

Stack Overflow is a community of 4.7 million programmers, just like you, helping each other.

Join them; it only takes a minute:

[Sign up](#)

Join the Stack Overflow community to:



Ask programming questions



Answer and help your peers



Get recognized for your expertise

Need a minimal Django file upload example [closed]

asked 4 years ago

viewed 160545 times

active 4 months ago

As a newbie to Django, I am having difficulty making an upload app in Django 1.3. I could not find any up-to-date example/snippets. May someone post a minimal but complete (Model, View, Template) example code to do so?

[django](#) [file](#) [upload](#)

share improve this question

edited Jun 26 '15 at 20:04



[pnovotnak](#)

350 ● 2 ● 14

asked May 3 '11 at 15:17



[qliq](#)

3,289 ● 8 ● 28 ● 44

closed as too broad by [Sayse](#), [Louis](#), [davidism](#), [Servy](#), [Tiny Giant](#) Jan 12 at 20:09

There are either too many possible answers, or good answers would be too long for this format. Please add details to narrow the answer set or to isolate an issue that can be answered in a few paragraphs.

If this question can be reworded to fit the rules in the [help center](#), please [edit the question](#).

[add a comment](#)

10 Answers

active

oldest

votes

Phew, Django documentation really does not have good example about this. I spent over 2 hours to dig up all the pieces to understand how this works. With that knowledge I implemented a project that makes possible to upload files and show them as list. To download source for the project, visit <https://github.com/axelpale/minimal-django-file-upload-example> or clone it:

```
> git clone https://github.com/axelpale/minimal-django-file-upload-example.git
```

Update 2013-01-30: The source at GitHub has also implementation for Django 1.4 in addition to 1.3. Even though there is few changes the following tutorial is also useful for 1.4.

Update 2013-05-10: Implementation for Django 1.5 at GitHub. Minor changes in redirection in urls.py and usage of url template tag in list.html. Thanks to [hubert3](#) for the effort.

Update 2013-12-07: Django 1.6 supported at GitHub. One import changed in myapp/urls.py. Thanks goes to [Arthedian](#).

Update 2015-03-17: Django 1.7 supported at GitHub, thanks to [aronymsidoro](#).

Update 2015-09-04: Django 1.8 supported at GitHub, thanks to [nerogit](#).

Project tree

A basic Django 1.3 project with single app and media/ directory for uploads.

```
minimal-django-file-upload-example/
src/
myproject/
database/
```

Linked

0 [django file upload using form in html template](#)

41 [Django: how do you serve media / stylesheets and link to them within templates](#)

11 [how to manually assign imagefield in Django](#)

5 [How to upload an image in django web application?](#)

2 [form.is_valid\(\) returns false \(django\)](#)

1 [Handling requests in Django](#)

0 [handling uploading image django admin python](#)

1 [Python get file name and change & save it in variable](#)

3 [Admin FileField current url incorrect](#)

3 [django-summernote image upload](#)

[see more linked questions...](#)

Related

1891 [How can I upload files asynchronously?](#)

2606 [Include a JavaScript file in another JavaScript file?](#)

0 [Django ajax file upload problem csrf](#)

1 [Need a minimal Django data store example](#)

1 [Send file from backbone Model to django](#)

```

    sqlite.db
media/
myapp/
    templates/
        myapp/
            list.html
    forms.py
    models.py
    urls.py
    views.py
    __init__.py
    manage.py
    settings.py
    urls.py

```

1. Settings: myproject/settings.py

To upload and serve files, you need to specify where Django stores uploaded files and from what URL Django serves them. `MEDIA_ROOT` and `MEDIA_URL` are in `settings.py` by default but they are empty. See the first lines in [Django Managing Files](#) for details. Remember also set the database and add `myapp` to `INSTALLED_APPS`

```

...
import os

BASE_DIR = os.path.dirname(os.path.dirname(__file__))
...
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'database.sqlite3'),
        'USER': '',
        'PASSWORD': '',
        'HOST': '',
        'PORT': '',
    }
}
...
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'
...
INSTALLED_APPS = (
    ...
    'myapp',
)

```

2. Model: myproject/myapp/models.py

Next you need a model with a `FileField`. This particular field stores files e.g. to `media/documents/2011/12/24/` based on current date and `MEDIA_ROOT`. See [FileField reference](#).

```

# -*- coding: utf-8 -*-
from django.db import models

class Document(models.Model):
    docfile = models.FileField(upload_to='documents/%Y/%m/%d')

