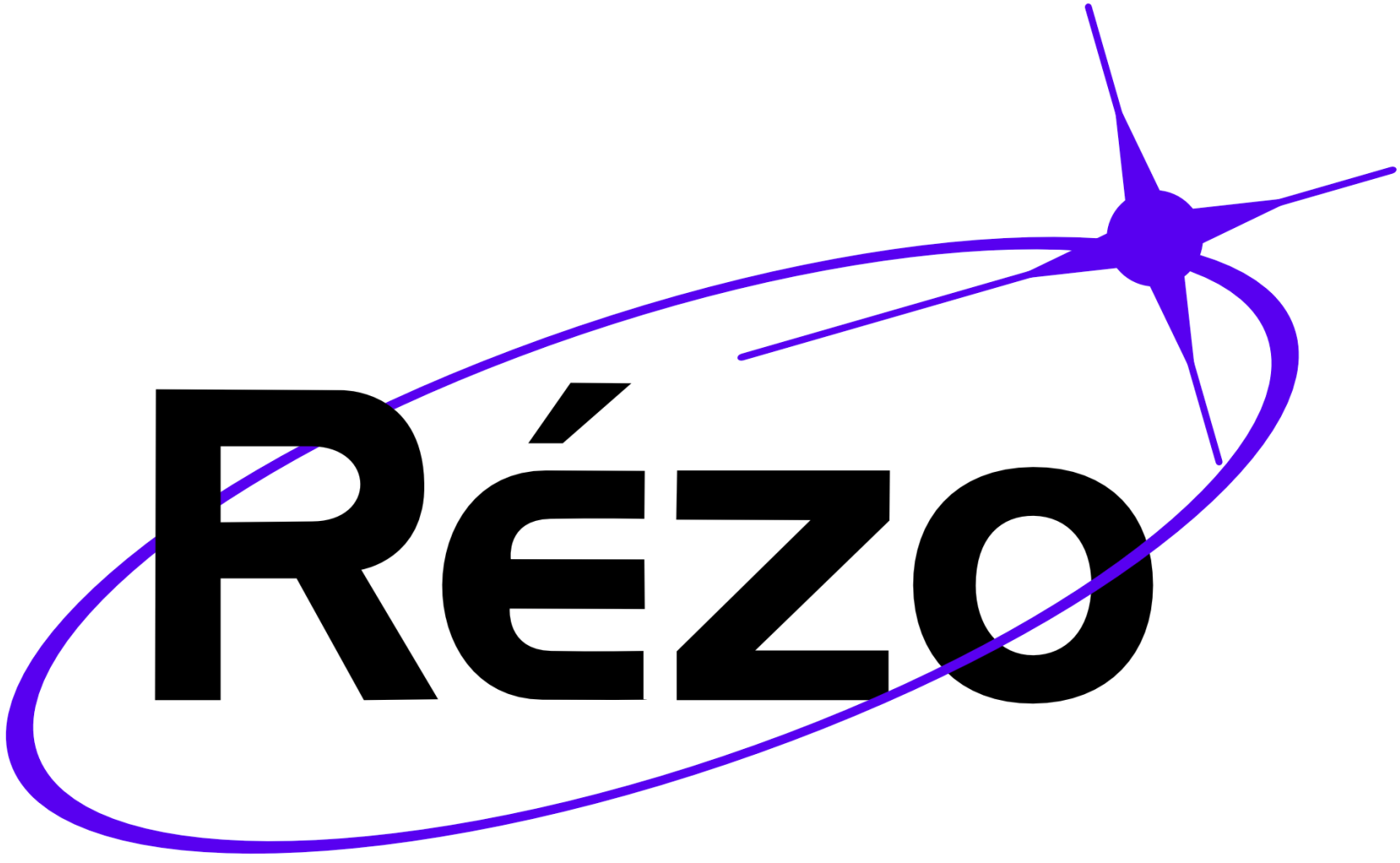

RÉZO


# Une formation par Klafyvel et Nanoy2

# Qu'est-ce que Django peut faire ?

# Qu'est-ce que Django peut faire ?

- coope.rez
- Re2o
- Le site de la NASA
- Blogs
- ...

