Django起源

Django 最初在 2003 年到 2005 年間，由負責創建和維護報紙網站的網絡團隊開發。在創建了許多網站後，團隊開始考慮、並重用許多常見的代碼和設計模式。這個共同的代碼，演變一個通用的網絡開發框架，2005 年 7 月，被開源為 “Django” 項目。

Django 不斷發展茁壯 — 從 2008 年 9 月的第一個里程碑版本（1.0），到最近發布的（2.0）-（2018）版本。每個版本都添加了新功能，和錯誤修復，從支持新類型的數據庫，模版引擎和緩存，到添加 “通用”的 View function和classes（這減少了開發人員在一些編程任務必須編寫的代碼量）。

Django的”無限制”用途

Web框架通常將自己稱為“特定”或“無限制”。特定框架是對處理任何特定任務的“正確方法”有意的框架。他們經常支持特定領域的快速發展（解決特定類型的問題），因為正確的做法是通常被很好地理解和記錄在案。然而，他們在解決其主要領域之外的問題時可能不那麼靈活，並且傾向於為可以使用哪些組件和方法提供較少的選擇。相比之下，無限制的框架對於將組件黏合在一起以實現目標或甚至應該使用哪些组件的最佳方式的限制较少。它們使開發人員更容易使用最合適的工具來完成特定任務，儘管您需要自己查找這些组件。

使用Django的優點

**完備**

Django 遵循 “功能完備” 的理念，提供開發人員可能想要 “開箱即用” 的幾乎所有功能。因為你需要的一切，都是一個 ”產品“ 的一部分，它們都可以無縫結合在一起，遵循一致性設計原則，並且具有廣泛、和[最新的文檔](https://docs.djangoproject.com/en/2.0/)。

**通用**

Django 可以（並已經）用於構建幾乎任何類型的網站—從內容管理系統和維基，到社交網絡和新聞網站。它可以與任何客戶端框架一起工作，並且可以提供幾乎任何格式（包括 HTML、RSS、JSON、XML等）的內容。你正在閱讀的網站就是基於 Django。在內部，儘管它為幾乎所有可能需要的功能（例如幾個流行的資料庫，模版引擎等）提供了選擇，但是如果需要，它也可以擴展到使用其他組件。

**安全**

Django 幫助開發人員，通過提供一個被設計為 “做正確的事情” 來自動保護網站的框架，來避免許多常見的安全錯誤。例如，Django 提供了一種安全的方式，來管理用戶帳號和密碼，避免了常見的錯誤，比如將 session 放在 cookie 中這種易受攻擊的做法（取而代之的是，cookies 只包含一個密鑰，實際數據存儲在數據庫中），或直接存儲密碼，而不是密碼的 hash 值。默認情況下，Django 可以防範許多漏洞，包括 SQL 注入，跨站點腳本，跨站點請求偽造，和點擊劫持。

**可擴展**

Django 使用基於組件的 “無共享” 架構 (架構的每一部分獨立於其他架構，因此可以根據需要進行替換或更改)。在不同部分之間，有明確的分隔，意味著它可以通過在任何級別添加硬件，來擴展服務：緩存服務器，數據庫服務器，或應用程序服務器。一些最繁忙的網站，已經在 Django 架構下成功地縮放了網站的規模大小，以滿足他們的需求（例如 Instagram 和 Disqus。

**可維護**

Django 代碼編寫，是遵照設計原則和模式，鼓勵創建可維護和可重複使用的代碼。特別是，它使用了不要重複自己（DRY）原則，所以沒有不必要的重複，減少了代碼的數量。 Django 還將相關功能，分組到可重用的 “應用程序” 中，並且在較低級別，將相關代碼分組或模塊（ 模型視圖控制器 [Model View Controller (MVC)](https://developer.mozilla.org/en-US/Apps/Fundamentals/Modern_web_app_architecture/MVC_architecture) 模式）。

**可移植**

Django 是用 Python 編寫的，它可以在許多平台上運行。這意味著，你不受任務特定的服務器平台的限制，並且可以在許多種類的 Linux，Windows 和 Mac OS X 上運行應用程序。此外，Django 得到許多網路託管提供商的好評，他們經常提供特定的基礎設施，和託管 Django 網站的文檔。