```

3. Form: myproject/myapp/forms.py

To handle upload nicely, you need a form. This form has only one field but that is enough. See [Form FileField reference](#) for details.

```

# -*- coding: utf-8 -*-
from django import forms

class DocumentForm(forms.Form):
    docfile = forms.FileField(
        label='Select a file',
        help_text='max. 42 megabytes'
    )

```

4. View: myproject/myapp/views.py

A view where all the magic happens. Pay attention how `request.FILES` are handled. For me, it was really hard to spot the fact that `request.FILES['docfile']` can be saved to `models.FileField` just like that. The model's `save()` handles the storing of the file to the filesystem automatically.

```

# -*- coding: utf-8 -*-
from django.shortcuts import render_to_response
from django.template import RequestContext
from django.http import HttpResponseRedirect
from django.core.urlresolvers import reverse

from myproject.myapp.models import Document
from myproject.myapp.forms import DocumentForm

```

4 django upload file no model

0

Django: Can't pass parameters to Python class, name 'uid' is not defined

0

How do you get Django's filer app to list new files at the top of the list?

1

Django 1.8 File Upload Example

0

Django file upload form with single button

Hot Network Questions

What Style is Better (Instance Variable vs. Return Value) in Java

You can control a Demon by knowing its True Name, but why?

New UK passport with new name; would the US know it's still me? Suspected outstanding warrant

(How) can a Warforged get the Dragonblood Subtype?

Why would I tar a single file?

Can I have fun with FizzBuzz?

What's the difference between these two types of compression fittings?

Does the Constitution of Bhutan state that 60% of Bhutan must be forested?

Getting ordered coordinates out of ConvexHullRegion

Easy partial fraction decomposition with complex numbers

Driftsort an array

Why is NP in EXPTIME?

Auto Breaking of Long Math Equations

Plain seltzer on Pesach

Bringing coins back home from abroad; Is that unethical to do?

What is a "Magic Packet" for waking a computer?

Challenge/response authentication for garage door opener

How can I keep stray cats off my car?

Concerns about a very old PhD advisor?

Languages beyond enumerable

Can the US military seize a country which has the ability to kill anyone based on the victim's name and face?

Any equivalent to this Persian proverb "The yellow dog is the jackal's brother"?

How do I "stock groceries" in Spanish?

What was on Professor Quirrell's list of mistakes he would never make as a Dark Lord?

```
def list(request):
    # Handle file upload
    if request.method == 'POST':
        form = DocumentForm(request.POST, request.FILES)
        if form.is_valid():
            newdoc = Document(docfile = request.FILES['docfile'])
            newdoc.save()

            # Redirect to the document list after POST
            return HttpResponseRedirect(reverse('myapp.views.list'))
    else:
        form = DocumentForm() # A empty, unbound form

    # Load documents for the list page
    documents = Document.objects.all()

    # Render list page with the documents and the form
    return render_to_response(
        'myapp/list.html',
        {'documents': documents, 'form': form},
        context_instance=RequestContext(request)
    )
```

5. Project URLs: myproject/urls.py

Django does not serve MEDIA_ROOT by default. That would be dangerous in production environment. But in development stage, we could cut short. Pay attention to the last line. That line enables Django to serve files from MEDIA_URL. This works only in development stage.

See [django.conf.urls.static.static](#) reference for details. See also [this discussion about serving media files](#).

```
# -*- coding: utf-8 -*-
from django.conf.urls import patterns, include, url
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = patterns('',
    (r'^$', include('myapp.urls')),
) + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

6. App URLs: myproject/myapp/urls.py

To make the view accessible, you must specify urls for it. Nothing special here.

