Module – 2

Learning Objectives

- Creating a new blogs app.
- Making its entry in settings.py file.
- Displaying "Welcome to my Blog" message on browser (HttpResponse()).
- Understanding significance of URL patterns and urls.py file.

Creating a new app

Now that your environment – a "project" – is set up, you're set to start doing work.

Each application you write in Django consists of a Python package that follows a certain convention. Django comes with a utility that automatically generates the basic directory structure of an app, so you can focus on writing code rather than creating directories.

Django uses the concept of projects and apps to keep code clean and readable. A single Django project contains one or more apps within it that all work together to power a web application. This is why the command for a new Django project is startproject! For example, a real-world Django e-commerce site might have one app for user authentication, another app for payments, and a third app to power item listing details. Each focus on an isolated piece of functionality.

Projects vs. apps

What's the difference between a project and an app? An app is a Web application that does something – e.g., a Weblog system, a database of public records or a simple poll app. A project is a collection of configuration and apps for a particular website. A project can contain multiple apps. An app can be in multiple projects.

```
D:\Django\MyBlog>python manage.py startapp blog

# Django has created a blog directory with the following files:

D:\Django\MyBlog\blog>dir

Directory of D:\Django\MyBlog\blog

03-09-2018  16:06    <DIR>          .
03-09-2018  16:06    <DIR>          ..
03-09-2018  16:06                66 admin.py
03-09-2018  16:06                88 apps.py
03-09-2018  16:06    <DIR>          migrations
03-09-2018  16:06                60 models.py
03-09-2018  16:06                63 tests.py
03-09-2018  16:06                66 views.py
03-09-2018  16:06                 0 __init__.py
```

Let's review what each new blog app file does:

- admin.py is a configuration file for the built-in Django Admin app
- apps.py is a configuration file for the app itself
- migrations/ keeps track of any changes to our models.py file so our database and models.py stay in sync
- models.py is where we define our database models, which Django automatically translates into database tables
- tests.py is for our app-specific tests
- views.py is where we handle the request/response logic for our web app

Even though our new app exists within the Django project, Django doesn't "know" about it until we explicitly add it. We do that in the file MyBlog/settings.py. We need to find INSTALLED_APPS and add a line containing 'blog', just above. So, the final product should look like this:

MyBlog/settings.py

```
# Application definition

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

Views and URLConfs

In Django, **Views** determine what content is displayed on a given page while **URLConfs** determine where that content is going. When a user requests a specific page, like the homepage, the URLConf uses a regular expression to map that request to the appropriate view function which then returns the correct data. In other words, our view will output the text "Hello, World" while our url will ensure that when the user visits the homepage they are redirected to the correct view.

MyBlog/urls.py

```
from django.contrib import admin
from django.urls import path
from blog import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.homePageView() , name = "home")
]
```

blog/views.py

```
from django.http import HttpResponse

def homePageView(request):
    return HttpResponse("Welcome to my Blog!!!")
```