

Digital Career Institute

Python Course - Django Web Framework - Views and Templates



Goal of the Submodule

The goal of this submodule is to introduce the learners to Django Views and Templates. By the end of this submodule, the learners will be able to:

- Define function and class-based views.
- Know the different properties and methods of the request and response objects.
- Know how to render templates using a context.
- Use template tags and variables to populate the template using the context data.

Topics

- Views and the request/response objects.
- Function and class-based views.
- Defining and using templates.
 - The context.
 - The `TemplateView`.
- Using the template language:
 - Template variables.
 - Template tags.
 - Template filters.
 - Defining custom tags and filters.
- Static URLs and files
 - The `static` template tag.
 - Serving `static` files in development and production.

Views

Views

shop/views.py

```
from django.http import HttpResponse

def home(request):
    """The shop home view."""
    text = "Hello World!"
    if some condition:
        text = "Hey People!"
    return HttpResponse(text)
```

Django's **views** store the **logic** of our application.

Views

shop/views.py

```
from django.http import HttpResponse

def home(request):
    """The shop home view."""
    text = "Hello World!"
    if some condition:
        text = "Hey People!"
    content = f"<html><body><h1>{text}</h1></body></html>"
    return HttpResponse(content)
```

Django returns to the browser the content of the argument used in **HttpResponse**.

The Request Object

shop/views.py

```
from django.http import HttpResponse

def home(request) :
    """The shop home view."""
    text = "Hello World!"
    if some condition:
        text = "Hey People!"
    return HttpResponse(text)
```

All views always receive a first argument that contains a **request object**.

The Request Object

- **request.method**
- request.GET
- request.POST
- request.COOKIES
- request.FILES
- request.META
- request.session
- request.user

shop/views.py

```
from django.http import HttpResponseRedirect

def home(request):
    """The shop home view."""
    if request.method == "GET":
        text = "Hello Visitor!"
    elif request.method == "POST":
        text = "Not allowed to post data."
    return HttpResponseRedirect(text)
```

The Request Object

- request.method
- **request.GET**
- request.POST
- request.COOKIES
- request.FILES
- request.META
- request.session
- request.user

shop/views.py

*http://localhost:8000/shop/browse/?**sort_by**=price*

```
from django.http import HttpResponseRedirect

def browse(request):
    """Browse the shop."""
    sort_by = request.GET.get("sort_by")
    if sort_by:
        # do something
    return HttpResponseRedirect(text)
```

The Request Object

- request.method
- request.GET
- **request.POST**
- request.COOKIES
- request.FILES
- request.META
- request.session
- request.user

shop/views.py

```
from django.http import HttpResponseRedirect

def buy_item(request):
    """Buy an item."""
    item_id = request.POST.get("item_id")
    if item_id:
        # do something
    return HttpResponseRedirect(text)
```

The Request Object

- request.method
- request.GET
- request.POST
- **request.COOKIES**
- request.FILES
- request.META
- request.session
- request.user

shop/views.py

```
from django.http import HttpResponseRedirect

def browse(request):
    """Browse the shop."""
    sessionid = request.COOKIES.get("sessionid")
    if sessionid:
        # do something
    return HttpResponseRedirect(text)
```

The Request Object

- request.method
- request.GET
- request.POST
- request.COOKIES
- **request.FILES**
- request.META
- request.session
- request.user

shop/views.py

```
from django.http import HttpResponse

def browse(request):
    """Browse the shop."""
    avatar = request.FILES.get("avatar")
    if avatar:
        # do something
    return HttpResponse(text)
```

The Request Object

- request.method
- request.GET
- request.POST
- request.COOKIES
- request.FILES
- **request.META**
- request.session
- request.user

shop/views.py

```
from django.http import HttpResponseRedirect

def browse(request):
    """Browse the shop."""
    referer = request.META.get("HTTP_REFERER")
    if referer:
        # do something
    return HttpResponseRedirect(text)
```