# Qu'est-ce que Django ne peut pas faire ?

# Qu'est-ce que Django ne peut pas faire ?

- Rien

# Généralités sur Python : PIP

Installation :

```
sudo apt install python3-pip
```

Utilisation :

```
pip3 install truc # installe truc
pip3 uninstall machin # vire truc
pip3 freeze > requirements.txt # Sauvegarde les packages
# installés
pip3 install -r requirements.txt # Installe les packages
# listés dans requirements.txt
```

# Généralités sur Python : VirtualEnv

**(ou comment ne pas polluer son PC)**

Installation :

```
pip3 install virtualenv
```

Utilisation :

```
virtualenv env_formation
source env_formation/bin/activate
```

# Généralités sur Python : VirtualEnvWrapper

**(réservé aux gens supérieurs sous linux)**

Installation :

```
pip install --user virtualenvwrapper
```

Dans votre `.bashrc`

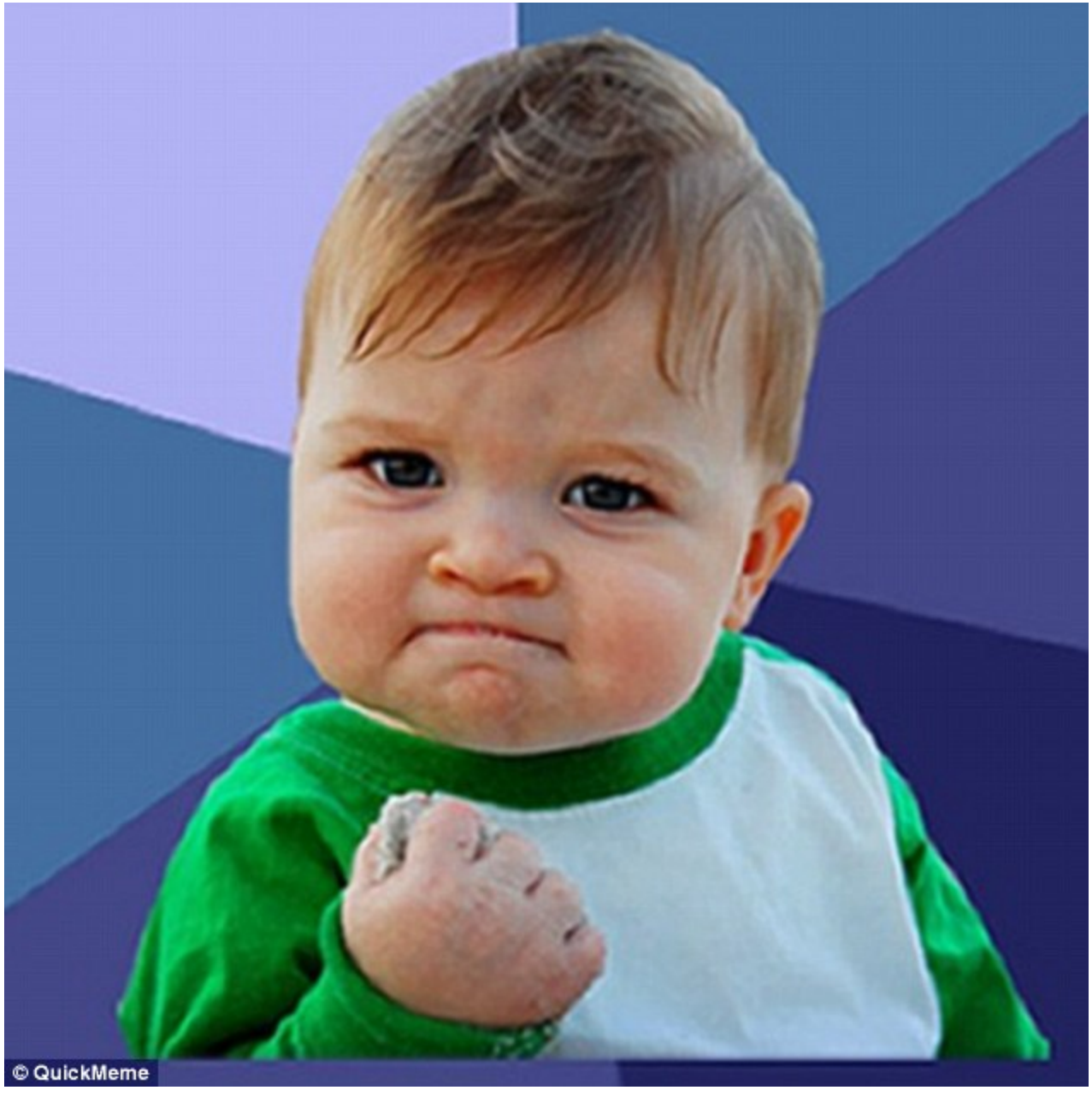
```
export WORKON_HOME=~/.virtualenvs
mkdir -p $WORKON_HOME
source ~/.local/bin/virtualenvwrapper.sh
```

