Кастомизация генераторов в Symfony 1.3/1.4.

Управление функционалом и структурой плагинов.

Николай Зык (relo_san)
relo.san@gmail.com

Краткое содержание:

- зачем нужна генерация?
- базовые возможности Symfony
- как сделать плагин универсальнее
- генерация моделей, форм, модулей
- удобное конфигурирование

Зачем нужна генерация?

Зачем нужна генерация?

- увеличение скорости разработки
- увеличение гибкости проекта
- «кеширование» кода

Основные направления генерации в Symfony

- генерация моделей и форм
- генерация модулей
- генерация пустых структур

Шаблоны генераторов можно найти тут:
lib/plugins/sfDoctrinePlugin/data/generator/
lib/plugins/sfPropelPlugin/data/generator/

Что они позволяют делать:

- менять методы в классах форм/фильтров
- менять шаблоны отображения в модулях
- менять методы контроллеров в модулях
- менять конфигурацию и хелперы модулей

Сами генераторы лежат здесь:
lib/plugins/sfDoctrinePlugin/lib/generator/
lib/plugins/sfPropelPlugin/lib/generator/

Что они позволяют делать:

- изменить логику генерации форм
- изменить структуру наследования
- изменить элементы (например виджеты)

Как сделать плагин универсальнее?

- гибкое управление моделями (вкл/выкл)
- произвольные внешние связи
- изменение моделей, по возможности без изменения кода выборок
- управление плагином «из одной точки»

Управление генерацией моделей

- включить только необходимое в проекте
- отключить лишние поля и бихевиоры
- добавить новые поля, бихевиоры, связи

Практически это должно выглядеть примерно так:

lib/config/YourPlugin/model.yml:

```
enabled_models:
  - 'Model_1'
  - 'Model_2'
models:
  Model_1:
    disabled_relations: ['RelationName']
    disabled_columns:  ['anonce',***]
    relation:
      NewRelation:
        ***
```

При этом мы можем задать общие для всех моделей настройки примерно так:

lib/config/model.yml:

```
disabled_behaviors:
  - 'I18n'
options:
  collate: 'utf8_unicode_ci'
  charset: 'utf8'
```

И нам не придется указывать это в конфигах моделей плагинов.

Чтобы плагин был действительно гибким, выборки должны автоматически адаптироваться под изменение схемы данных.

Для этого добавляется серия методов в базовый класс таблицы, позволяющая управлять:

- набором полей в выборках
- допустимыми джойнами

Управление генерацией форм и фильтров

- управление формами (вкл/выкл)
- смена виджетов/валидаторов и их опций
- настройки форм (авто embed I18n, формат имени, лейблов и сообщений и т.д.)

Практически это должно выглядеть примерно так:

lib/config/YourPlugin/model.yml:

```yaml
disabled_forms:
  - 'Model_1'
disabled_filters:
  - 'Model_1'
forms:
  Model_2:
    field_name:
      widget: 'yourCustomWidget'
    other_field_name: false
```

Настройки по умолчанию для плагина вы можете задать в конфиге плагина:

yourPlugin/config/model.yml:

```yaml
enabled_forms:
  - 'Model_1'
forms:
  Model_2:
    field_name: false
```

Это позволит включить в плагине по-умолчанию самое необходимое, чтобы не указывать одно и тоже в каждом проекте.

Управление генерацией модулей

- зависимость от изменения в модели
- выбор типа UI и темы
- управление вложенностью модулей
- управление доступом
- поддержка UI в виде клиентских приложений на Sencha (ExtJS), Adobe AIR

Как и в случае с моделями, мы конфигурируем модуль примерно так:

lib/config/YourPlugin/module.yml:

