Django App & Template

Zayed

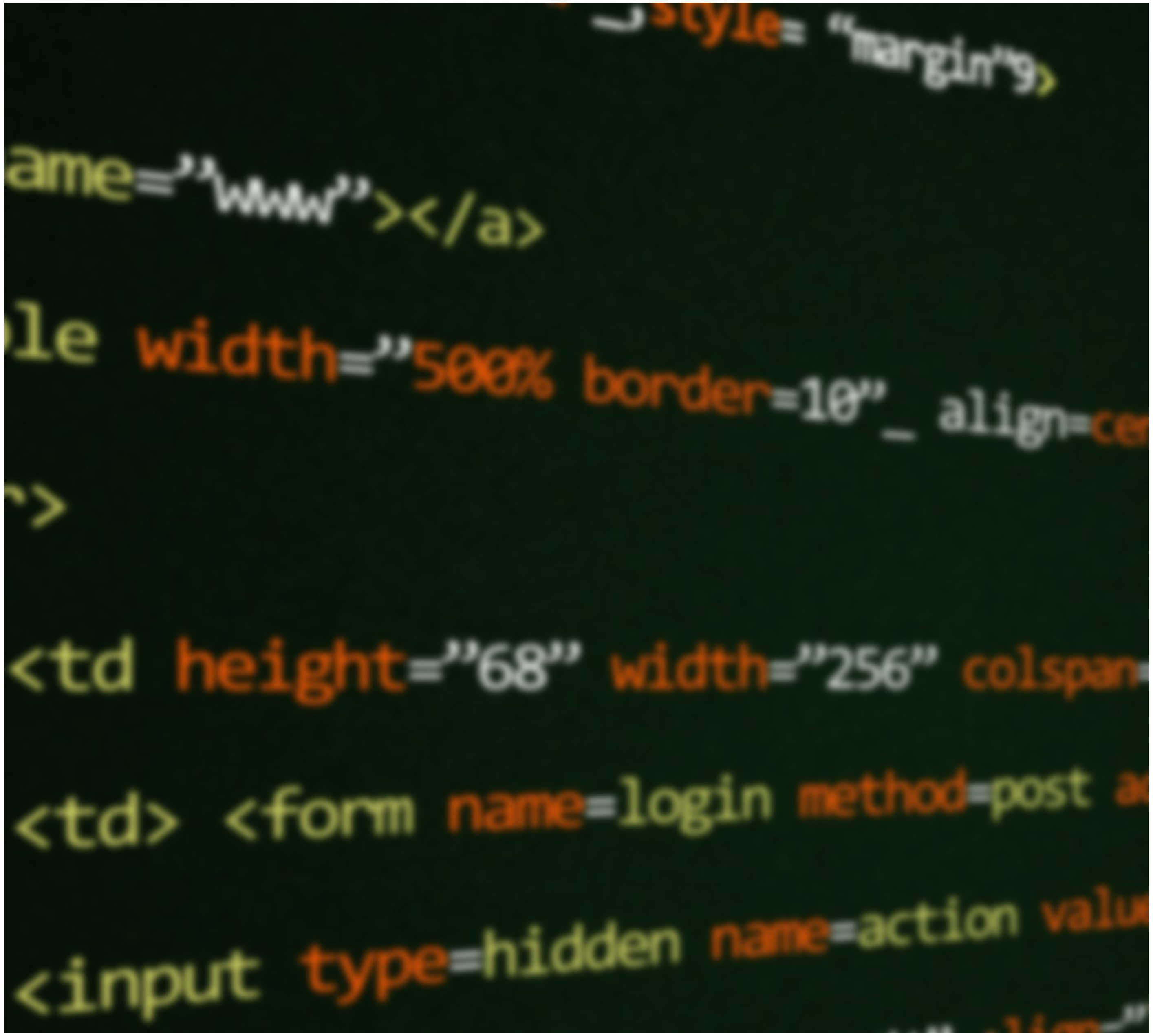
contact@zayedabdullah.com

Goal

Continuation from URL

Django Apps

Django Templates

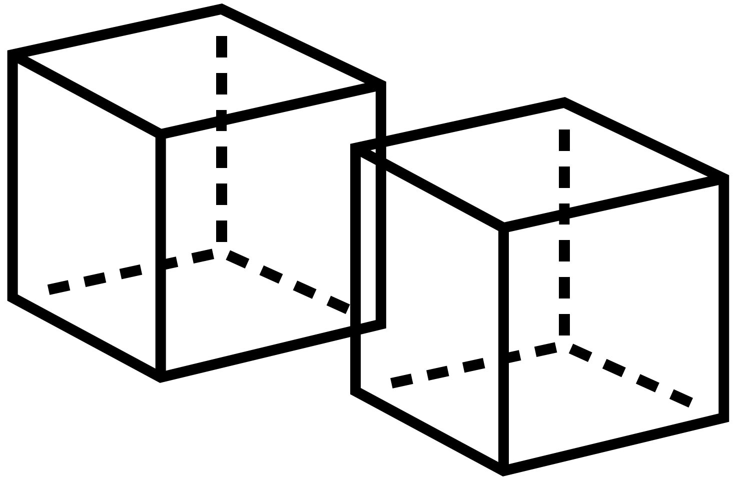


Continuation from URL

UNDERSTAND THE URL

DJANGO VIEW

SEPARATE THE PARTS



Understand the URL

DATA PASSING THROUGH THE URL

USING PLACEHOLDER IN URL

Understand the URL

How can we pass data through URL

Django supports passing data through URLs.

A URL string in a path function can hold **placeholders** to receive data in a “urlpattern” list like:

```
“/books/<book_id>/authors”
```

Understand the URL

How can we pass data through URL

So, the URL path should be like this:

```
path("/books/<book_id>/authors", author_list)
```

That book id is passed in the view function as the book_id variable like this:

```
def author_list(request, book_id):
        .....
```

Django View

DATA PASSING THROUGH THE URL

USING PLACEHOLDER IN URL

Django View

What & How

The view is simply the **function/class** that handles the **request-response** cycle and where response **logic** is written.

It is generally connected to the path function with the URL path in the "urlpatterns" named list.

