

Brancher AnyBlok à 14 ans d'historique métier

Une histoire terrifiante de PHP, MySQL et MsSQL

Jean-Sébastien SUZANNE et Hugo QUEZADA



2 novembre 2019

Qui sommes nous ?


Sensee.

**lentilles
moinscheres.com**

Jean-Sébastien Suzanne

- Répond aussi au nom de PAPABLOK
-  @jssuzanne
-  js.suzanne@sensee.com

Hugo Quezada

- Répond seulement au nom de Hugo
-  h.quezada@sensee.com

Existant



14 ans de code...

Existant

- Code legacy en PHP5 (pas de troll SVP)
- Pas de framework, beaucoup de code, très peu d'objet

14 ans de code...

Existant

- Code legacy en PHP5 (pas de troll SVP)
- Pas de framework, beaucoup de code, très peu d'objet
- Pas de tests
- Pas de CI

14 ans de code...

Existant

- Code legacy en PHP5 (pas de troll SVP)
- Pas de framework, beaucoup de code, très peu d'objet
- Pas de tests
- Pas de CI
- Pas d'ORM

14 ans de code...

Existant

- Code legacy en PHP5 (pas de troll SVP)
- Pas de framework, beaucoup de code, très peu d'objet
- Pas de tests
- Pas de CI
- Pas d'ORM



Une BdD un peu complexe...

Existant

- Schémas de base de données multiples

Une BdD un peu complexe...

Existant

- Schémas de base de données multiples
- Écosystème avec plusieurs SGBD (MySQL, MsSQL) souvent sans API

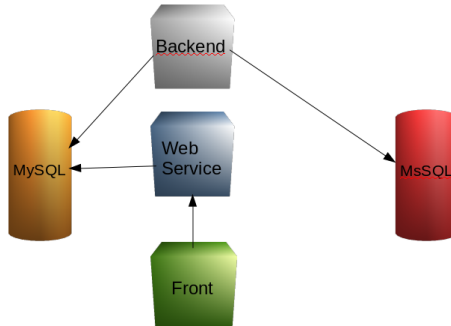


Figure: Schéma simplifié

Vision finale



L'objectif

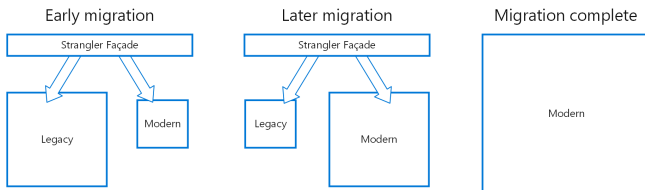
Vision finale

- Un code python propre et testé
- Une API simple et uniforme, utilisé dans tous nos projets
- Une application plus proche des standards actuels
- Un projet plus séduisant et plus attrayant pour des potentiels futurs développeurs

La stratégie

Vision finale

- Modèle d'étranglement (Strangle pattern)



- Mapping des tables avec un nommage plus clair
- Développement piloté par les tests (TDD)

Pourquoi avoir choisi AnyBlok ?



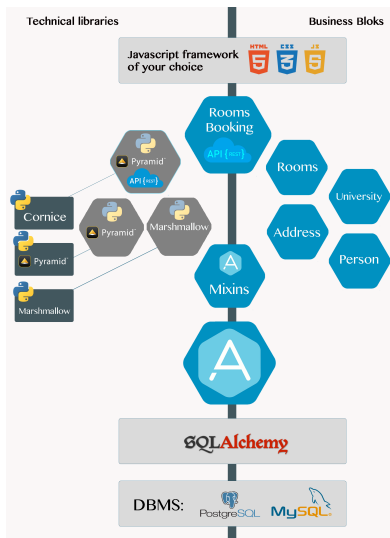
AnyBlok: Présentation

Pourquoi avoir choisi AnyBlok ?

- Python 3.6 et plus
- MPL2
- PyPi
- Modulaire
- Dépendances fiables (SQLAlchemy, Alembic, ...)
- Possibilité de reprendre une base existante.

AnyBlok: Structure

Pourquoi avoir choisi AnyBlok ?



AnyBlok: Écrire des tests pour notre Model

Pourquoi avoir choisi AnyBlok ?

```
import pytest

@pytest.mark.usefixtures('rollback_registry')
class TestProduct:

    def test_create_new_product(self, rollback_registry):
        rollback_registry.LMC.Product.insert(
            label="A product label",
            code="A product code",
        )
        assert rollback_registry.Product.query().count() == 1
        assert (
            rollback_registry.Product.query().one().code ==
            "A product code"
        )
```


AnyBlok: Définir le Model

Pourquoi avoir choisi AnyBlok ?

```
from anyblok import Declarations
from anyblok.column import String, Integer

@Declarations.register(Declarations.Model.LMC)
class Product:

    id = Integer(primary_key=True)
    code = String(nullable=False, index=True)
    label = String(nullable=False)
```

AnyBlok: Définir un Model sur une table existante

Pourquoi avoir choisi AnyBlok ?

```
from anyblok import Declarations
from anyblok.column import String, Integer

@Declarations.register(Declarations.Model.LMC, tablename="ProdDef")
class Product:

    id = Integer(primary_key=True, db_column_name="IDD")
    code = String(nullable=False, index=True, db_column_name="Attr1")
    label = String(nullable=False, db_column_name="Display")
```