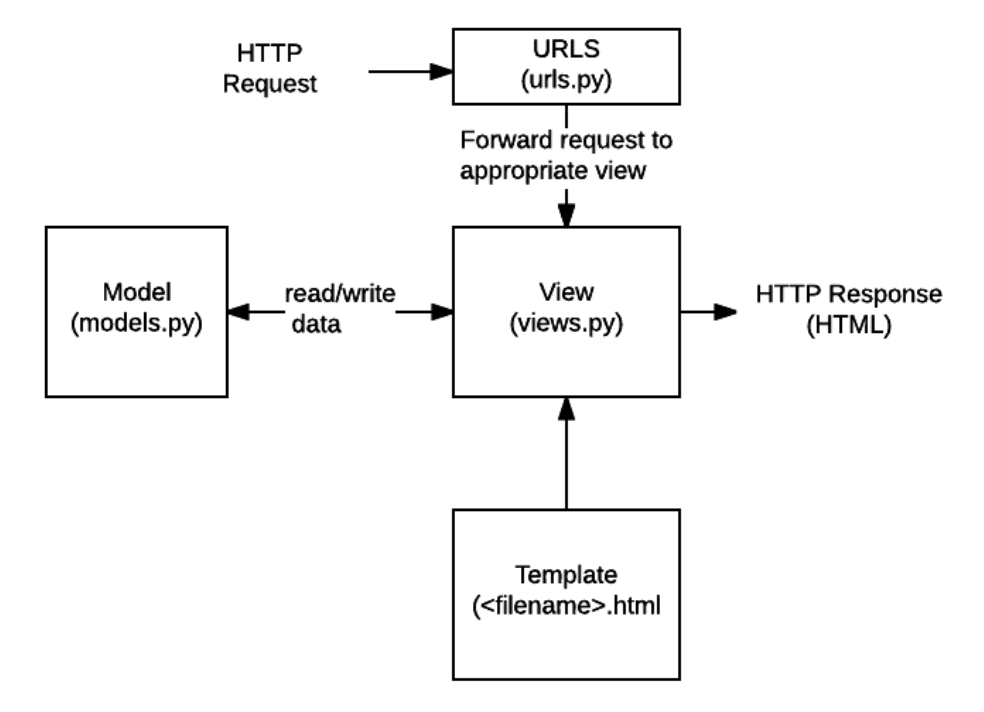
基於Django web framework開發的網站有

[Disqus](https://disqus.com/)、[Instagram](https://www.instagram.com/)、[Knight Foundation](https://www.knightfoundation.org/)、[MacArthur Foundation](https://www.macfound.org/)、[Mozilla](https://www.mozilla.org/)、[National Geographic](http://www.nationalgeographic.com/)、[Open Knowledge Foundation](https://okfn.org/)、[Pinterest](https://www.pinterest.com/)、[Open Stack](https://www.openstack.org/)

可見其現在為流行的一個framework

Django 核心

下圖為Django web framwork 在處理HTTP request 時的流程



Django 網絡應用程序通常將處理每個步驟的代碼分組到單獨的文件中:

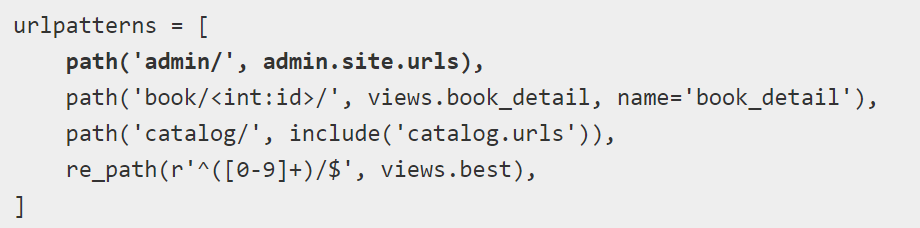
* **URLs:**雖然可以通過單個功能來處理來自每個URL的請求，但是編寫單獨的views函數來處理每個資源是更加可維護的。URL映射器用於根據請求URL將HTTP請求重定向到相應的view。
* **View:** View是一個用來處理HTTP request的函數，它接收HTTP request並返回HTTP response。View通過訪問Models滿足request所需的data，並將response的格式交給Templates處理。
* **Models:** Models是定義application data structure的Python檔案，並提供在database中管理（添加，修改，刪除）和查詢記錄的機制。
* **Templates:** Templates是定義文件（例如HTML頁面）的structure或layout的text file。一個View可以使用HTML Templates，從Database抓取data填充Templates動態地創建一個HTML頁面。