Separate The Parts

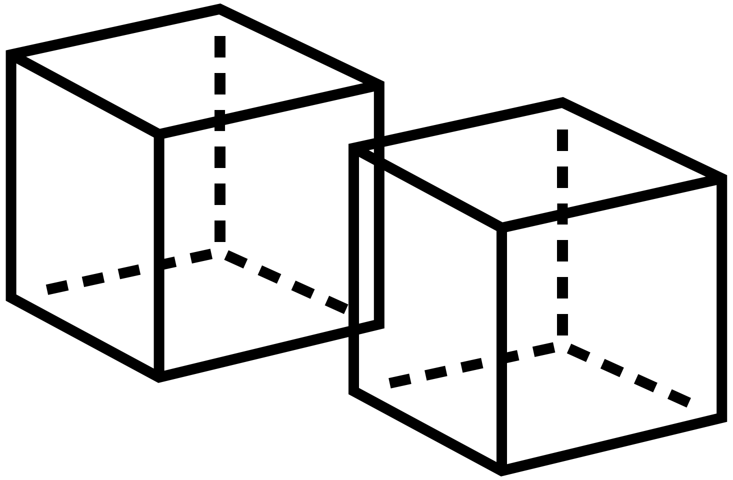
SEPARATE THE VIEW FUNCTIONS

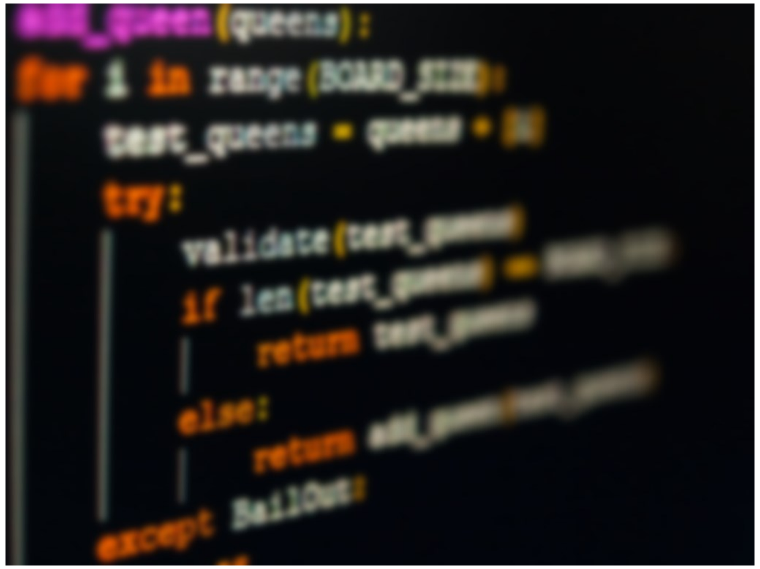
MAKE YOUR WAY THROUGH THE STRUCTURE

Django Apps

THE DJANGO APP & STRUCTURE

PUTTING THE PARTS WHERE THEY BELONG





```
add_queen(queens):
for i in range(BOARD_SIZE):
    test_queens = queens + [i]
    try:
        validate(test_queens)
        if len(test_queens) == BOARD_SIZE:
            return test_queens
        else:
            return add_queen(test_queens)
    except BailOut:
```

CodeWithVirus

Django Apps

What is a Django App

A Django App is simply a **directory** containing related files for a feature or part of a feature.

You can simply **group** related files for a specific feature or part of a feature in an app. So, your app stays clean and structured.

Django Apps

Create an App

```
cool_project
├── cool_project
│   ├── settings.py
│   └── urls.py
└── manage.py
```

Command:

```
python manage.py startapp <app_name>
```

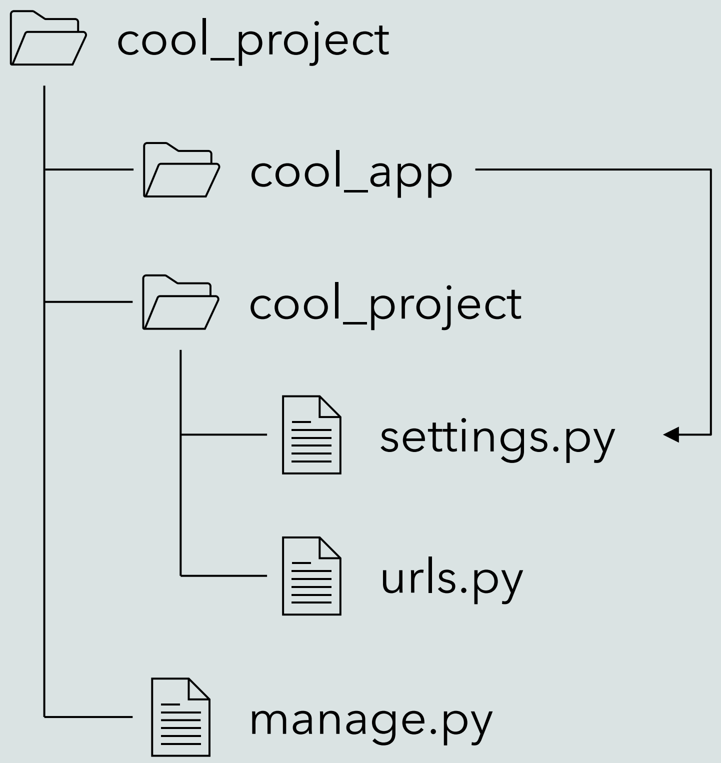
Example:

```
python manage.py startapp cool_app
```

```
── cool_app
```

Django Apps

Install the app into the Settings



```
INSTALLED_APPS= [
  ......,
  ......,
  ......,
  “cool_app”,
]
```