```
# -*- coding: utf-8 -*-
from django.conf.urls import patterns, url

urlpatterns = patterns('myapp.views',
    url(r'^list/$', 'list', name='list'),
)
```

7. Template: myproject/myapp/templates/myapp/list.html

The last part: template for the list and the upload form below it. The form must have enctype-attribute set to "multipart/form-data" and method set to "post" to make upload to Django possible. See [File Uploads documentation](#) for details.

The FileField has many attributes that can be used in templates. E.g. {{ document.docfile.url }} and {{ document.docfile.name }} as in the template. See more about these in [Using files in models article](#) and [The File object documentation](#).

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Minimal Django File Upload Example</title>
  </head>
  <body>
    <!-- List of uploaded documents -->
    {% if documents %}
      <ul>
        {% for document in documents %}
          <li><a href="{ document.docfile.url }">{{ document.docfile.name }}</a></li>
        {% endfor %}
      </ul>
    {% else %}
      <p>No documents.</p>
    {% endif %}

    <!-- Upload form. Note enctype attribute! -->
    <form action="{% url 'list' %}" method="post" enctype="multipart/form-data">
      {% csrf_token %}
      <p>{{ form.non_field_errors }}</p>
      <p>{{ form.docfile.label_tag }} {{ form.docfile.help_text }}</p>
      <p>
        {{ form.docfile.errors }}
        {{ form.docfile }}
      </p>
      <p><input type="submit" value="Upload" /></p>
    </form>
```

```
</body>
</html>
```

8. Initialize

Just run syncdb and runserver.

```
> cd myproject
> python manage.py syncdb
> python manage.py runserver
```

Results

Finally, everything is ready. On default Django development environment the list of uploaded documents can be seen at `localhost:8000/list/`. Today the files are uploaded to `/path/to/myproject/media/documents/2011/12/17/` and can be opened from the list.

I hope this answer will help someone as much as it would have helped me.

share improve this answer

edited Sep 4 '15 at 14:56

answered Dec 17 '11 at 1:59



Akseli Palén

13.1k ● 1 ● 27 ● 44

-
- 67 Thanks for this post. This has been very helpful! – [Samad Lotia](#) Feb 28 '12 at 20:15
-
- 36 Wow this is an amazing answer for such a (some what) flippant question! Good work! – [jk1p](#) Mar 31 '12 at 6:46
-
- 10 This is one incredible example on how to use a file upload in Django. I'm blown away at how detailed and comprehensive it is. Great job. – [Helen Neely](#) Oct 5 '12 at 14:54
-
- 4 This is great, it blows the standard doc example out of the water. Thanks for taking the time to write this out. – [itsachen](#) Oct 23 '12 at 22:18
-
- 5 Found the location in django docs that shows file uploads. The example in this answer is excellent, but the info in the django docs will be kept up to date with new releases. docs.djangoproject.com/en/dev/topics/http/file-uploads – [TaiwanGrapefruitTea](#) Nov 29 '12 at 8:20

[show 13 more comments](#)

39 Generally speaking when you are trying to 'just get a working example' it is best to 'just start writing code'. There is no code here to help you with, so it makes answering the question a lot more work for us.

If you want to grab a file, you need something like this in an html file somewhere:

```
<form method="post" enctype="multipart/form-data">
  <input type="file" name="myfile" />
  <input type="submit" name="submit" value="Upload" />
</form>
```

That will give you the browse button, an upload button to start the action (submit the form) and note the enctype so Django knows to give you `request.FILES`

In a view somewhere you can access the file with

```
def myview(request):
    request.FILES['myfile'] # this is my file
```

There is a huge amount of information in the [file upload docs](#)

I recommend you read the page thoroughly and *just start writing code* - then come back with examples and stack traces when it doesn't work.

share improve this answer

answered May 3 '11 at 15:25



Henry

3,704 ● 10 ● 21

-
- 6 Thanks Henry. Actually I've read the docs and have written some code but since the docs has some gaps (for example "from somewhere import handle_uploaded_file") and my code was flawed, thought that it would be much better if I could start from a working example. – [qliq](#) May 3 '11 at 15:32
-
- 10 Agree with qliq. A simple working example is the most efficient way to get newbies going, not docs – [Philip007](#) Oct 29 '12 at 7:42
-
- 5 The `enctype="multipart/form-data"` what I needed to make this work. thanks! – [john-charles](#) Jan 9 '13

at 20:25

Too much bla bla in the other answers, your's is perfect! – [rsm](#) Aug 11 '15 at 22:18

1 Just don't miss the {% csrf_token %} within the form tags. – [jonincanada](#) Nov 15 '15 at 22:28

[add a comment](#)

Update of [Akseli Palén's answer](#). see the [github repo](#)

25 Start a Django Project

1). Create a project: `django-admin.py startproject sample`

now a folder(`sample`) is created:

```
sample/
  manage.py
  sample/
    __init__.py
    settings.py
    urls.py
    wsgi.py
```

