URL Patterns in Django

# 1. What is a URL Pattern in Django?

A URL pattern in Django maps a URL path to a view function or class.  
Defined in urls.py using the path() function.  
Each project has a main urls.py, and each app can have its own urls.py (best practice).

# 2. path() Function

The path() function is used to map URLs to views.  
Syntax:

path(route, view, kwargs=None, name=None)

Arguments:  
- route (str): The URL pattern string (e.g., 'home/', 'about/'). Can also capture dynamic parameters.  
- view (function or class): The view that should handle the URL.  
- kwargs (optional): Dictionary of extra arguments to pass to the view.  
- name (optional): Used to reference the URL by name in templates or reverse().

Example:

from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('', views.home, name='home'), # Root URL  
 path('about/', views.about, name='about'), # Simple URL  
 path('contact/', views.contact, {'via': 'web'}, name='contact'), # kwargs  
]

# 3. Dynamic URLs (Arguments in Route)

You can capture parts of the URL and pass them as arguments to the view.

urlpatterns = [  
 path('article/<int:id>/', views.article\_detail, name='article-detail'),  
 path('user/<str:username>/', views.user\_profile, name='user-profile'),  
]

How it works in views.py:

from django.http import HttpResponse  
  
def article\_detail(request, id):  
 return HttpResponse(f"Article ID: {id}")  
  
def user\_profile(request, username):  
 return HttpResponse(f"Hello, {username}")

Path Converters:  
- <int:var> → Integer  
- <str:var> → String (no slashes)  
- <slug:var> → Slug text (letters, numbers, hyphens, underscores)  
- <uuid:var> → UUID format  
- <path:var> → String including slashes

# 4. URL Names and reverse()

Using the name argument allows referencing the URL in templates or code.

path('about/', views.about, name='about')

In Template:

<a href="{% url 'about' %}">About Us</a>

In Python code:

from django.urls import reverse  
reverse('about') # Returns "/about/"

# 5. Multiple urls.py Files Using include()

Best practice is to create a separate urls.py for each app and include it in the project-level urls.py.

Example Project Structure:

myproject/  
 myproject/  
 urls.py <-- Main project URLs  
 blog/  
 urls.py <-- App-level URLs  
 shop/  
 urls.py

App-level urls.py (blog/urls.py):

from django.urls import path  
from . import views  
  
urlpatterns = [  
 path('', views.blog\_home, name='blog-home'),  
 path('post/<int:id>/', views.blog\_post, name='blog-post'),  
]

Main urls.py (myproject/urls.py):

from django.contrib import admin  
from django.urls import path, include  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('blog/', include('blog.urls')), # Include blog app URLs  
 path('shop/', include('shop.urls')), # Include shop app URLs  
]

# 6. Namespaces (Optional but Useful)

If multiple apps have URLs with the same name, use namespaces.

app\_name = 'blog'  
  
urlpatterns = [  
 path('', views.blog\_home, name='home'),  
]

Main urls.py:

urlpatterns = [  
 path('blog/', include('blog.urls', namespace='blog')),  
]

Usage in template:

<a href="{% url 'blog:home' %}">Blog Home</a>

# 7. Summary

- path() is used for URL mapping: route, view, kwargs, name.  
- Dynamic URLs use converters (<int:id>, <str:name>, etc.).  
- Use name to easily reference URLs in templates and code.  
- Best practice:  
 \* Create urls.py for each app.  
 \* Include them in the main project urls.py using include().  
- Use namespaces when you have multiple apps with similar URL names.