django

yes, the movie
jk ur a fool

last week

- regexes!
- urls!
- models!
- views!
- templates!
- exclamation points!!!!!!!!!!!!!!!!!!!!

this week

- admin!
- moar templates!
- migrations!
- gifs!
- fewer exclamation points.
- jk!!!!!!!!!!!!!!!!!!!!!!

non-technical users

non-technical users

- need to upload content
- don't know shit about computers
- how do they do it?

admin interface!

- automagically generates CRUD interface
- also pretty customizable

but how to?

1. create file `<app_name>/admin.py`
2. import admin
3. register your model with admin

admin.py

```
from django.contrib import admin
from blogger.models import BlogPost, Author

admin.site.register(BlogPost)
admin.site.register(Author)
```

is it really that easy?

- yes.
- yes it is.

some customization

- ModelAdmin - class that contains settings

some customization

```
from django.contrib import admin
from blogger.models import BlogPost, Author
```

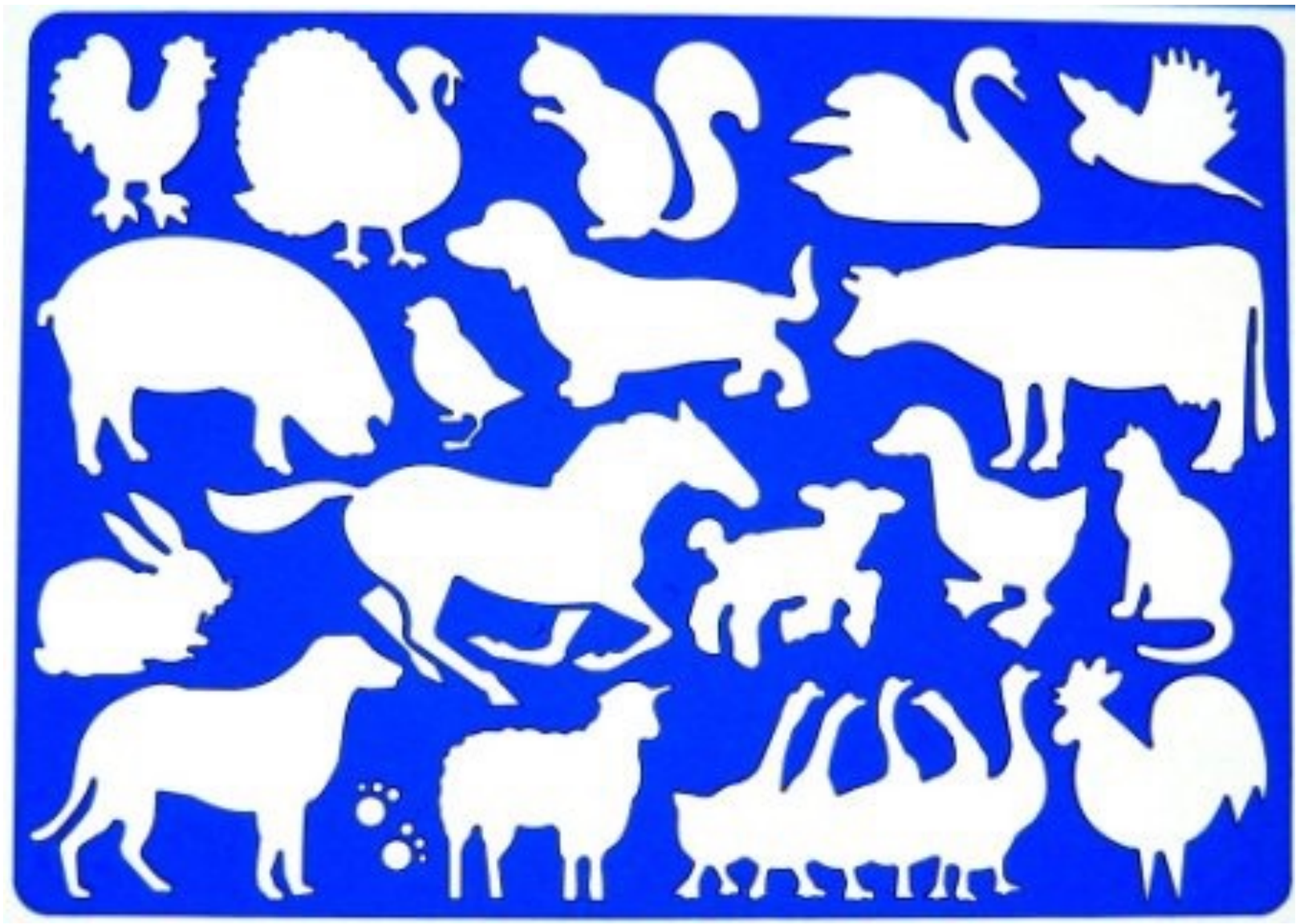
```
class BlogPostAdmin(admin.ModelAdmin):
    fields = ['title', 'post', 'time', 'author']
```

```
admin.site.register(BlogPost, BlogPostAdmin)
```