2). On `setting.py` add:

```
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
MEDIA_URL = '/media/'
```

4). `urls.py` add:

```
...<other imports>...
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = patterns('',
    url(r'^upload/$', 'uploader.views.home', name='imageupload'),
    ...<other url patterns>...
)+ static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Create a Django App:

5). Create an app: `python manage.py startapp uploader`

6). Now a folder(`uploader`) with these files are created:

```
uploader/
  __init__.py
  models.py
  admin.py
  tests.py
  views.py
```

7). On `setting.py` -> `INSTALLED_APPS` add `'uploader',` , ie:

```
INSTALLED_APPS = (
    ...
    'uploader',
    ...
)
```

8) update `models.py`

```
from django.db import models
from django.forms import.ModelForm

class Upload(models.Model):
    pic = models.ImageField("Image", upload_to="images/")
    upload_date=models.DateTimeField(auto_now_add =True)

# FileUpload form class.
class UploadForm(ModelForm):
    class Meta:
        model = Upload
```

9) update `views.py`

```
from django.shortcuts import render
from uploader.models import UploadForm,Upload
from django.http import HttpResponseRedirect
from django.core.urlresolvers import reverse
# Create your views here.
def home(request):
    if request.method=="POST":
        img = UploadForm(request.POST, request.FILES)
        if img.is valid():
```

```

        img.save()
        return HttpResponseRedirect(reverse('imageupload'))
    else:
        img=UploadForm()
        images=Upload.objects.all()
        return render(request, 'home.html', {'form':img, 'images':images})

```

10). Create a folder `templates` and create a file `home.html` :

```

<div style="padding:40px;margin:40px;border:1px solid #ccc">
  <h1>picture</h1>
  <form action="#" method="post" enctype="multipart/form-data">
    {% csrf_token %} {{form}}
    <input type="submit" value="Upload" />
  </form>
  {% for img in images %}
    {{forloop.counter}}.<a href="{{ img.pic.url }}">{{ img.pic.name }}</a>
    ({{img.upload_date}})<hr />
  {% endfor %}
</div>

```

Project tree:

```

sample/
  manage.py
  sample/
    __init__.py
    settings.py
    urls.py
    wsgi.py
  uploader/
    __init__.py
    models.py
    views.py
    templates/
      home.html

```

11). Synchronize database and runserver:


```

python manage.py syncdb
python manage.py runserver -- should be localhost.com:8000/upload>; This worked for django 1.6 and
Python 3.3. – Steve Jun 10 '14 at 15:02
visit <http://localhost.com:8000>
+1 for reusable django app design pattern – Marcel Jun 17 '14 at 15:22

```

As I used a `FileField` while suhail used an `ImageField`, could someone please explain the choices?
 – David Tan Nov 15 '15 at 20:06

good answer to understand quick – Amir Apr 12 at 12:31

 **suhail**
 5,300 ● 3 ● 26 ● 41

add a comment

Extending on Henry's example:

```

import tempfile
import shutil

FILE_UPLOAD_DIR = '/home/imran/uploads'

def handle_uploaded_file(source):
    fd, filepath = tempfile.mkstemp(prefix=source.name, dir=FILE_UPLOAD_DIR)
    with open(filepath, 'wb') as dest:
        shutil.copyfileobj(source, dest)
    return filepath

```

You can call this `handle_uploaded_file` function from your view with the uploaded file object. This will save the file with a unique name (prefixed with filename of the original uploaded file) in filesystem and return the full path of saved file. You can save the path in database, and do something with the file later.

share improve this answer

answered May 3 '11 at 15:42



Imran

28.4k ● 15 ● 70 ● 105

Imran, I tried your code on my view but got this error: 'WSGIRequest' object has no attribute 'name'. – qliq
 May 3 '11 at 16:14

2 Pass the uploaded file object (`request.FILES['myfile']`) to `handle_uploaded_file`, not the `request`

[add a comment](#)

9 I must say I find the documentation at django confusing. Also for the simplest example why are forms being mentioned? The example I got to work in the views.py is :-

```
for key, file in request.FILES.items():
    path = file.name
    dest = open(path, 'w')
    if file.multiple_chunks():
        for c in file.chunks():
            dest.write(c)
    else:
        dest.write(file.read())
    dest.close()
```

The html file looks like the code below, though this example only uploads one file and the code to save the files handles many :-

