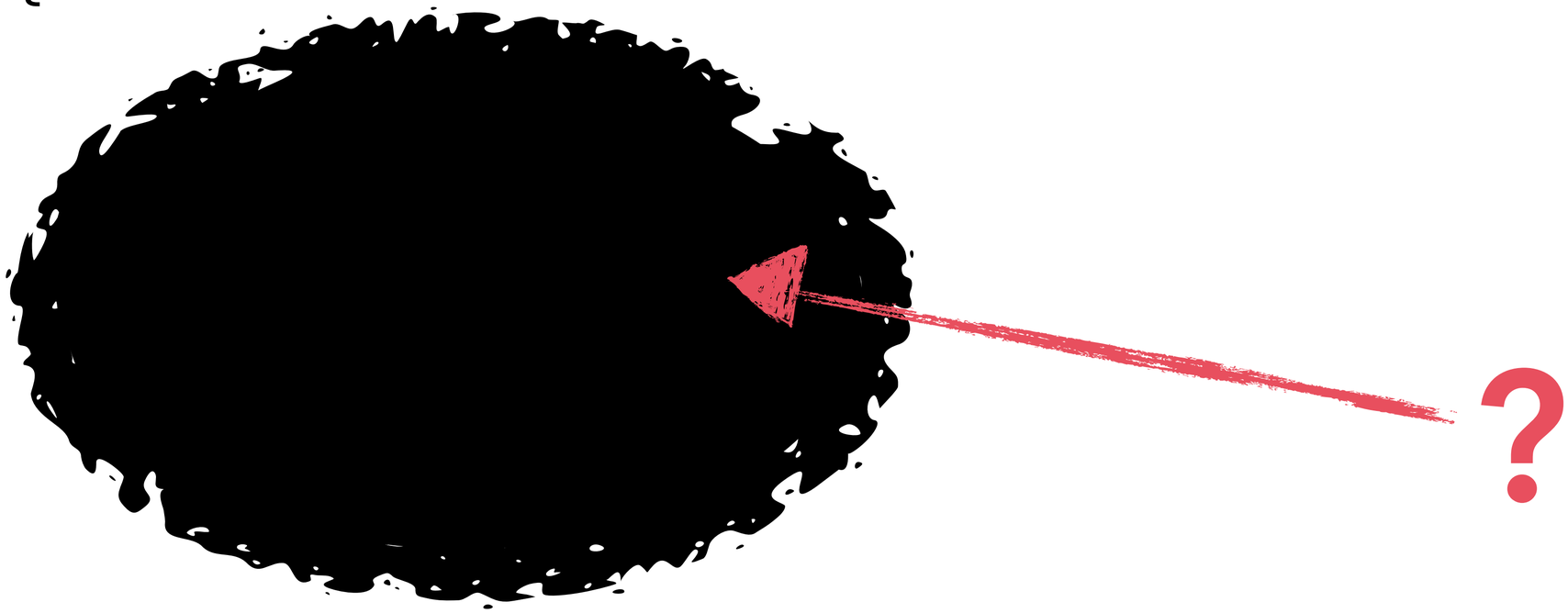
http://symfony.com/doc/current/bundles/DoctrineFixturesBundle/index.html

```php
// src/AppBundle/DataFixtures/ORM/LoadUserData.php
class LoadUserData extends AbstractFixture
{
    public function load(ObjectManager $manager)
    {
        // Testdaten erstellen ...
    }
}
```

```
$ bin/console doctrine:fixtures:load
```

```php
// src/AppBundle/DataFixtures/ORM/LoadUserData.php
class LoadUserData extends AbstractFixture
{
    public function load(ObjectManager $manager)
    {

    }
}
```



```php
// src/AppBundle/DataFixtures/ORMLoadUserData.php
class LoadUserData extends AbstractFixture
{
    public function load(ObjectManager $manager)
    {
        for($i = 1; $i <= 3; $i++) {
            $user = (new User())
                ->setName("Name $i")
                ->setEmail("user-$i@example.org")
                ->setBirthday('1970-01-01')
                ->setCity("City $i");
            $manager->persist($user);
        }
        $manager->flush();
    }
}
```

Eigene Code-Snippets für Fake-Daten

Eindeutiger Name

```
$names = ['Tim Buktu', 'Klara Fall', ...];

shuffle($names);

$uniqueName = array_pop($names);
```