Django Apps

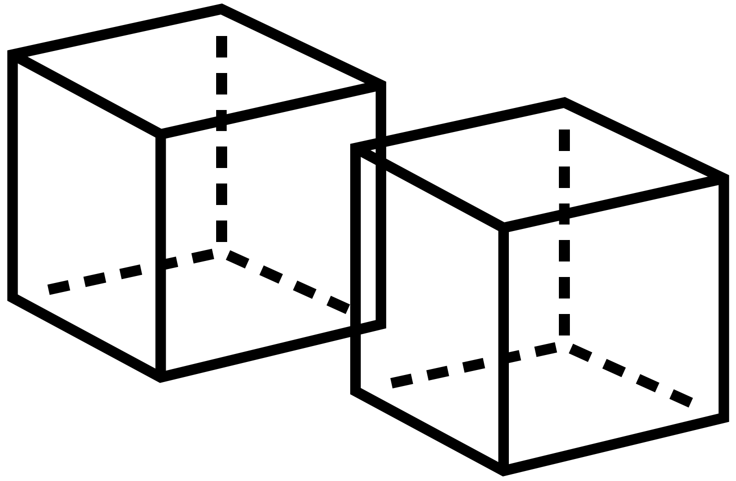
PUTTING THE PARTS WHERE THEY BELONG

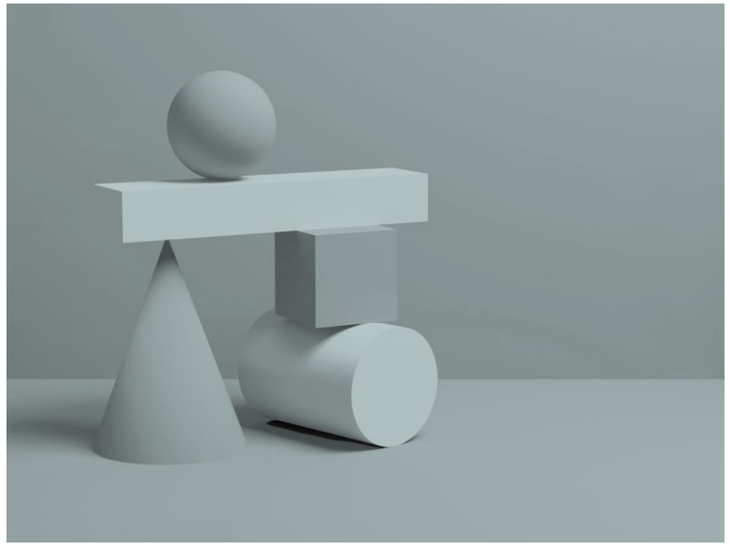
Django Templates

WHAT'S A DJANGO TEMPLATE

HOW TEMPLATE WORKS

RE-USING TEMPLATE





CodeWithVirus

Django Templates

What is a Django Template

A Django Template is simply a **text document** (mostly HTML) containing some **special tags** and **variables** to generate the document with dynamic content.

The Django template is like many common template libraries for Python like jinja2 and others. It supports **extending** a template, **including** a template, using **blocks** to define static or dynamic data into a block, using **static files**, and so on.

Django Templates

How Templating Works

Just simply passing the **HttpRequest object**, a **template** (mostly HTML) name where the file contains Django Template Tags / Variables and a **context object** in the "**render**" function will generate a HttpResponse with the final output.

Note: Context must be a mapping object (like **Dictionary**).

Django Templates

Template Finding Strategy | What we see

`home/templates`

- home.html
- cool_one.html

`about/templates`

- about.html
- cool_two.html

Django Templates

Template Finding Strategy | What actually happens

`Actual templates directory`

- home.html
- cool_one.html
- about.html
- cool_two.html

Variable in Template

USE VARIABLES PASSED IN CONTEXT

Template Include

MAKE TEMPLATE MODULAR

Template Include

Partial Templates

header.html

```
<header>
This is my cool header
</header>
```

footer.html

```
<footer>
This is my cool footer
</footer>
```

main.html

```
<!DOCTYPE html><html>
<head></head>
<body>

{% include “header.html” %}


<main>Main Section</main>

{% include “footer.html” %}


</body>
</html>
```

Template Include

Merged View

main.html

```
<!DOCTYPE html><html>
<head></head>
<body>

<header>
This is my cool header
</header>

<main>Main Section</main>

<footer>
This is my cool footer
</footer>

</body>
</html>
```

Template Extend

RE-USE DJANGO TEMPLATE

Template Extend

Re-using Django Template

layout.html

```
<!DOCTYPE html><html>
<head></head>
<body>

{% block body_content %}

{% endblock %}

</body>
</html>
```

index.html

```
{% extends “layout.html” %}

{% block body_content %}

<h1>
  This is my index page
</h1>

{% endblock %}
```

Template Extend

Merged View

index.html

```
<!DOCTYPE html><html>
<head></head>
<body>

<h1>
  This is my index page
</h1>

</body>
</html>
```

What’s Next

- Loops in Template
- Condition in Template
- Static files in Template
- Any who knows what lies ahead…

Thank you

Abdullah Zayed

contact@zayedabdullah.com

https://zayedabdullah.com

https://linkedin.com/in/abdullahzayed01