Package Index > **django-robokassa** > **1.1**

django-robokassa 1.1

Приложение для интеграции платежной системы ROBOKASSA в проекты на Django.

Download
django-robokassa-1.1.tar.gz

django-robokassa - это приложение для интеграции платежной системы ROBOKASSA в проекты на Django.

До использования следует ознакомиться с официальной документацией ROBOKASSA (**http://robokassa.ru/Doc/Ru/Interface.aspx**). Приложение реализует протокол взаимодействия, описанный в этом документе.

Установка

```
$ pip install django-robokassa
```

Потом следует добавить 'robokassa' в INSTALLED_APPS и выполнить

```
$ python manage.py syncdb
```

или, если используется South,

```
$ python manage.py migrate
```

Для работы требуется django >= 1.3.1. Используйте django-robokassa версии 0.9.3, если проект на django 1.2.x или django 1.1.x.

Настройка

В settings.py нужно указать следующие настройки:

- ROBOKASSA_LOGIN - логин
- ROBOKASSA_PASSWORD1 - пароль №1

Необязательные параметры:

- ROBOKASSA_PASSWORD2 - пароль №2. Его можно не указывать, если django-robokassa используется только для вывода формы платежа. Если django-robokassa используется для приема платежей, то этот параметр обязательный.
- ROBOKASSA_USE_POST - используется ли метод POST при приеме результатов от ROBOKASSA. По умолчанию - True. Считается, что для Result URL, Success URL и Fail URL выбран один и тот же метод.
- ROBOKASSA_STRICT_CHECK - использовать ли строгую проверку (требовать предварительного уведомления на ResultURL). По умолчанию - True.
- ROBOKASSA_TEST_MODE - включен ли тестовый режим. По умолчанию False (т.е. включен боевой режим).
- ROBOKASSA_EXTRA_PARAMS - список (list) названий дополнительных параметров, которые будут передаваться вместе с запросами. "Shp" к ним приписывать не нужно.

Использование

Форма для приема платежей

Для того, чтобы упростить конструирование html-форм для отправки пользователей в Robokassa, в django-robokassa есть форма RobokassaForm. Она нужна для упрощения вывода информации в шаблонах, вычисления контрольной суммы и формирования параметров GET-запросов.

Пример:

```
# views.py

from django.shortcuts import get_object_or_404, render
from django.contrib.auth.decorators import login_required

from robokassa.forms import RobokassaForm

@login_required
def pay_with_robokassa(request, order_id):
    order = get_object_or_404(Order, pk=order_id)

    form = RobokassaForm(initial={
               'OutSum': order.total,
               'InvId': order.id,
               'Desc': order.name,
               'Email': request.user.email,
               # 'IncCurrLabel': '',
               # 'Culture': 'ru'
           })

    return render(request, 'pay_with_robokassa.html', {'form': form})
```

В initial все параметры необязательны. Детальную справку по параметрам лучше посмотреть в **документации** к Robokassa. Можно передавать в initial значения "пользовательских параметров", описанных в ROBOKASSA_EXTRA_PARAMS ('shp' к ним приписывать опять не нужно).

Соответствующий шаблон:

```
{% extends 'base.html' %}

{% block content %}
    <form action="{{ form.target }}" method="POST">
        <p>{{ form.as_p }}</p>
        <p><input type="submit" value="Купить"></p>
    </form>
{% endblock %}
```

Форма выведется в виде набора скрытых input-тегов.

У формы есть атрибут target, содержащий URL, по которому форму следует отправлять. В тестовом режиме это будет тестовый URL, в боевом - боевой.

Обратите внимание, {% csrf_token %} в форме не нужен (и более того, добавлять его к форме небезопасно), т.к. форма ведет на внешний сайт - сайт робокассы.

Вместо отправки формы можно сформировать GET-запрос. У формы есть метод

get_redirect_url, который возвращает нужный адрес со всеми параметрами. Редирект на этот адрес равносилен отправке формы методом GET.

django-robokassa не включает в себя модели "Покупка" (`Order` в примере), т.к. эта модель будет отличаться от сайта к сайту. Обработку смены статусов покупок следует осуществлять в обработчиках сигналов.

Получение результатов платежей

В Robokassa есть несколько методов определения результата платежа:

1. При переходе на страницы Success и Fail гарантируется, что платеж соответственно прошел и не прошел
2. При успешном или неудачном платеже Robokassa отправляет POST или GET запрос на Result URL.
3. Можно запрашивать статус платежа через XML-сервис.

В django-robokassa на данный момент поддерживаются методы 1 и 2 и их совмещение (дополнительная проверка, что при переходе на Success URL уже было уведомление на Result URL при использовании опции ROBOKASSA_STRICT_CHECK = True).

В целях безопасности лучше всегда использовать строгую проверку (с подтверждением через Result URL). Ее механизм:

1. После оплаты robokassa.ru отправляет "фоновый" запрос на ResultURL.
2. Внутри view, связанного с ResultURL, происходит проверка содержащейся в запросе md5-подписи через ROBOKASSA_PASSWORD2 (это второй пароль, который не передается по сети и известен только отправителю и получателю). ROBOKASSA_PASSWORD2 нужен для подтверждения того, что запрос был послан именно с robokassa.ru.
3. Если запрос правильный, то view шлет сигнал `robokassa.signals.result_received`. Чтоб производить манипуляции внутри сайта (например, начислять средства согласно пришедшему запросу или менять статус заказа), нужно добавить соответствующий обработчик этого сигнала.
4. Если все в порядке, то view, связанный с Result URL, отдает robokassa.ru ответ вида `OK<operation_id>`, где `<operation_id>` - уникальный id текущей операции. Этот ответ необходим для того, чтобы robokassa.ru получила подтверждение того, что все необходимые действия произведены.
5. Если robokassa.ru получает этот ответ, то пользователь перенаправляется на Success URL. На этой страничке обычно лучше вывести сообщение об успешном прохождении платежа/оплаты. Если ответ view, связанной с Result URL, не соответвтует ожидаемому, то пользователь перенаправляется не на Success URL, а на Fail URL; там ему хорошо бы показать сообщение о произошедшей ошибке.