Zufällige Zeitzone

```php
$timezones = \DateTimeZone::listIdentifiers();

$timezone = $timezones[array_rand($timezones)];
```

Zufälliger “Letzter Login” – muss in der Vergangenheit liegen

```
$someHoursAgo = time() - mt_rand(0, 3600*24);

$lastLogin = \DateTime::createFromFormat('U', $someHoursAgo);
```

Zufällige Anzahl an Zufalls-Elementen aus einem Wertepool

```
$skills = ['PHP', 'SQL', 'NoSQL', 'Symfony', /*...*/];

$randKeys = (array) array_rand($skills, mt_rand(1, 3));
shuffle($randKeys);

$randomSkills = array_map(
    function ($key) use ($skills) {
        return $skills[$key];
    },
    $randKeys
);
```

Lokalisierung?

Faker

"PHP-Bibliothek zur Erzeugung realistischer Testdaten"

@francoisz – François Zaninotto

```
$ composer req fzaninotto/Faker
```

Faker Generator-Instanz erzeugen

```
require_once 'vendor/autoload.php';

$faker = Faker\Factory::create();
```

Fake-Daten erzeugen

```
$faker->name; // z.B. 'Lucy Hessel'

$faker->email; // z.B. 'inienow@rippin.net'

$faker->url; // z.B. 'http://reichert.com/sit-autem'

$faker->currencyCode; // z.B. 'CHF'
```

Fake-Daten erzeugen

```
$faker->numberBetween(13, 49) // z.B. 29

$faker->randomElements(['pi','pa','po'], 2) // z.B. ['pi','po']

$faker->lexify('?????') // z.B. 'gjwzz'
```

Personen-Namen
Zeitzonen
MIME-Types
Dateiendungen
Straßen-Name
Währungen
Länder
Städte
Domains
User-Agent-Strings
Firmen-Namen
E-Mail-Adressen
MAC-Adressen
UUIDs

ISBNs
Kreditkarten
IP-Adressen
IBANs
Farbcodes
EANs
Telefonnummern
Bilder
Sprachen
Texte
Zufalls-Zahlen
Datumswerte
Zeitzonen
Locales

Passwörter
Zufalls-Strings
Postleitzahlen
Geo-Koordinaten
Benutzernamen
Dateien
URLs
Slugs
Booleans
Zufallselemente
BICs
Adressen
Staaten
…

Faker-Vokabular

Formatter · Provider · Modifier

Formatter – ohne Argumente

```
$faker->ipv4;  // e.g. '116.132.63.32'

$faker->ip4(); // e.g. '116.132.63.32'
```

Formatter - mit Argumenten

```
$faker->dateTimeBetween('-63 years',  '-18 years');
// z.B. \DateTime('1982-12-09 13:42:22')
```

```
$faker->imageUrl(640, 480);
// z.B. 'http://lorempixel.com/640/480/'
```

```
$faker->numerify('#####');
// z.B. '72839'
```

Provider

```
class Person

function title();
function name();
function suffix();
...
```

```
class Internet

function ipv6();
function email();
function password();
...
```

```
class Address

function street();
function postcode();
function country();
...
```

```
class DateTime

...
```

```
class Image

...
```

```
...
```

Modifier

unique・optional

Eindeutige Daten

```
$faker->email;
```

Eindeutige Daten

```
$faker->email;

$faker->unique()->email;
```

Optionale Daten

```
$faker->text(150);
```

Optionale Daten

```
$faker->text(150);

$faker->optional(.35)->text(150);
```

Optionale Daten

```
$faker->text(150);

$faker->optional(.35)->text(150);

$faker->optional(.35, 'Keine Beschreibung vorhanden’)->text(150);
```

Lokalisierung

Lokalisierung

```
$de_DE = Faker\Factory::create('de_DE');
```

Lokalisierung

```
$de_DE = Faker\Factory::create('de_DE');
$de_DE->name; // z.B. 'Ingo Barth'
$de_DE->city; // z.B. 'Bremen'
$de_DE->phonenumber; // z.B. '07239 14169'
```