AnyBlok: Écrire des tests pour notre API

Pourquoi avoir choisi AnyBlok ?

```
import pytest

@pytest.mark.usefixtures('rollback_registry')
class TestApiProduct:

    def test_addresses_post(self, rollback_registry, webserver):
        response = webserver.post_json('/api/v1/lmc/products',
                                       [{'code': 'C1', 'label': 'My product'}])
        assert response.status_code == 200
        assert response.json_body[0].get('code') == 'C1'
        assert rollback_registry.LMC.Product.query().filter_by(code='C1').one()

    def test_addresses_get(self, rollback_registry, webserver):
        """Address GET /api/v1/addresses"""
        rollback_registry.LMC.Product.insert(
            code="C1", label="My product")
        response = webserver.get('/api/v1/lmc/products')
        assert response.status_code == 200
        assert len(response.json_body) == 1
        assert response.json_body[0].get('code') == 'C1'
```

AnyBlok: Écrire une API REST

Pourquoi avoir choisi AnyBlok ?

```
from anyblok_pyramid_rest_api.crud_resource import (
    CrudResource, resource)

@resource(
    collection_path='/api/v1/lmc/products',
    path='/api/v1/lmc/products/{id}',
    installed_blok=current_blok()
)
class LensProductResource(CrudResource):
    model = "Model.LMC.Product"
```

Évolutions et intégrations dans AnyBlok

Compatibilité avec MySQL

Évolutions et intégrations dans AnyBlok

- Deux semaines de travail
- Beaucoup de recherches et d'incompréhensions

Compatibilité avec MySQL : Les tests unitaires

Évolutions et intégrations dans AnyBlok

- Mise à jour de la configuration Travis-CI
- Activer le mode transactionnel de innoDB
- Les commits implicites

Compatibilité avec MySQL : Limitations

Évolutions et intégrations dans AnyBlok

- Pas de Python 3.5
- Contrainte exclusive à PostgreSQL
- Datetime naïves
- Pas de chiffrement sur les colonnes de type UUID
- Pas de véritable Booléen

Compatibilité avec MariaDB : Limitations

Évolutions et intégrations dans AnyBlok

- Pas de Python 3.5
- Contrainte exclusive à PostgreSQL
- Datetime naïves
- Pas de chiffrement sur les colonnes UUID
- Pas de véritable Booléen

Compatibilité avec MariaDB : Limitations

Évolutions et intégrations dans AnyBlok

- Pas de Python 3.5
- Contrainte exclusive à PostgreSQL
- Datetime naïves
- Pas de chiffrement sur les colonnes UUID
- Pas de véritable Booléen
- Taille des clés primaires plus petite
- Pas de colonnes JSON

Compatibilité avec MsSQL : Limitations

Évolutions et intégrations dans AnyBlok

- Pas de Python 3.5
- Contrainte exclusive à PostgreSQL

Compatibilité avec MsSQL : Limitations

Évolutions et intégrations dans AnyBlok

- Pas de Python 3.5
- Contrainte exclusive à PostgreSQL
- Lent... Très très lent...

Définition de schémas

Évolutions et intégrations dans AnyBlok

Choix :

- Programmatique ou par configuration
- Poser sur un modèle ou un namespace
- Ajout de suffixes ou préfixes pour les tests

Définition de schémas

Évolutions et intégrations dans AnyBlok

Choix :

- Programmatique ou par configuration
- Poser sur un modèle ou un namespace
- Ajout de suffixes ou préfixes pour les tests

Problématiques :

- Génération de foreign key
- Migration

Lier un Model à un schéma

Évolutions et intégrations dans AnyBlok

Dans le Model :

```
from anyblok import Declarations
from anyblok.column import String, Integer

@Declarations.register(Declarations.Model.LMC, tablename="ProdDef")
class Product:
    __db_schema__ = "Lens"

    id = Integer(primary_key=True, db_column_name="IDD")
    code = String(nullable=False, index=True, db_column_name="Attr1")
    label = String(nullable=False, db_column_name="Display")
```

Par la configuration :

```
[AnyBlok]
db_schema.Model.LMC.*=Common
db_schema.Model.LMC.Product=Lense
```



Interface Graphique

FuretUI

Templates des produits (499)

[+ Créer](#)

Marque	Code	Nom	Vue	Fantaisie	Couleur	Torique	Sphérique	Multifocale	Périodicité	Poids (g)	Enveloppe	Statut	Rupture
Cartier S. Cartier	1-888-888888-00	1-888-888888-00	Oui	Non	Non	Non	Oui	Non	DAY	83	Non	Inactif	Non
Cartier S. Cartier	1-888-888888-01	1-888-888888-01	Oui	Non	Non	Non	Oui	Non	DAY	83	Non	Actif	Non
Cartier S. Cartier	1-888-888888-02	1-888-888888-02	Oui	Non	Non	Non	Oui	Non	DAY	278	Non	Inactif	Non
Cartier S. Cartier	1-888-888888-03	1-888-888888-03	Oui	Non	Oui	Non	Non	Non	DAY	33	Oui	Actif	Non

FuretUI

A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

AnyBlok

Documentation

- <https://anyblok.gitbooks.io/anyblok-book/content/en/>
- <https://doc.anyblok.org>
- <https://github.com/AnyBlok>
- <https://gitter.im/AnyBlok/community>

MERCI !

Avez-vous des questions ?