Utilisation :

```
mkvirtualenv monprojet
workon monprojet
rmvirtualenv monprojet
```

# Mon premier site : Un blog

- Écrire des articles
- Lire des articles


© QuickMeme

# Comment démarrer un projet ?

Virtualenv :

```
cd là/où/vous/mettez/vos/projets/
virtualenv env_formation
source env_formation/bin/activate
```

VirtualenvWrapper :

```
mkvirtualenv env_formation
```

Création du projet :

```
pip install django
django-admin startproject mon_site
cd blog
./manage.py migrate
./manage.py runserver
```

127.0.0.1:8000

django

View release notes for Django 2.0

The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

**Django Documentation**
Topics, references, & how-to's

**Tutorial: A Polling App**
Get started with Django

**Django Community**
Connect, get help, or contribute

# Comment démarrer un projet ?

Création de l'application :

```
./manage.py startapp blog
```

Enregistrement de l'application ( dans `mon_site/settings.py` ) :

```
...
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'blog'
]
...
```

```
(env_formation)  klafyvel@batman > ~/mon_site > tree
.
├── blog
│   ├── admin.py
│   ├── apps.py
│   ├── __init__.py
│   ├── migrations
│   │   └── __init__.py
│   ├── models.py
│   ├── tests.py
│   └── views.py
├── db.sqlite3
├── manage.py
└── mon_site
    ├── __init__.py
    ├── __pycache__
    │   ├── __init__.cpython-36.pyc
    │   ├── settings.cpython-36.pyc
    │   └── urls.cpython-36.pyc
    ├── settings.py
    ├── urls.py
    └── wsgi.py
```

# L'architecture MVT

Models Views Templates

# M comme Model

Les imports

```
from django.db import models
```

# M comme Models

```
class Article(models.Model):
    """Un article sur mon super site."""
    text = models.TextField(verbose_name="Texte")
    title = models.CharField(
        max_length=255,
        verbose_name="Titre"
    )
    date = models.DateField(
        verbose_name="Date de parution"
    )

    def __str__(self):
        return "'{}' : {}".format(
            self.title,
            self.date
        )
```

# Modifier la base de données

```
./manage.py makemigrations blog
./manage.py migrate
```

# Time to play !

```
./manage.py shell
>>> from blog.models import Article
>>> a = Article()
>>> a
<Article: '' : None>
>>> from django.utils import timezone
>>> a.date = timezone.now()
>>> a.title = "Un super titre"
>>> a.text = "Un contenu vraiment très intéressant !"
>>> a
<Article: 'Un super titre' : 2018-04-07 12:34:01.509609+00
>>> a.save()
```

# Time to play !

```
>>> b = Article()
>>> b.title = "Un autre article"
>>> b.date = timezone.now()
>>> b.text = "Du contenu"
>>> b.save()
>>> Article.objects.all()
<QuerySet [<Article: 'Un super titre' : 2018-04-07>,
        <Article: 'Un autre article' : 2018-04-07>]>
```

```
>>> Article.objects.get(pk=1)
<Article: 'Un super titre' : 2018-04-07>
>>> Article.objects.order_by('date')
<QuerySet [<Article: 'Un super titre' : 2018-04-07>,
        <Article: 'Un autre article' : 2018-04-07>]>
```

# Time to play !

```
>>> import datetime
>>> d = datetime.timedelta(days=1)
>>> b.date += d
>>> b.save()
>>> Article.objects.filter(date__lte=timezone.now())
<QuerySet [<Article: 'Un super titre' : 2018-04-07>]>
```

# Mais quand est-ce qu'on affiche quelque chose dans le navigateur ?

# L'architecture MVT

## V comme Views

```
from django.shortcuts import render
from django.http import HttpResponse

def index(request):
    s = ("Bonjour et bienvenue"
        "sur mon super site trop cool")
    return HttpResponse(s)
```

# Routons mes bons

`blog/urls.py` (à créer) :

```
from django.urls import path
from . import views

app_name = "blog"
urlpatterns = [
    path('', views.index),
]
```

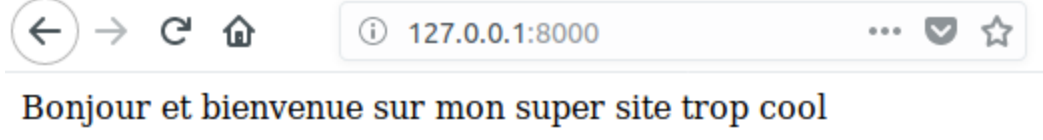
`mon_site/urls.py` :

```
...
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('blog.urls')),
]
```

