Kostenloses eBook

LERNEN

tastypie

Free unaffiliated eBook created from
Stack Overflow contributors.

#tastypie

Inhaltsverzeichnis

Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: tastypie

It is an unofficial and free tastypie ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official tastypie.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Kapitel 1: Erste Schritte mit tastypie

Bemerkungen

In diesem Abschnitt erhalten Sie einen Überblick darüber, was tastypie ist und warum ein Entwickler es verwenden möchte.

Es sollte auch alle großen Themen innerhalb von tastypie erwähnen und auf die verwandten Themen verweisen. Da die Dokumentation für tastypie neu ist, müssen Sie möglicherweise erste Versionen dieser verwandten Themen erstellen.

Examples

Installation oder Setup

Detaillierte Anweisungen zum Einrichten oder Installieren von tastypie.

Erste Schritte mit tastypie online lesen: https://riptutorial.com/de/tastypie/topic/9618/erste-schritte-mit-tastypie

Kapitel 2: Codierung mit Tastypie

Einführung

Tastypie ist ein Webservice-API-Framework für Django. Es bietet eine komfortable, jedoch leistungsstarke und anpassbare Abstraktion für die Erstellung von Schnittstellen im REST-Stil. Tastypie macht das Freilegen Ihrer Modelle einfach, gibt Ihnen jedoch die vollständige Kontrolle über das, was Sie freigeben, und lässt die Datenbank so weit wie nötig abstrahlen. Mit Tastypie lässt sich die Integration in Datenquellen, die nicht aus ORM-Daten stammen, leicht machen. Daher kann Tastypie mit ORM- und Nicht-ORM-Datenbanken verwendet werden.

Examples

Schnellstart

1. Fügen Sie tastypie zu INSTALLED_APPS hinzu.
2. Erstellen Sie ein API-Verzeichnis in Ihrer App mit einem bloßen **init** .py.
3. Erstellen Sie eine <my_app> /api/resources.py -Datei und fügen Sie Folgendes ein:

```
from tastypie.resources import ModelResource
from my_app.models import MyModel


class MyModelResource(ModelResource):
    class Meta:
        queryset = MyModel.objects.all()
        allowed_methods = ['get']
```

Fügen Sie in Ihrer Stamm-URLconf den folgenden Code hinzu (um den Admin-Code herum):

```
from django.conf.urls import url, include
from tastypie.api import Api
from my_app.api.resources import MyModelResource

v1_api = Api(api_name='v1')
v1_api.register(MyModelResource())

urlpatterns = [
  # ...more URLconf bits here...
  # Then add:
  url(r'^api/', include(v1_api.urls)),
]
```

Tastypie Installation

Tastypie kann mit Python Package Management installiert werden, dh `pip` oder wir können den Code direkt von Github auschecken

1. pip install django-tastypie

2. Kasse bei Github

Warum Tastypie?

Es gibt andere API-Frameworks für Django. Sie müssen die verfügbaren Optionen bewerten und selbst entscheiden. Dies sind jedoch einige häufige Gründe für den Geschmack.

- Sie benötigen eine API, die RESTful ist und HTTP gut verwendet.
- Sie möchten tiefe Beziehungen unterstützen.
- Sie müssen NICHT Ihren eigenen Serializer schreiben, um die Ausgabe richtig zu machen.
- Sie möchten ein API-Framework, das wenig Magie besitzt, sehr flexibel ist und der Problemdomäne gut zugeordnet ist.
- Sie wollen / benötigen eine XML-Serialisierung, die genauso wie JSON behandelt wird (und YAML ist auch da).

Um API mit Django zu entwickeln, haben wir viele Möglichkeiten. Die wichtigsten Optionen sind **Tastypie** und **Django Rest Framework** (DRF).

Was macht ein anständiges API-Framework aus?

Die Merkmale:

1. Seitennummerierung
2. Buchung von Daten mit Validierung
3. Veröffentlichen von Metadaten zusammen mit Querysets
4. API-Erkennung
5. richtige HTTP-Antwortbehandlung
6. Caching
7. Serialisierung
8. Drosselung
9. Berechtigungen
10. Authentifizierung

Geeignete API-Frameworks benötigen außerdem:

11. Wirklich gute Testabdeckung für ihren Code
12. Anständige Leistung
13. Dokumentation
14. Eine aktive Gemeinschaft, um den Rahmen voranzubringen und zu unterstützen

Wenn Sie diese Faktoren berücksichtigen, gibt es derzeit nur zwei API-Frameworks, die es wert sind (django-tastypie und django-rest-Framework).

Welches ist besser? Django-Tastypie oder Django-Rest-Framework?

Ich sage, dass sie gleich sind. Sie können mit keinem der beiden einfach nichts falsch machen. Die Autoren und Communities hinter beiden sind aktiv, der Code ist solide und getestet. Und hier

sind meine spezifischen Gedanken zu beiden:

Tastypie:

Vorteile:

- Einfache Inbetriebnahme und Bereitstellung grundlegender Funktionen OOB (out of the box)
- Meistens werden Sie sich nicht mit fortgeschrittenen Django-Konzepten wie CBVs, Forms usw. Beschäftigen
- Lesbarer Code und weniger Magie!
- Wenn Ihre Modelle NON-ORM sind, wählen Sie es aus.