Lokalisierung

```
$de_DE = Faker\Factory::create('de_DE');
$de_DE->name; // z.B. 'Ingo Barth'
$de_DE->city; // z.B. 'Bremen'
$de_DE->phonenumber; // z.B. '07239 14169'

$fr_FR = Faker\Factory::create('fr_FR');
$fr_FR->name; // z.B. 'Hélène Moreau'
$fr_FR->city; // z.B. 'Lacroix-les-Bains'
$fr_FR->phonenumber; // z.B. '08 16 15 65 48'
```

Lokalisierung

```
$de_DE = Faker\Factory::create('de_DE');
$de_DE->name; // z.B. 'Ingo Barth'
$de_DE->city; // z.B. 'Bremen'
$de_DE->phonenumber; // z.B. '07239 14169'

$fr_FR = Faker\Factory::create('fr_FR');
$fr_FR->name; // z.B. 'Hélène Moreau'
$fr_FR->city; // z.B. 'Lacroix-les-Bains'
$fr_FR->phonenumber; // z.B. '08 16 15 65 48'

$ja_JP = Faker\Factory::create('ja_JP');
$ja_JP->name; // z.B. '廣川 くみ子'
$ja_JP->city; // z.B. '小泉市'
$ja_JP->phonenumber; // z.B. '80-0014-1949'
```

Lokalisierung II

```
$fr_FR->departmentName; // z.B. 'Pyrénèes-Atlantiques'

$sv_SE->personalIdentityNumber; // z.B. '740306-0077'

$kk_KZ->businessIdentificationNumber; // z.B. '160341348678'

$ne_NP->district; // z.B. 'Khotang'

$ko_KR->borough; // z.B. '송파구'
```

Eigene Provider & Formatter

Eigener Provider

```
class MyProvider
{
    public function encodedPassword($plainPassword)
    {
        $digest = hash('sha512', $plainPassword, true);
        for ($i = 1; $i < 1000; $i++) {
            $digest = hash('sha512', $plainPassword, true);
        }

        return base64_encode($digest);
    }
}
```

Eigener Provider

```
class MyProvider
{
    public function encodedPassword($plainPassword)
    {
        // ...
    }
}

$faker->addProvider(new MyProvider());

$faker->encodePassword('mySecretPassword');
```

Dritt-Provider

Cron-Ausdrücke, Kleidergrößen, IMEI-Nummern,
Bitcoin-Adressen, Sport-Teams, Emoticons …

Seeding

Seeding

```
$faker = Faker\Factory::create('de_DE');
$faker->seed(1);
```

Seeding

```
$faker = Faker\Factory::create('de_DE');
$faker->seed(1);

$faker->name; // 'Adriana Rudolph'
$faker->name; // 'Zdenka Stumpf'
$faker->name; // 'Natascha Ullmann'
```

Seeding

```php
$faker = Faker\Factory::create('de_DE');
$faker->seed(1);

$faker->name; // 'Adriana Rudolph'
$faker->name; // 'Zdenka Stumpf'
$faker->name; // 'Natascha Ullmann'
```

```php
$faker = Faker\Factory::create('de_DE');
$faker->seed(1);

$faker->name; // 'Adriana Rudolph'
$faker->name; // 'Zdenka Stumpf'
$faker->name; // 'Natascha Ullmann'
```

Faker integrieren

Persistenz

```
$populator = new Faker\ORM\Doctrine\Populator(
    $faker,
    $entityManager
);

$populator->addEntity('Acme\Entity\User', 50);

$primaryKeys = $populator->execute();
```

Faker im Symfony-Container konfigurieren

```
app.faker:
    class: Faker\Generator
    factory: [ Faker\Factory, create ]
    arguments:
        - de_DE
    calls:
        - [ seed, [ 1 ] ]
```

Persistenz in Symfony

```
class LoadData extends AbstractFixture implements ContainerAwareInterface
{
    public function load(ObjectManager $manager)
    {
        $faker = $this->container->get('app.faker');
        for($i = 1; $i <= 3; $i++) {
            $user = (new User())
                ->setFirstName($faker->firstName)
                ->setLastName($faker->lastName);
                ->setEmail($faker->email);
                ->setCity($faker->city);
            $manager->persist($user);
        }
        $manager->flush();
    }
}
```