The Request Object

- request.method
- request.GET
- request.POST
- request.COOKIES
- request.FILES
- request.META
- **request.session**
- request.user

shop/views.py

```
from django.http import HttpResponse

def browse(request):
    """Browse the shop."""
    sessionid = request.session.get("sessionid")
    if sessionid:
        # do something
    return HttpResponse(text)
```

The Request Object

- request.method
- request.GET
- request.POST
- request.COOKIES
- request.FILES
- request.META
- request.session
- **request.user**

shop/views.py

```
from django.http import HttpResponseRedirect

def browse(request):
    """Browse the shop."""
    is_logged = request.user.is_authenticated
    if is_logged:
        # do something
    return HttpResponseRedirect(text)
```


The Response Object

shop/views.py

```
from django.http import HttpResponse

def home(request):
    """The shop home view."""
    text = "Hello World!"
    if some condition:
        text = "Hey People!"
    return HttpResponse(text)
```

All views must always return
a **response object**.

The Response Object

shop/views.py

```
...  
  
def home(request):  
    """The shop home view."""  
    response = HttpResponse()  
    if some condition:  
        response.write("Hey People!")  
    else:  
        response.write("Hello World!")  
    return response
```

The **response** object is a file-like object (a I/O stream) and has the usual methods.

The Response Object

shop/views.py

```
...  
  
def download(request):  
    """Download a pricing file."""  
    return HttpResponse(my_data, headers={  
        'Content-Type': 'application/vnd.ms-excel',  
        'Content-Disposition': 'attachment;filename="pricing.xls"'  
    })
```

The **HttpResponse** function has a **headers** argument that lets us modify the default ones.

The Response Object

shop/views.py

```
from django.http import HttpResponseRedirect, HttpResponseRedirect

def browse(request):
    """Browse the shop."""
    text = "Hello World!"
    if some condition:
        return HttpResponseRedirect("/")
    return HttpResponseRedirect(text)
```

There are a variety of **subclasses and higher-level** functions that return a response object.

The Response Object

shop/views.py

```
from django.http import HttpResponseRedirect, HttpResponseNotFound

def browse(request):
    """Browse the shop."""
    text = "Hello World!"
    if some condition:
        return HttpResponseNotFound("<html>Not found.</html>")
    return HttpResponseRedirect(text)
```

The **HttpResponseNotFound** returns a 404 error, but also requires us to define the complete HTML.

View decorators

shop/views.py

```
from django.http import HttpResponseRedirect
from django.views.decorators.http import require_http_methods

@require_http_methods(["GET", "POST"])
def my_view(request):
    if request.method == "POST":
        return HttpResponseRedirect("This was a POST request.")
    return HttpResponseRedirect("This was a GET request".)
```

The render function

shop/views.py

```
from django.shortcuts import render
```

```
def browse(request):
    """Browse the shop."""
    text = "Hello World!"
    return render(request, <template_path>,
<{context_dictionary}>)
```

The render function takes 3 parameters

The request object

The path to the template to be rendered.

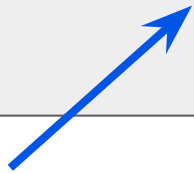
A dictionary containing context or values to be passed to the template..

The render function

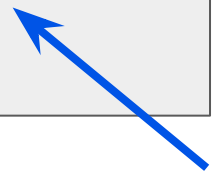
shop/views.py

```
from django.shortcuts import render

def browse(request):
    """Browse the shop."""
    text = "Hello World!"
    return render(request, "shop/hello.html", {"text": text})
```



The path to the template to be rendered.