此四個區塊構成了Django的四大核心，其中的Models、Templates、View三個區塊也被稱作為MTV架構。

以下將逐一介紹Django四個核心之使用方法

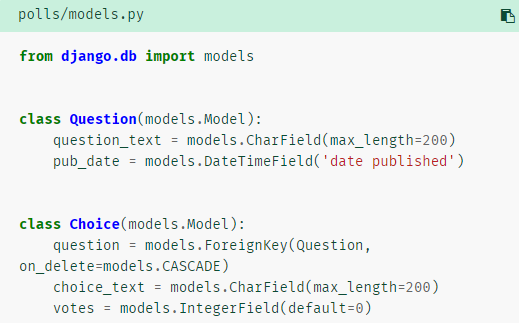
將請求發送到正確的View(urls.py)

URL映射器通常存儲在名為urls.py的文件中。在下面的示例中，mapper（urlpatterns）定義了特定URL 模式和相應View函數之間的映射列表。如果接收到具有與指定模式匹配的URL（例如r'^$'）的HTTP請求，則將調用相關聯的視圖功能（上例會呼叫views.index）並傳遞請求。



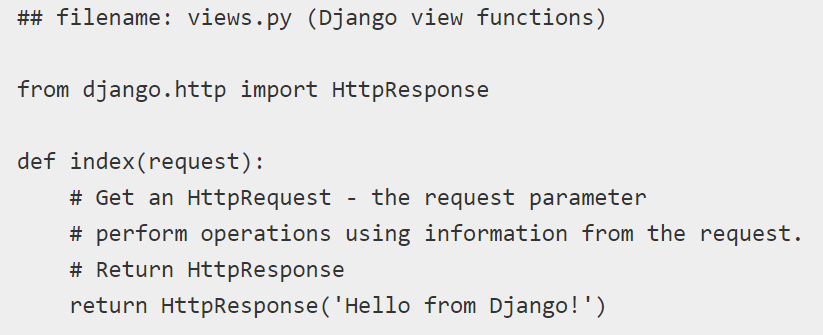
定義Database模型(models.py)

Django Web應用程序，通過被稱為模型的Python檔案，來管理和查詢數據。模型定義存儲數據的結構，包括字段類型 以及字段可能的最大值，默認值，選擇列表選項，表單的標籤文本等。模型的定義與底層數據庫無關-您可以任選其中一個資料庫，作為項目設置的一部分。一旦您選擇了要使用的資料庫，您就不需要直接與之交談，只需編寫模型結構和其他代碼，Django可以處理與資料庫通信的所有辛苦的工作。

下圖定義了兩個models，分別為Question和Choice，Question中設置兩個欄位:question\_text、和pub\_date(發布時間)，Choice中則設置三個欄位:question(此choice對應到的question)、choice\_text、votes(幾個人選了)，且分別為這些欄位設丁了一些屬性。

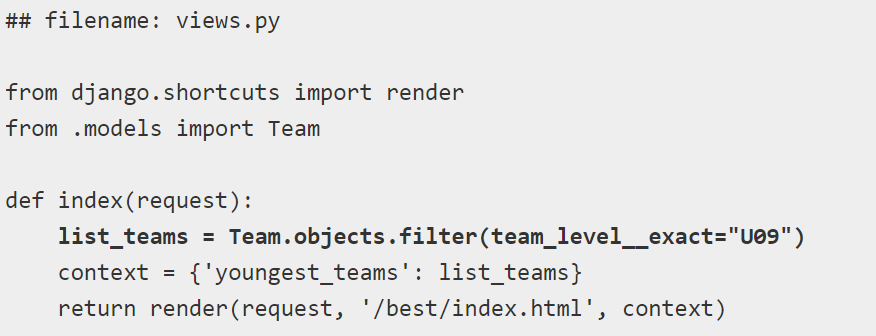
處理HTTP request(views.py)