Trait für PHPUnit

```php
trait FakerSetup
{
    private $faker;

    /**
     * @before
     */
    public function setUpFaker()
    {
        $this->faker = \Faker\Factory::create();
        $this->faker->seed(1);
    }
}
```

Trait für PHPUnit

```
class BookTest extends \PHPUnit_Framework_TestCase {
    use FakerSetup;

    public function testSomething() {
        $book = $this->createBook();
        // test something ...
    }

    private function createBook() {
        return (new Book())
            ->setIsbn($this->faker->isbn13)
            ->setAuthor($this->faker->name)
            ->setDescription($this->faker->text);
    }
}
```

Mit Faker gelöst

- Realistische Daten
- Datenbeschaffung
- Bedingungen erfüllen
- Zufallslogik
- Reproduzierbar
- Lokalisierung

Was fehlt? Support bei:

- Erstellen/Persistieren von Objekten mit Fake-Daten
- Organisation

Foto © Florian Kohlert; http://www.floriankohlert.de

Alice

"PHP-Bibliothek zur Erzeugung und Organisation von Testdaten"

@seldaek – Jordi Boggiano & Tim Shelburne

```
$ composer req nelmio/alice
```

```php
require_once 'vendor/autoload.php';

$loader = new \Nelmio\Alice\Fixtures\Loader();
```

```php
require_once 'vendor/autoload.php';

$loader = new \Nelmio\Alice\Fixtures\Loader();

$objects = $loader->load(__DIR__.'/fixtures.yml');
```

Fixtures definieren

```
# fixtures.yml
Acme\Entity\User:
    user_max:
        name: Tim Buktu
        email: tim@buktu.de
        ...
    user_erika:
        name: Klara Fall
        email: klara@fall.de
        ...
```

Alice Code

Objekt-Range

```
Acme\Entity\User:
    user_{1..50}:
        name: John Doe
        email: john@doe.net
        ...
```

Dynamische Daten

```
Acme\Entity\User:
    user_{1..50}:
        name: John Doe <current()>
        email: john-<current()>@doe.net
```

Faker Integration

```
<fakerFormatter()>
```

Faker Integration

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>
```

Faker Integration

```
Acme\Entity\User:
    user_{1..50}:
        ...
        birthday: <dateTimeBetween('-75 years', '-13 years')>
```

Variablen

```
Acme\Entity\User:
    user_{1..50}:
        ...
        created: <dateTimeBetween('-2 months', '-2 hour')>
        updated: <dateTimeBetween($created, '-1 hour')>
```

Parameter

```
parameters:
    domain: acme.com

Acme\Entity\User:
    user_{1..50}:
        ...
        email: <username()>@<{domain}>
```

Faker in Faker

```
Acme\Entity\User:
    user_{1..50}:
        ...
        skills: <randomElements(
                    ['PHP', 'SQL', 'NoSQL', ‘Symfony'],
                    2
                )>
```

Faker in Faker

```
Acme\Entity\User:
    user_{1..50}:
        ...
        skills: <randomElements(
                    ['PHP', 'SQL', 'NoSQL', ‘Symfony'],
                    $fake('numberBetween', null, 0, 3)
                )>
```

Eindeutige Felder

```
Acme\Entity\User:
    user_{1..50}:
        ...
        email (unique): <email()>
```

Optionale Felder

```
Acme\Entity\User:
    user_{1..50}:
        ...
        description: 35%? <text()>
```

Lokalisierung

```
Acme\Entity\User:
    user_{1..50}:
        name: <de_DE:name()>
        city: <fr_FR:city()>
```

Konstruktor

```
AppBundle\Entity\Money:
    money_{1..10}:
        __construct:
            - <randomFloat()>
            - <currencyCode()>
```

Konstruktor

```
AppBundle\Entity\Money (local):
    money_{1..10}:
        __construct:
            - <randomFloat()>
            - <currencyCode()>
```

Objekt-Referenzen

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>

Acme\Entity\Team:
    team_red:
        members: ['@user_11', '@user_38', '@user_41']
    team_blue:
        ...
```

Objekt-Referenzen II

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>

Acme\Entity\Team:
    team_red:
        members: ['@user_*', '@user_*', '@user_*']
    team_blue:
        ...
```

Objekt-Referenzen III

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>

Acme\Entity\Team:
    team_red:
        members: 3x @user_*
    team_blue:
        ...
```