## Lancer le serveur :

```
./manage.py runserver
```

## Tadaaaa :



# Afficher des données !

```
from django.shortcuts import render
from django.http import HttpResponse

from .models import Article

def index(request):
    articles = Article.objects.order_by('-date')
    s = ("Bonjour et bienvenue"
         " sur mon super site trop cool"
         "\nMes articles :"
    )
    for a in articles:
        s += a.title + "\n"
    return HttpResponse(s)
```

# Afficher des données !

## Votre site :


## Vous :

# L'architecture MVT

## T comme Templates

Dans `blog/templates/blog/list_articles.html` :

```
<h3>Liste des articles</h3>
{% for article in articles %}
<div>
  <h4>{{article.title}}</h4>
  <p>Article écrit le {{article.date}}</p>
{% endfor %}
```

# T comme Templates

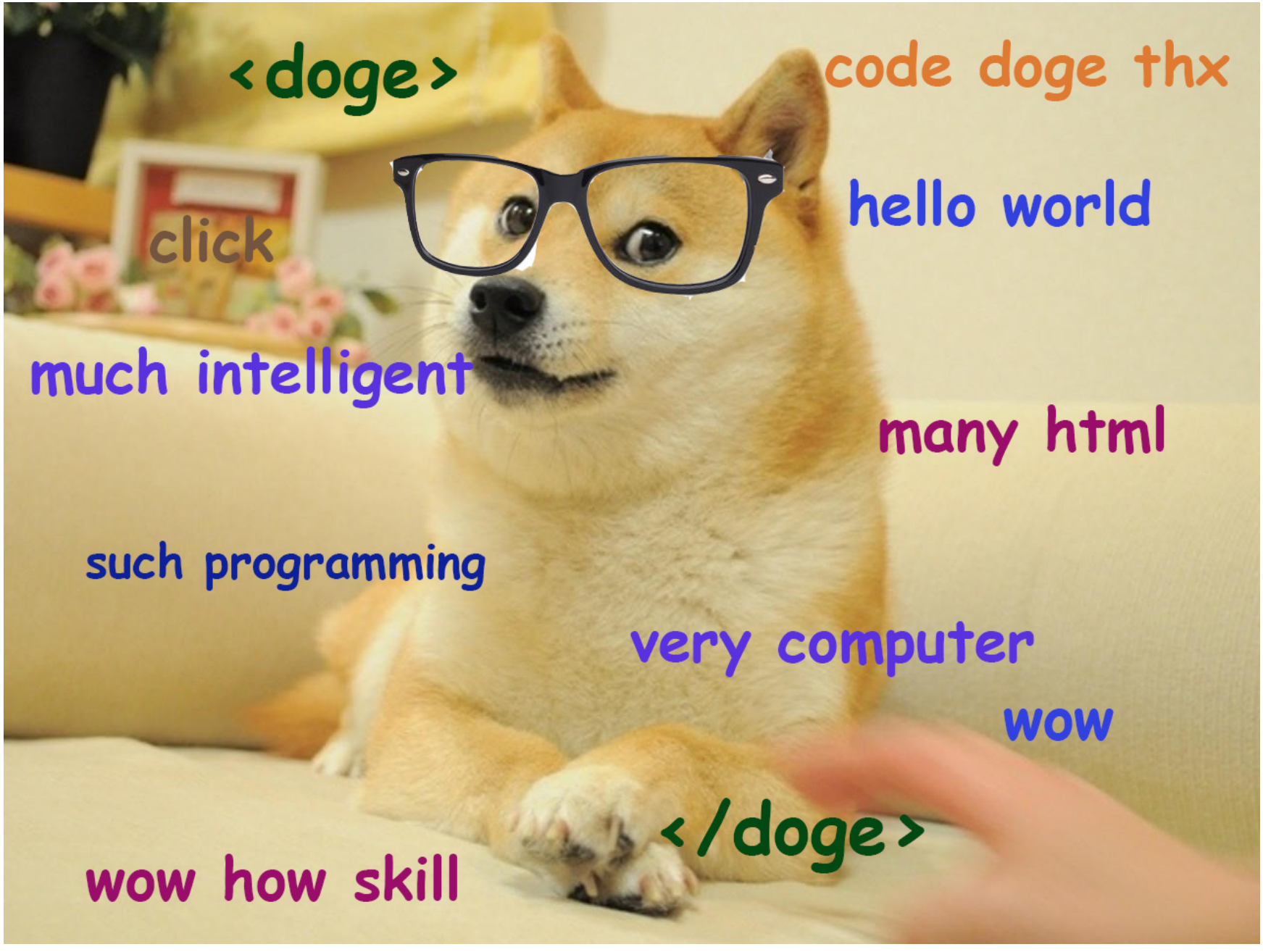
Dans `blog/views.py` :

```
from django.shortcuts import render
from django.http import HttpResponse

from .models import Article

def index(request):
    articles = Article.objects.order_by('-date')
    return render(
        request,
        'blog/list_articles.html',
        {'articles':articles}
    )
```