The **text** variable will now be available in the template

Class-Based Views (CBV)

shop/views.py

```
from django.http import HttpResponse
from django.views import View

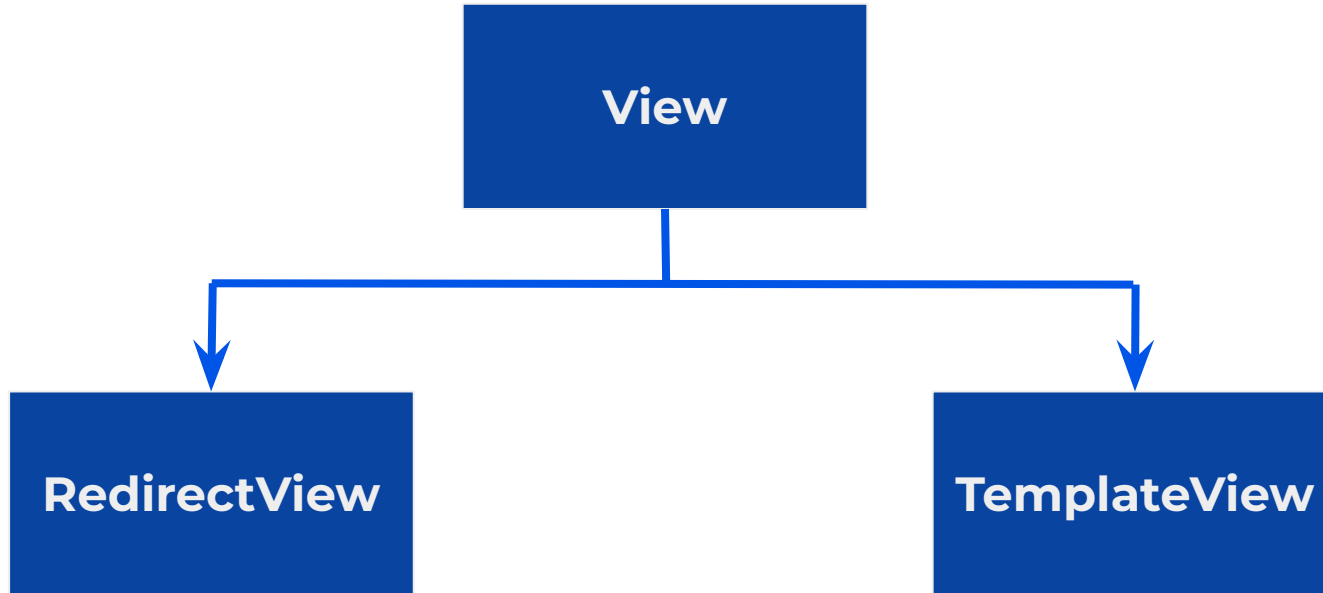
class Browse(View):
    text = "Hello World!"
    def get(self, request):
        if some condition:
            self.text = "Hey there!"
        return HttpResponse(self.text)
```

hello/urls.py

```
from shop.views import Browse

urlpatterns = [
    path('admin/',
        admin.site.urls),
    path('shop/browse/',
        Browse.as_view())
]
```

Base Views



Base Views: View

shop/views.py

```
from django.http import HttpResponseRedirect
from django.views import View
```

```
class Browse(View):
    http_method_names = ["get"]
    def setup():
        ...
    def dispatch():
        ...
    def [method_name]():
        ...
```

HTTP methods allowed for this view.

Any initialization that may be required before dispatching.

Triggers a call to the object method with the name of the allowed HTTP method.

The logic of our app will often be placed here.

File Strategy: Project vs. App

```
+ hello
  + hello
    - settings.py
    - urls.py
    - views.py
+ common
  - views.py
+ shop
  - views.py
- manage.py
- views.py
```

Some views can be easily identified with an app (shop list, item detail,...), and should be written in the app's directory.

Some views may not be so easy to place (contact view, about us view,...) and may not make sense to abstract each of them as apps. Where can those views be defined?

In the project settings directory?

In a new directory for common elements?

In an app directory?

In the project root directory?

We learned ...

- That views are functions that get executed as a result of an HTTP request and return a response object.
- That Django's request object has a variety of properties to access the data transferred in the request.
- That Django's response object is a file-like type.
- That we can use class-based views and that there are built-in class-based views that will simplify some of our code.
- How to organize our views in different files in the directory tree.



THANK YOU

Contact Details
DCI Digital Career Institute gGmbH