Objekt-Referenzen IV

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>

Acme\Entity\Team:
    team_red:
        members: <numberBetween(0, 3)>x @user_*
    team_blue:
        ...
```

Objekt-Referenzen V

```
Acme\Entity\User:
    user_{1..50}:
        name: <name()>
        email: <email()>

Acme\Entity\Team:
    team_red:
        members: 3x @user_*
    team_blue:
        members: @team_red->members
```

Persistenz

```
$loader = new \Nelmio\Alice\Fixtures\Loader();
$objects = $loader->load(__DIR__.'/fixtures.yml');
```

Persistenz

```
$loader = new \Nelmio\Alice\Fixtures\Loader();
$objects = $loader->load(__DIR__.'/fixtures.yml');

$persister = new \Nelmio\Alice\Persister\Doctrine($objectManager);
$persister->persist($objects);
```

Persistenz: Shortcut

```
$objects = Nelmio\Alice\Fixtures::load(
    [
        __DIR__.'/fixtures/*.yml',
        __DIR__.'/res/User.yml'
    ],
    $objectManager
);
```

Da ist noch mehr

Templates · Datei-Includes · Processors …

Einschränkungen

- Große Datenmengen
- Komplexe Fake-Daten
- Konsistente Daten-Sets
- Langsame Release-Zyklen

Integration

Symfony, Behat & mehr

Symfony-Bundle

KnpLabs/rad-fixtures-load

Lädt Fixture-Dateien aus:

`*Bundle/Resources/fixtures/orm/*.yml`

```
$ app/console rad:fixtures:load
```

```
$ app/console rad:fixtures:load --reset-schema \
                                --locale=fr_FR \
                                --bundle=AppBundle \
                                --bundle=BlogBundle \
                                --filter=client
```

Eigener Faker-Provider

```
app.my_provider:
    class: AppBundle\Faker\MyProvider
    tags:
        - { name: knp_rad_fixtures_load.provider }
```

Behat Integration

KnpLabs/FriendlyContexts

EntityContext

```
Scenario: ...
    Given there are 33 users
    When ...
    Then ...
```

```
Scenario: ...
    Given the following users:
        | username |
        | tic      |
        | tac      |
        | toe      |
    When ...
    Then ...
```

```
@reset-schema
Scenario: ...
    Given there are 33 users
    When ...
    Then ...
```

```yaml
# behat.yml
default:
    # ...
    extensions:
        Knp\FriendlyContexts\Extension:
            entities:
                namespaces:
                    - AppBundle
                    - BlogBundle
```

AliceContext

```yaml
# behat.yml
default:
    # ...
    Knp\FriendlyContexts\Extension:
        alice:
            fixtures:
                use_case: path/to/use-case.yml
```

```
@alice(use_case)
Scenario: ...
    Given ...
    When ...
    Then ...
```

```yaml
# behat.yml
default:
    # ...
    Knp\FriendlyContexts\Extension:
        alice:
            fixtures:
                User: path/to/users.yml
```

```
@alice(User)
Scenario: ...
    Given ...
    When ...
    Then ...
```

```
@alice(use_case) @alice(User)
Scenario: ...
    Given ...
    When ...
    Then ...
```

Weitere Integrationen

Faker CLI

Faker für die Kommandozeile

```
$ faker postcode --locale en_GB
G4 1FW
B51 4KL
E15 5ED
GF9 9AT
L55 9WV
DM5M 2UR
...
```