moar customization

- widgets
- forms
- css i guess if that's what you're into

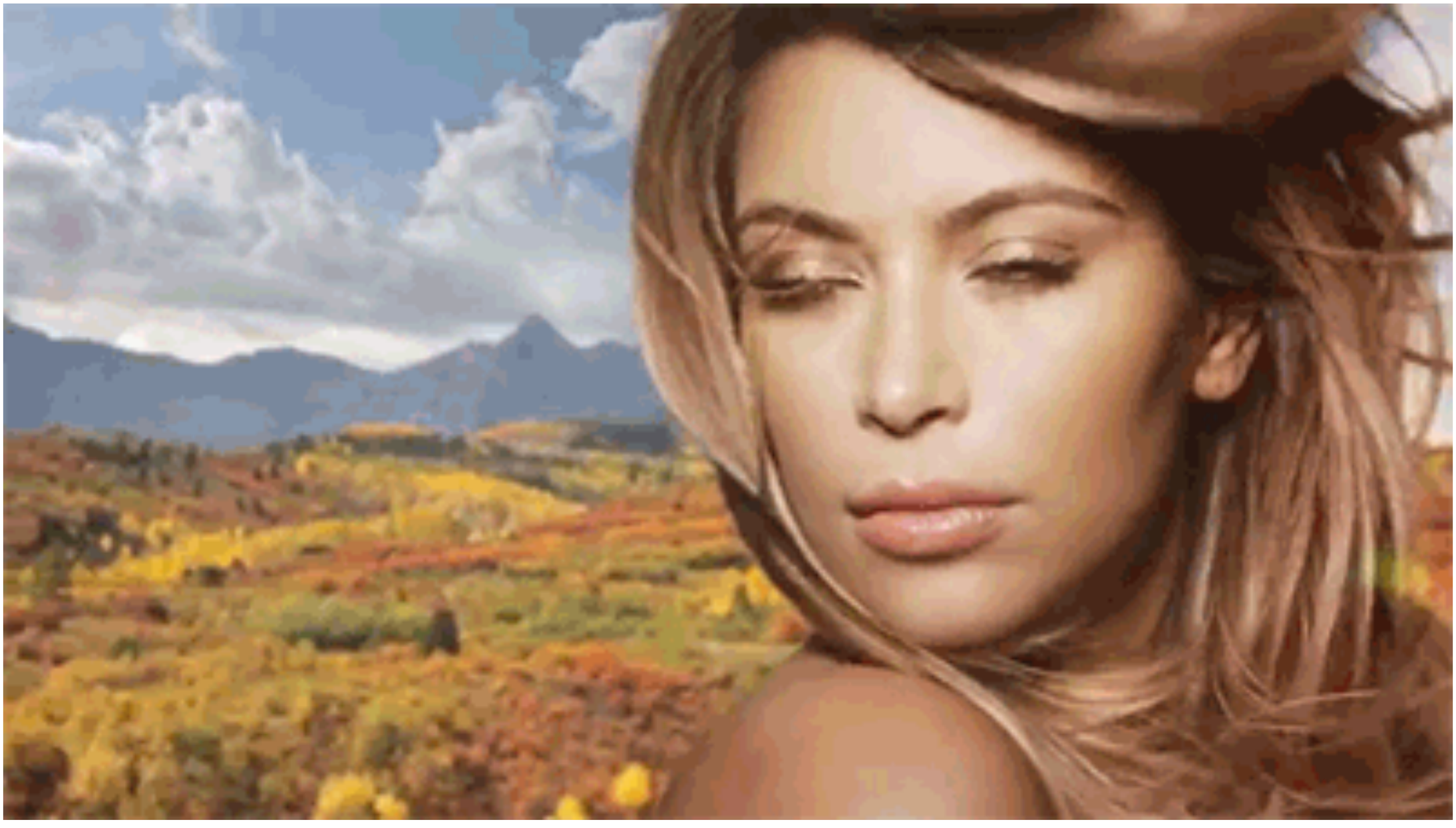
templates



going back to our blog site...

- jk
- that shit was boring
- let's talk kanye

going back to our blog site...



going back to our blog site...

going back to our blog site...

#1 kanye fan site

- **^$:** homepage
- **^/quote/\d+$:** quote page
- **^/fan/rank_\d+$:** ye's number *n* fan

index.html

```
<html>
  <head>
    <title>kanye's biggest baddest fansite</title>
  </head>
  <body>
    <ul id="best quotes">
      {% for q in quotes %}
        <li><a href="{{ q.url }}">{{ q }}</a></li>
      {% endfor %}
    </ul>
  </body>
</html>
```

quotes.html

```
<html>
  <head>
    <title>kanye's biggest baddest fansite</title>
  </head>
  <body>
    <h1>quote</h1>
    <p id="quote">{{ quote }}</p>
    <div id="comments">
      ...
    </div>
  </body>
</html>
```

fan.html

```
<html>
  <head>
    <title>kanye's biggest baddest fansite</title>
  </head>
  <body>
    <h1>kanye's #{{ fan.rank }} fan</h1>
    <span id="name">{{ fan.name }}</span>
    <div id="stats">
      {{ fan.stats }}
    </div>
  </body>
</html>
```

what's wrong with this picture?

template inheritance

- define "base" template
- define "blocks"
- child templates fill in blocks

_base.html

```
<html>
  <head>