Сигналы

Обработку смены статусов покупок следует осуществлять в обработчиках сигналов.

- `robokassa.signals.result_received` - шлется при получении уведомления от Robokassa. Получение этого сигнала означает, что оплата была успешной. В качестве sender передается экземпляр модели SuccessNotification, у которой есть атрибуты InvId и OutSum.

- `robokassa.signals.success_page_visited` - шлется при переходе пользователя на страницу успешной оплаты. Этот сигнал следует использовать вместо result_received, если не используется строгая проверка (ROBOKASSA_STRICT_CHECK=False)
- `robokassa.signals.fail_page_visited` - шлется при переходе пользователя на страницу ошибки оплаты. Получение этого сигнала означает, что оплата не была произведена. В обработчике следует осуществлять разблокирвку товара на складе и т.д.

Все сигналы получают параметры InvId (номер заказа), OutSum (сумма оплаты) и extra (словарь с дополнительными параметрами, описанными в ROBOKASSA_EXTRA_PARAMS).

Пример:

```
from robokassa.signals import result_received
from my_app.models import Order

def payment_received(sender, **kwargs):
    order = Order.objects.get(id=kwargs['InvId'])
    order.status = 'paid'
    order.paid_sum = kwargs['OutSum']
    order.extra_param = kwargs['extra']['my_param']
    order.save()

result_received.connect(payment_received)
```

urls.py

Для настройки Result URL, Success URL и Fail URL можно подключить модуль robokassa.urls:

```
urlpatterns = patterns('',
    #...
    url(r'^robokassa/', include('robokassa.urls')),
    #...
)
```

Адреса, которые нужно указывать в панели robokassa, в этом случае будут иметь вид

- Result URL: `http://yoursite.ru/robokassa/result/`
- Success URL: `http://yoursite.ru/robokassa/success/`
- Fail URL: `http://yoursite.ru/robokassa/fail/`

Шаблоны

- `robokassa/success.html` - показывается в случае успешной оплаты. В контексте есть переменная form типа `SuccessRedirectForm`, InvId и OutSum с параметрами заказа, а также все дополнительные параметры, описанные в ROBOKASSA_EXTRA_PARAMS.
- `robokassa/fail.html` - показывается в случае неуспешной оплаты. В контексте есть переменная form типа `FailRedirectForm`, InvId и OutSum с параметрами заказа, а также все дополнительные параметры, описанные в ROBOKASSA_EXTRA_PARAMS.
- `robokassa/error.html` - показывается при ошибочном запросе к странице "успех" или "неудача" (например, при ошибке в контрольной сумме). В контексте есть переменная form класса `FailRedirectForm` или `SuccessRedirectForm`.

Разработка

Разработка ведется на bitbucket и github:

- **https://bitbucket.org/kmike/django-robokassa/**
- **https://github.com/kmike/django-robokassa**

Пожелания, идеи, баг-репорты и тд. пишите в трекер: **https://bitbucket.org/kmike/django-robokassa/issues**

Лицензия - MIT.

Тестирование

Для запуска тестов установите **tox**, склонируйте репозиторий и выполните команду

```
$ tox
```

из корня репозитория.

История изменений

1.1 (2013-04-12)

- На странице FailURL больше не проверяется подпись (т.к. Робокасса ее больше не передает) - спасибо @amureki;
- улучшена справка - спасибо @bo858585.

1.0 (2012-03-24)

- Для работы теперь требуется django >= 1.3;
- добавлена поддержка django 1.4;
- все вьюхи возвращают теперь TemplateResponse;
- миграции переведены на современную версию south;
- запуск тестов через tox;
- небольшие улучшения в README.

0.9.3 (2010-08-05)

Сообщения с ошибочной подписью не вызывают исключения.

0.9.2 (2010-06-23)

Добавлена поддержка django 1.2.

0.9.1 (2010-05-10)

Исправлена работа с дополнительными (пользовательскими) параметрами.

0.9.0 (2010-04-15)

Первая версия

File	Type	Py Version	Uploaded on	Size
django-robokassa-1.1.tar.gz (**md5**)	Source		2013-04-12	11KB

Downloads (All Versions):
14 downloads in the last day
138 downloads in the last week
519 downloads in the last month

Author: Mikhail Korobov
Home Page: https://bitbucket.org/kmike/django-robokassa/
License: MIT license
Requires django (>= 1.3)
Categories
- **Development Status :: 4 - Beta**
- **Environment :: Web Environment**
- **Framework :: Django**
- **Intended Audience :: Developers**
- **License :: OSI Approved :: MIT License**
- **Natural Language :: Russian**
- **Programming Language :: Python**
- **Programming Language :: Python :: 2**
- **Programming Language :: Python :: 2.5**
- **Programming Language :: Python :: 2.6**
- **Programming Language :: Python :: 2.7**
- **Topic :: Software Development :: Libraries :: Python Modules**

Package Index Owner: kmike
DOAP record: django-robokassa-1.1.xml