```
$ faker creditCardNumber --count 1 | pbcopy
```

Faker.js

Node・Browser・Hosted API Microservice

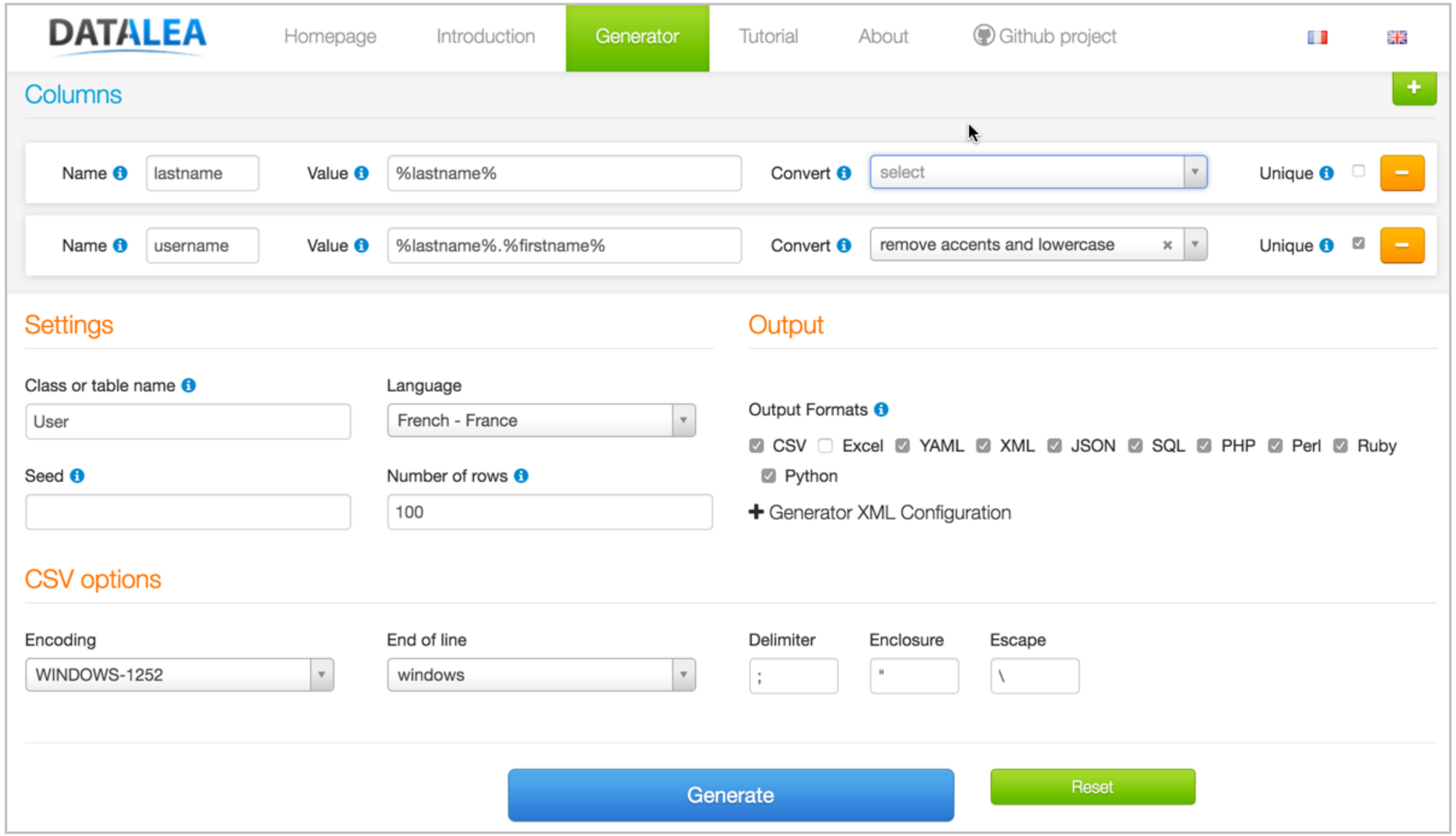
```
$ curl 'http://faker.hook.io?property=image.avatar'
```

```
$ curl 'http://faker.hook.io?property=image.avatar'

=> 'https://s3.amazonaws.com/uifaces/faces/twitter/ovall/128.jpg'
```

Testdaten-GUI

http://datalea.spyrit.net



Faker + Alice

& verschiedene Integrationen

= “Testdaten-Stack”

Noch Fragen?

@bicpi

http://philipp-rieber.net

Talk bewerten:

https://joind.in/talk/f05b4

Links

- Faker: https://github.com/fzaninotto/Faker
- Alice: https://github.com/nelmio/alice

- https://github.com/willdurand/BazingaFakerBundle
- https://github.com/KnpLabs/rad-fixtures-load
- https://github.com/hautelook/AliceBundle
- https://github.com/h4cc/AliceFixturesBundle
- https://github.com/KnpLabs/FriendlyContexts

- https://github.com/bit3/faker-cli
- https://github.com/marak/Faker.js & http://marak.com/faker.js
- http://datalea.spyrit.net