# Votre site :


127.0.0.1:8000
Liste des articles
Un autre article
Article écrit le April 8, 2018
Un super titre
Article écrit le April 7, 2018

## Vous :


<doge>
code doge thx
hello world
click
much intelligent
many html
such programming
very computer
wow
</doge>
wow how skill

# Étendre un template

Dans `templates/base.html` :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mon super blog</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <h1>Mon super titre qu'on verra partout</h1>
      {% block content %}{% endblock %}
  </body>
</html>
```

# Étendre un template

Dans `mon_site/settings.py` (ligne 55):

```
#...
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.Django
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug
                'django.template.context_processors.reques
                'django.contrib.auth.context_processors.au
                'django.contrib.messages.context_processor
            ],
        },
    },
]
#...
```

# Étendre un template

Dans `blog/templates/blog/list_articles.html`

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

{% block content %}
<h3>Liste des articles</h3>
{% for article in articles %}
<div>
  <h4>{{article.title}}</h4>
  <p>Article écrit le {{article.date}}</p>
{% endfor %}
{% endblock %}
```

# Votre site


127.0.0.1:8000
Mon super titre qu'on verra partout
Liste des articles
Un autre article
Article écrit le April 8, 2018
Un super titre
Article écrit le April 7, 2018

# Exercice : afficher un article

Objectif :

## Exercice : afficher un article

- Créer la vue dédiée ( `def view_article(request, pk):` )
- La remplir (conseil regarder `django.shortcuts.get_object_or_404` )
- Créer l'url dédiée dans `blog/urls.py` (elle sera de la forme `article/<int:pk>` )
- Créer le template associé (dans `blog/templates/blog/view_article.html` )

## Ma solution

Dans `blog/views.py` :

```
def view_article(request, pk):
    article = get_object_or_404(Article, pk=pk)
    return render(
        request,
        'blog/view_article.html',
        {'article':article}
    )
```

Dans `blog/urls.py` :

```
path('article/<int:pk>', views.view_article)
```

## Ma solution

Dans `blog/templates/blog/view_article.html` :

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

{% block content %}
<h2>{{article.title}}</h2>
Publié le {{article.date}}.
<br/>
<br/>
{{article.text}}
{% endblock %}
```

# Tags

## → Commandes pour les templates

Exemple : `{% url %}`

Dans `blog/urls.py` :

```
from django.urls import path
from . import views

app_name = "blog"
urlpatterns = [
    path('', views.index, name="index"),
    path(
        'article/<int:pk>',
        views.view_article,
        name="article"
    )
]
```

# Tags

Dans `blog/templates/blog/list_articles.html` :

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

{% block content %}
<h3>Liste des articles</h3>
{% for article in articles %}
  <h4>
    <a href="{% url 'blog:article' article.pk %}">
      {{article.title}}
    </a>
  </h4>
  <p>Article écrit le {{article.date}}</p>
{% endfor %}
{% endblock %}
```

# Tags

Dans `templates/base.html` :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mon super blog</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <h1>Mon super titre qu'on verra partout</h1>
    <a href="{% url 'blog:index' %}">
      Retour à l'accueil
    </a>
    {% block content %}{% endblock %}
    </body>
</html>
```


# Mon super titre qu'on verra partout

Retour à l'accueil

### Liste des articles

**Un autre article**

Article écrit le April 8, 2018

**Un super titre**

Article écrit le April 7, 2018


# Mon super titre qu'on verra partout

Retour à l'accueil

## Un autre article

Publié le April 8, 2018.

Du contenu

# Site admin

# Forms

# Sites intéressants

- Le blog Sam et Max
  - Article sur virtualenv
  - Article sur Pip
  - Un autre article pour comprendre à quel point l'écosystème Python c'est le feu
- La doc de Django
- Zeste de Savoir
- Djangogirl

# Demander de l'aide :

- Vos Rézo(wo)mens ❤️
  - IRC
  - Telegram
  - Mail
  - Facebook
- Forums
  - Zeste de Savoir
  - Stack Overflow