    <title>kanye's biggest baddest fansite</title>
  </head>
  <body>
    {% block content %} {% endblock content %}
  </body>
</html>
```

index.html (w/ facelift)

```
{% extends "_base.html" %}

{% block content %}
  <ul id="best quotes">
    {% for q in quotes %}
      <li><a href="{{ q.url }}">{{ q }}</a></li>
    {% endfor %}
  </ul>
{% endblock content %}
```

logic in templates (yay!)

- template filters!
- define functions!
- transform your inputs!
- moar info!

how to make

- create directory `<your_app>/templatetags`
- create files
  - `templatetags/__init__.py`
  - `templatetags/my_templatetags.py`

my_templatetags.py

```
from django import template
register = template.Library()

@register.filter()
def excite(val):
    return val.upper()
```

usage (blog_post.html)

```
{% extends "base.html" %}
{% load blogger_extras %}

{% block content %}
    <h1>{{ post.title|excite }}</h1>
    ... more stuff here too probably
{% endblock %}
```

migrations :o

- guys guys guys
- huge development guys
- i added a field to the model :o
- wat do

migrations

1. you add a field to your models
2. you run `python manage.py makemigrations`

    1. `python manage.py makemigrations –name changed_my_model your_app_label`

3. creates migration file

but what is migration file

- numbered
- code automatically updates db
- run them with with `python manage.py migrate`