```
<form action="/upload_file/" method="post" enctype="multipart/form-data">{% csrf_token %}
<label for="file">Filename:</label>
<input type="file" name="file" id="file" />
<br />
<input type="submit" name="submit" value="Submit" />
</form>
```

These examples are not my code, they have been obtained from two other examples I found. I am a relative beginner to django so it is very likely I am missing some key point.

[share](#) [improve this answer](#)

edited Apr 7 '13 at 20:28



Ray

816 ● 7 ● 14

answered Jun 7 '12 at 12:11



jimscafe

461 ● 1 ● 6 ● 14

[add a comment](#)

4 You can refer to server examples in Fine Uploader, which has django version.

<https://github.com/FineUploader/server-examples/tree/master/python/django-fine-uploader>

It's very elegant and most important of all, it provides featured js lib. Template is not included in server-examples, but you can find demo on its website. Fine Uploader: <http://fineuploader.com/demos.html>

django-fine-uploader

views.py

UploadView dispatches post and delete request to respective handlers.

```
class UploadView(View):

    @csrf_exempt
    def dispatch(self, *args, **kwargs):
        return super(UploadView, self).dispatch(*args, **kwargs)

    def post(self, request, *args, **kwargs):
```

```

"""A POST request. Validate the form and then handle the upload
based on the POSTed data. Does not handle extra parameters yet.
"""
form = UploadFileForm(request.POST, request.FILES)
if form.is_valid():
    handle_upload(request.FILES['qqfile'], form.cleaned_data)
    return make_response(content=json.dumps({ 'success': True }))
else:
    return make_response(status=400,
        content=json.dumps({
            'success': False,
            'error': '%s' % repr(form.errors)
        }))

def delete(self, request, *args, **kwargs):
    """A DELETE request. If found, deletes a file with the corresponding
    UUID from the server's filesystem.
    """
    qquuid = kwargs.get('qquuid', '')
    if qquuid:
        try:
            handle_deleted_file(qquuid)
            return make_response(content=json.dumps({ 'success': True }))
        except Exception, e:
            return make_response(status=400,
                content=json.dumps({
                    'success': False,
                    'error': '%s' % repr(e)
                }))
    return make_response(status=404,
        content=json.dumps({

```

forms.py

```

class UploadFileForm(forms.Form):
    """ This form represents a basic request from Fine Uploader.
    The required fields will always be sent, the other fields are optional
    based on your setup.
    Edit this if you want to add custom parameters in the body of the POST
    request.
    """
    qqfile = forms.FileField()
    qquuid = forms.CharField()
    qqfilename = forms.CharField()
    qqpartindex = forms.IntegerField(required=False)
    qqchunksize = forms.IntegerField(required=False)
    qqpartbyteoffset = forms.IntegerField(required=False)
    qqtotalfilesize = forms.IntegerField(required=False)
    qqtotalparts = forms.IntegerField(required=False)

```

share improve this answer

edited Nov 30 '15 at 13:02



Gaurav Jain
384 ● 5 ● 17

answered Apr 7 '15 at 4:35



Jiawei Dai
236 ● 2 ● 5

While this link may answer the question, it is better to include the essential parts of the answer here and provide the link for reference. Link-only answers can become invalid if the linked page changes.

– Suhaib Janjua Apr 7 '15 at 5:06

add a comment

Here it may helps you: create a file field in your models.py

2

For uploading the file(in your admin.py):

```

def save_model(self, request, obj, form, change):
    url = "http://img.youtube.com/vi/%s/hqdefault.jpg" %(obj.video)
    url = str(url)

    if url:
        temp_img = NamedTemporaryFile(delete=True)
        temp_img.write(urllib2.urlopen(url).read())
        temp_img.flush()
        filename_img = urlparse(url).path.split('/')[-1]
        obj.image.save(filename_img, File(temp_img))

```

and use that field in your template too.

share improve this answer

edited Feb 17 '14 at 14:07



Vlad T.
696 ● 9 ● 25

answered Oct 20 '13 at 10:39



Vijesh Venugopal
335 ● 2 ● 13

This is useful, when you have to manually temper with files you want to save. If so, you may also need this section: docs.djangoproject.com/en/dev/topics/files/#the-file-object – kecske Jul 17 '14 at 22:25

[add a comment](#)

2

I also had the similar requirement. Most of the examples on net are asking to create models and create forms which I did not wanna use. Here is my final code.