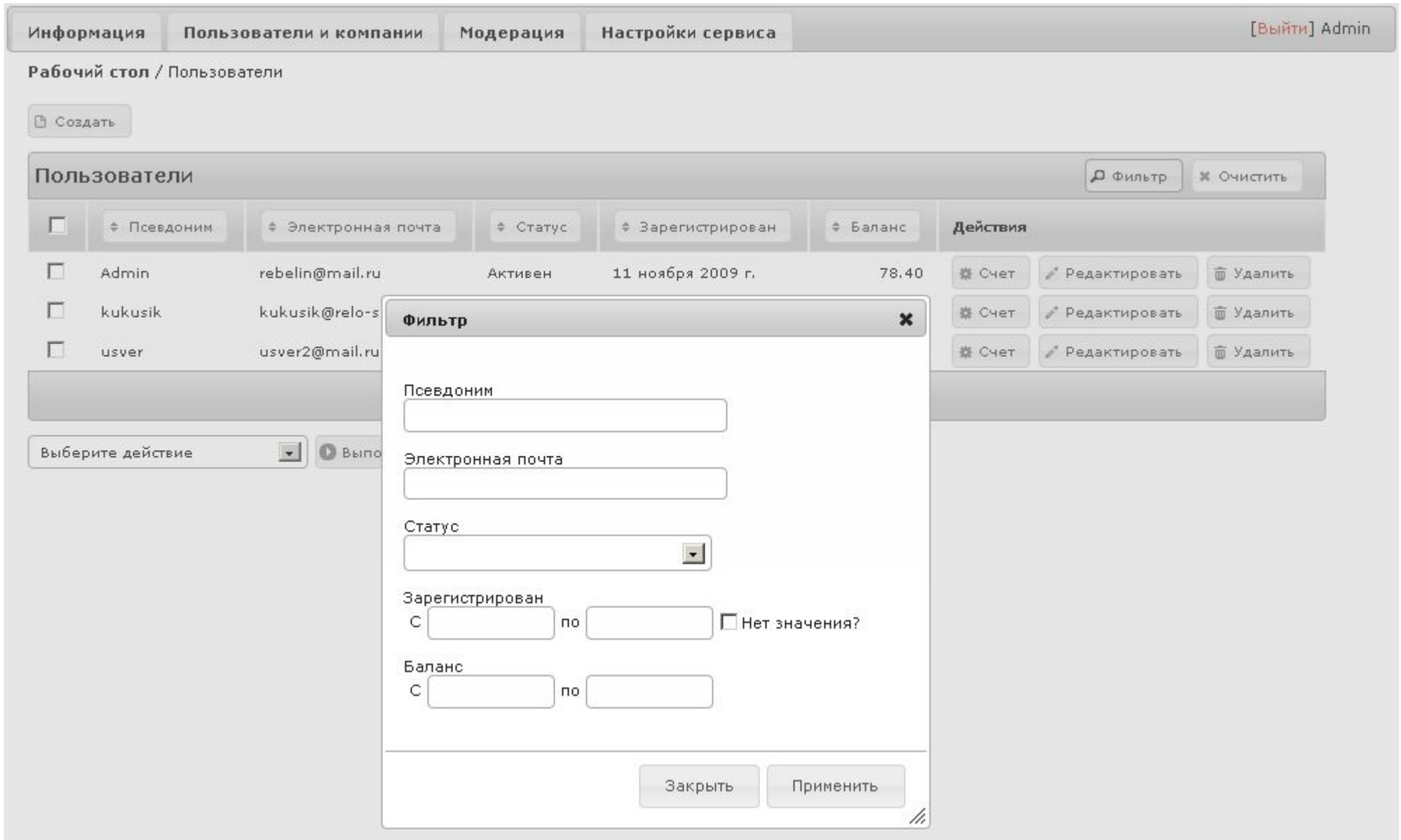
```
all: #all apps or appname
  Module1:
    child:
      module:  'Module2'
      key:     'category_id'
      label:   'label1'
    filter: false
    i18n_catalogue: 'catalogue1'
```

И для всех модулей указываем общие параметры:

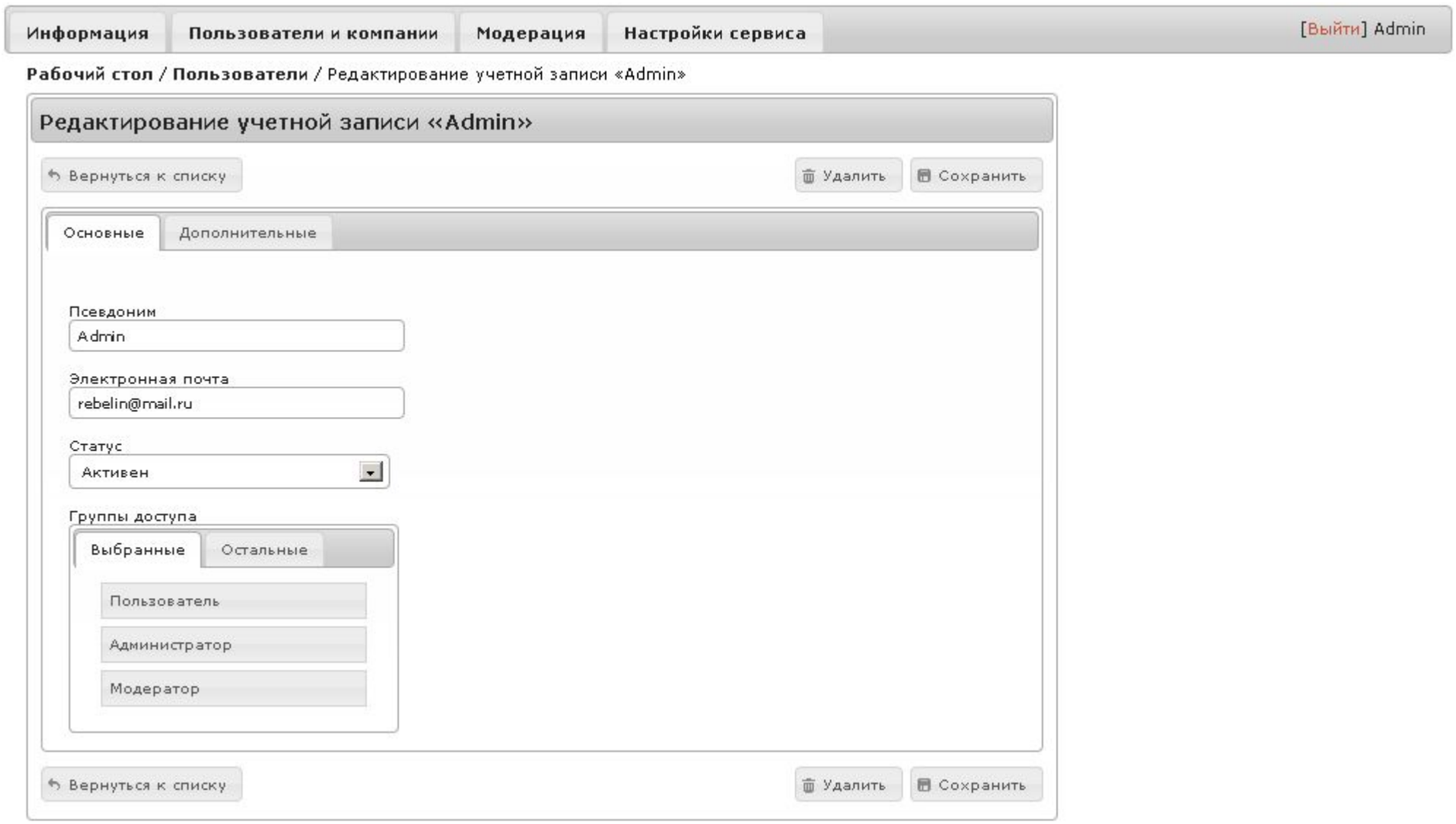
lib/config/module.yml:

```
all: #all apps or appname
  UI:    'jquery'
  theme: 'default'
  actions_base_class:  'dinActions'
  ipp: 20      #items per page
```

Текущая версия плагина поддерживает jQuery UI, сортировку i18n полей в списках и другие мелкие доработки.



В самом ближайшем будущем планируется имплементация существующих расширений модулей (например древовидные списки).



Как не заблудиться в конфигурационном лесу?

symfony UA '10

DineCat

Структура конфигурации генераторов:

/lib/config/model.yml
/lib/config/module.yml

plugins/yourPlugin/config/model.yml
plugins/yourPlugin/config/module.yml

/lib/config/yourPlugin/model.yml
/lib/config/yourPlugin/module.yml

Конфиги указаны в порядке перекрытия.

Возможности, поддерживаемые конфигурацией:

- лаконичный набор команд
- поддержка обоих принципов: «все включено» и «все выключено»
- генерация пустой структуры конфигов
- ограниченная валидация

Опубликованные плагины:

Общие надстройки над фреймворком:
svn://dinecat.com/dinSymfonyExtraPlugin

Надстройки над Doctrine:
svn://dinecat.com/dinDoctrineExtraPlugin

Генерация административной панели:
svn://dinecat.com/dinAdminStdPlugin

Постепенно мной будут публиковаться также другие плагины, в том числе «универсальные» плагины с общим для большинства проектов функционалом.

Плагины публикуются под лицензией MIT.

Я приглашаю всех, заинтересованных в разработке этих и других плагинов, к совместной работе в этом направлении.

В ближайшее время в сети появится два ресурса, использующих эти плагины:

http://maitavr.org/

Ассоциация разработчиков

http://aristotel.ua/

Благотворительный аукцион

Доклад подготовил:

Николай Зык (relo_san)
relo.san@gmail.com

http://dinecat.com/
http://relo-san.com/

symfony UA '10

DineCat