Nachteile:

- Folgt nicht streng dem idiomatischen Django (Geistesphysik und Djangos Philosophie sind ziemlich unterschiedlich)
- Wahrscheinlich etwas schwierig, APIs anzupassen, sobald Sie groß sind
- Kein O-Auth

DRF:

Vorteile:

- Folgen Sie dem idiomatic Django. (Wenn Sie Django von innen und außen kennen, und mit CBV, Forms usw. sehr vertraut sind, gehen Sie ohne Zweifel davon aus)
- Stellt die REST-Funktionalität mit ModelViewSets bereit. Gleichzeitig bietet es eine bessere Kontrolle für die Anpassung mit CustomSerializer, APIView, GenericViews usw.
- Bessere Authentifizierung
- Einfachere Erstellung benutzerdefinierter Berechtigungsklassen.
- Arbeiten Sie sehr gut und vor allem sehr einfach, damit es mit Bibliotheken von Drittanbietern und OAuth funktioniert.
- DJANGO-REST-AUTH ist eine Erwähnung der Bibliothek für Auth / SocialAuthentication / Registration.

Nachteile:

- Wenn Sie Django nicht gut kennen, sollten Sie dies nicht tun.
- Zauber! Einige Zeit sehr schwer zu verstehen, Magie.
- Weil es auf djangos CBV geschrieben wurde, die wiederum recht komplex sind.
- Hat eine steile Lernkurve.

Erste Schritte mit Tastypie

Tastypie ist eine wiederverwendbare App (dh sie basiert nur auf ihrem eigenen Code und konzentriert sich darauf, nur eine API im REST-Stil bereitzustellen) und eignet sich zur

Bereitstellung einer API für jede Anwendung, ohne die Quellen dieser App ändern zu müssen.

Da nicht alle Bedürfnisse gleich sind, bietet Tastypie viel Wert darauf, viele Haken zum Überschreiben oder Erweitern der Funktionsweise bereitzustellen.

Zu Beispielzwecken fügen wir einer einfachen Bloganwendung eine API hinzu.

Die einzige obligatorische Konfiguration ist das Hinzufügen von ' ***tastypie*** *' zu Ihrem* ***INSTALLED_APPS*** *. Dies ist nicht unbedingt erforderlich, da Tastypie nur über zwei nicht benötigte Modelle verfügt, dies kann jedoch die Verwendung erleichtern.*

Hier ist myapp / models.py:

```
from tastypie.utils.timezone import now
from django.contrib.auth.models import User
from django.db import models
from django.utils.text import slugify


class Entry(models.Model):
    user = models.ForeignKey(User)
    pub_date = models.DateTimeField(default=now)
    title = models.CharField(max_length=200)
    slug = models.SlugField(null=True, blank=True)
    body = models.TextField()
```

Ressourcen erstellen

Bei der Architektur im REST-Stil geht es um *Ressourcen* . Daher ist es nicht überraschend, dass bei der Integration in Tastypie *Ressourcenklassen erstellt werden* . Für unsere einfache Anwendung erstellen wir dafür eine Datei in **myapp / api.py** , die jedoch überall in Ihrer Anwendung leben kann:

myapp / api.py

```
from tastypie.resources import ModelResource
from myapp.models import Entry


class EntryResource(ModelResource):
    class Meta:
        queryset = Entry.objects.all()
        resource_name = 'entry'
```

In REST ist alles eine Ressource, dh Objekte werden als Ressourcen bezeichnet. Zum Erstellen einer Tastypie-REST-API muss also eine Ressourcenklasse erstellt werden. Um eine Ressource zu erstellen, müssen Sie die Klasse ModelResource unterteilen. Diese EntryResource-Klasse überprüft alle nicht relationalen Felder im Entry-Modell und erstellt eigene Api-Felder. *Dies funktioniert genauso wie das ModelForm von Django Forms.*

Anschließen der Ressource: Wenn Sie die Ressource erstellen, müssen Sie dem Django

mitteilen, dass eine Ressource erstellt wird, indem Sie unsere EntryResource mit der URL verknüpfen.

urls.py

```
from django.conf.urls import url, include
from myapp.api import EntryResource

entry_resource = EntryResource()

urlpatterns = [
    url(r'^blog/', include('myapp.urls')),
    url(r'^api/', include(entry_resource.urls)),
]
```

Wenn wir nun **localhost** überprüfen **: 8000 / api / entry /, erhalten** wir die API-Antwort.

*(Der **Ressourcenname** innerhalb der Meta-Klasse ist optional. Wenn er nicht angegeben wird, wird er automatisch aus dem Klassennamen generiert, entfernt alle Instanzen von Resource und setzt den String in eine **niedrigere Position** . Daher wird **EntryResource nur ein Eintrag** . Deshalb werden in der URL **../eintrag** angegeben **/**)*

Tastypie-Autorisierung

Bis jetzt haben wir versucht, eine Anfrage zu bekommen. *(Wenn Sie andere HTTP-Anfragen durchführen müssen, verwenden Sie einige API-Test-Tools wie curl oder postman.)*

Wenn Sie versuchen, ein POST / PUT / DELETE an die Ressource zu senden, werden "401 Unauthorized" -Fehler angezeigt. Aus Sicherheitsgründen wird bei Tastypie die Berechtigungsklasse ReadOnlyAuthorization festgelegt. Dies macht es sicher, im Web zu zeigen, verhindert jedoch POST / PUT / DELETE. Aktivieren wir diese.

myapp / api.py

```
from tastypie.authorization import Authorization
from tastypie.resources import ModelResource
from myapp.models import Entry


class EntryResource(ModelResource):
    class Meta:
        queryset = Entry.objects.all()
        resource_name = 'entry'
        authorization = Authorization()
```

Codierung mit Tastypie online lesen: https://riptutorial.com/de/tastypie/topic/10640/codierung-mit-tastypie

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit tastypie	Community
2	Codierung mit Tastypie	AKHIL MATHEW