```
if request.method == 'POST':
    file1 = request.FILES['file']
    contentOfFile = file1.read()
    if file1:
        return render(request, 'blogapp/Statistics.html', {'file': file1, 'contentOfFile': cor
```

And in HTML to upload I wrote:

```
{% block content %}
<h1>File content</h1>
<form action="{% url 'blogapp:uploadComplete'%}" method="post" enctype="multipart/form-dat
{% csrf_token %}
<input id="uploadbutton" type="file" value="Browse" name="file" accept="text/csv" />
<input type="submit" value="Upload" />
</form>
{% endblock %}
```

Following is the HTML which displays content of file:

```
{% block content %}
<h3>File uploaded successfully</h3>
{{file.name}}
</br>content = {{contentOfFile}}
{% endblock %}
```

[share](#) [improve this answer](#)

answered Nov 10 '14 at 2:16



chetan pawar
35 ● 5

[add a comment](#)

0

Not sure if there any disadvantages to this approach but even more minimal, in views.py:

```
entry = form.save()

# save uploaded file
if request.FILES['myfile']:
    entry.myfile.save(request.FILES['myfile']._name, request.FILES['myfile'], True)
```

[share](#) [improve this answer](#)

[edited Sep 8 '14 at 11:14](#)

answered May 19 '11 at 13:00



Juan Céspedes
814 ● 2 ● 18



PhoebeB
3,655 ● 4 ● 33 ● 55

[add a comment](#)

-2

I faced the similar problem, and solved by django admin site.

```
# models
class Document(models.Model):
    docfile = models.FileField(upload_to='documents/Temp/%Y/%m/%d')

    def doc_name(self):
        return self.docfile.name.split('/')[-1] # only the name, not full path

# admin
from myapp.models import Document
class DocumentAdmin(admin.ModelAdmin):
    list_display = ('doc_name',)
admin.site.register(Document, DocumentAdmin)
```

[share](#) [improve this answer](#)

answered May 14 '14 at 6:58



hlpmee
10 ● 2

[add a comment](#)

Not the answer you're looking for? Browse other questions tagged [django](#) [file](#) [upload](#) or [ask your own question](#).

[about us](#) [tour](#) [help](#) [blog](#) [chat](#) [data](#) [legal](#) [privacy policy](#) [work here](#) [advertising info](#) [mobile](#) [contact us](#) [feedback](#)

TECHNOLOGY

[Stack Overflow](#)

[Server Fault](#)

[Super User](#)

[Web Applications](#)

[Ask Ubuntu](#)

[Webmasters](#)

[Game Development](#)

[TeX - LaTeX](#)

[Programmers](#)

[Unix & Linux](#)

[Ask Different \(Apple\)](#)

[WordPress](#)

[Development](#)

[Geographic Information](#)

[Systems](#)

[Electrical Engineering](#)

[Android Enthusiasts](#)

[Information Security](#)

[Database](#)

[Administrators](#)

[Drupal Answers](#)

[SharePoint](#)

[User Experience](#)

[Mathematica](#)

[Salesforce](#)

[ExpressionEngine®](#)

[Answers](#)

[more \(13\)](#)

LIFE / ARTS

[Photography](#)

[Science Fiction &
Fantasy](#)

[Graphic Design](#)

[Movies & TV](#)

[Seasoned Advice](#)

[\(cooking\)](#)

[Home Improvement](#)

[Personal Finance &](#)

[Money](#)

[Academia](#)

[more \(9\)](#)

CULTURE / RECREATION

[English Language &
Usage](#)

[Skeptics](#)

[Mi Yodeya \(Judaism\)](#)

[Travel](#)

[Christianity](#)

[Arqade \(gaming\)](#)

[Bicycles](#)

[Role-playing Games](#)

[more \(21\)](#)

SCIENCE

[Mathematics](#)

[Cross Validated \(stats\)](#)

[Theoretical Computer
Science](#)

[Physics](#)

[MathOverflow](#)

[Chemistry](#)

[Biology](#)

[more \(5\)](#)

OTHER

[Stack Apps](#)

[Meta Stack Exchange](#)

[Area 51](#)

[Stack Overflow Careers](#)

site design / logo © 2016 Stack Exchange Inc; user contributions licensed under [cc by-sa 3.0](#) with [attribution required](#)
rev 2016.4.22.3499