django-google-address Documentation

Release 0.1.0

Leonardo Arroyo

May 03, 2017

Contents:

django-google-address provides you a quick way to integrate addresses in your models backed by Google Maps API.

CHAPTER 1

Getting Started

The module is easy to use and you can get started with django-google-address in just a few minutes.

Installing

1. Install the module like any other python module:

```
pip install django-google-address
```

2. Add it to *INSTALLED_APPS* on *settings.py*:

```
INSTALLED_APPS = [
  ...,
  'google_address'
]
```

3. Migrate:

```
./manage.py migrate
```

Configuring

There are two settings you can and should use to make the module work correctly. Here is an example of such configuration:

```
GOOGLE_ADDRESS = {
  'API_KEY': 'BIzaSyFCcHqskKlpsUJ5jdQGVzdP-7uNu9O4g8w'
  'API_LANGUAGE': 'en_US'
}
```

You can read more about configuring at *Settings* section.

Using

Create an Address object with the raw address. Requests will be made to the Google API when saving the address:

```
>>> from google_address.models import Address
>>> a = Address(raw="Av. Paulista, 1000")
>>> a.save()
>>> a.address_line
'Avenida Paulista, 1000, Bela Vista, São Paulo, SP, Brazil'
```

You can easily use foreign keys to the model:

```
class Building(models.Model):
  address = models.ForeignKey('google_address.Address')
```

CHAPTER 2

Settings

All settings must be put inside a dictionary named GOOGLE_ADDRESS in your settings.py

API_KEY

This is the Google Maps API key acquired through Google console.

Default: *None*

API_LANGUAGE

The language which addresses will be requested in.

Default: *en_US*

ASYNC_CALLS

When saving an *Address* object, a request will be made to Google Maps API to fetch address information. When set to *True*, this will make the requests asynchronous so you can return a response to the user without waiting for an API response.

Default: *False*

Example configuration

```
GOOGLE_ADDRESS = {
  'API_KEY': 'BIzaSyFCcHqskKlpsUJ5jdQGVzdP-7uNu9O4g8w',
  'API_LANGUAGE': 'en_US'
}
```

CHAPTER 3

Models

django-google-address uses 3 models to store addresses.

You should only create objects on the Address model. The other two models are only used internally to hold data about addresses and to allow advanced filtering.

Address

The Address model is the reason for this module to exist.

Creating objects

While creating Address objects, you can set raw and raw2 fields.

raw field is the address that may include street name, number, neighborhood, city, state and country. *raw2* is any extra data you might want to include with the address, such as an apartment number.

When the object *save* method is called, a request is made to Google Maps API to fetch information about the address.

Existing objects

After an addres is saved, you can access a few extra fields such as *address_line*, *city_state*, *lat* and *lng*

Example

```
>>> from google_address.models import Address
>>> a = Address(raw="Av. Paulista, 1000", raw2="Room number 101")
>>> a.save()
>>> a.address_line
'Avenida Paulista, 1000, Bela Vista, São Paulo, SP, Brazil'
```

```
>>> a.city_state
'São Paulo, SP'
>>> (a.lat, a.lng)
(-23.5647577, -46.6518495)
```

Fields

```
class Address(models.Model):
  raw = models.CharField(max_length=400, blank=True, null=True)
  raw2 = models.CharField(max_length=400, blank=True, null=True)
  address_line = models.CharField(max_length=400, blank=True, null=True)
  city_state = models.CharField(max_length=400, blank=True, null=True)
  lat = models.FloatField('lat', blank=True, null=True)
  lng = models.FloatField('lng', blank=True, null=True)
  address_components = models.ManyToManyField(AddressComponent)
```

AddressComponent

This model should not be created manually. It is automatically created while saving an Address model. It can be used for advanced filtering. Read more at *Filtering*.

Fields

```
class AddressComponent(models.Model):
  long_name = models.CharField(max_length=400)
  short_name = models.CharField(max_length=400)
  types = models.ManyToManyField(AddressComponentType)
```

AddressComponentType

This model should not be created manually. It is automatically created while saving an Address model. It can be used for advanced filtering. Read more at *Filtering*.

Fields

```
class AddressComponentType(models.Model):
  name = models.CharField(max_length=100)
```

CHAPTER 4

Filtering

Due to addresses structure in the database, you can easily filter them by any address component type. We recommend you get familiarized with the *model structure* and Google Maps API address components.

Examples

Filtering by country

```
>> Address.objects.filter(address_components__long_name="Brazil", address_components__
↪types__name="country")
```

Filtering by state

```
>> Address.objects.filter(address_components__long_name="São Paulo", address_
↪components__types__name="administrative_area_level_1")
```

Filtering by city

```
>> Address.objects.filter(address_components__long_name="São Paulo", address_
↪components__types__name="locality")
```

Filtering by neighborhood

```
>> Address.objects.filter(address_components__long_name="Bela Vista", address_
↪components__types__name__in=["neighborhood", "sublocality_level_1"])
```