View是Web應用程序的核心，從Web客戶端接收HTTP請求並返回HTTP響應。在兩者之間，他們編制框架的其他資源來訪問數據庫，渲染模板等。下面的例子顯示了一個最小的View function index()，這可以通過我們的URL映射器在上一節中調用。像所有View function一樣，它接收一個HttpRequest對像作為參數（request）並返回一個HttpResponse對象。在以下的例子，我們對請求不做任何事情，我們的響應只是return一個字符串。



查詢數據(views.py)

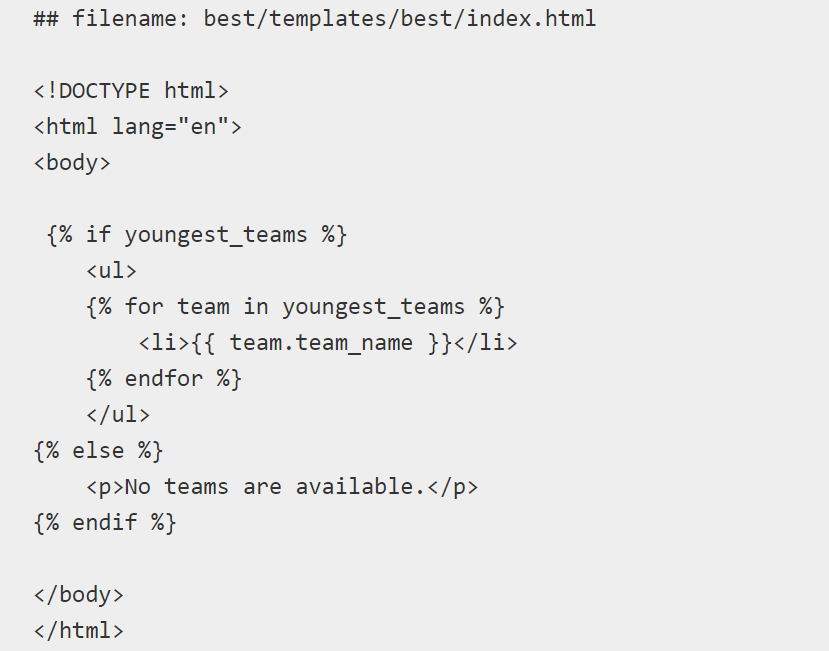
代碼片段顯示了一個視圖函數（資源處理程序），用於顯示我們所有的**U09**團隊。**粗體**顯示如何使用模型查詢API，過濾所有記錄，其中該 **team\_level**字段，具有正確的文本“ **U09** ”（請注意，該條件如何filter()作為參數傳遞給該函數，該字段名稱和匹配類型由雙下劃線： **team\_level\_\_exact**）



此功能使用**render** ()功能創建**HttpResponse**發送回瀏覽器的功能。這個函數是一個快捷方式；它通過組合指定的HTML模版和一些數據來插入模版（在名為“ **content** ”的變量中提供）來創建一個**HTML**文件。

呈現數據(HTML Templates)(此例為index.html)

模板系統允許您使用佔位符指定輸出文檔的結構，以便在生成頁面時填充數據。模板通常用於創建HTML，但也可以創建其他類型的文檔。 Django支持其本機模板系統，和另一個流行的Python庫，名為 Jinja2（如果需要，它也可以支持其他系統）。

代碼片段，顯示了上一View.py例子中render（）函數調用的HTML模板的外觀。這個模板的編寫假設它在渲染時可以訪問名為youngest\_teams的列表變量（包含在上面render（）函數中的上下文變量context中）。在HTML框架內部，我們有一個表達式，首先檢查youngest\_teams變量是否存在，然後在for循環中迭代它。在每次迭代中，模板在{{htmlelement（“li”）}}元素中顯示每個團隊的team\_name值。

文件參考

<https://developer.mozilla.org/zh-TW/docs/Learn/Server-side/Django>

(MDN web docs Django 網站框架(Python))

<https://www.djangoproject.com/> (Django 官網)