Django – Part I

2015-05-25

SPARCS 11 undead

Greatly Inspired by SPARCS 10 hodduc

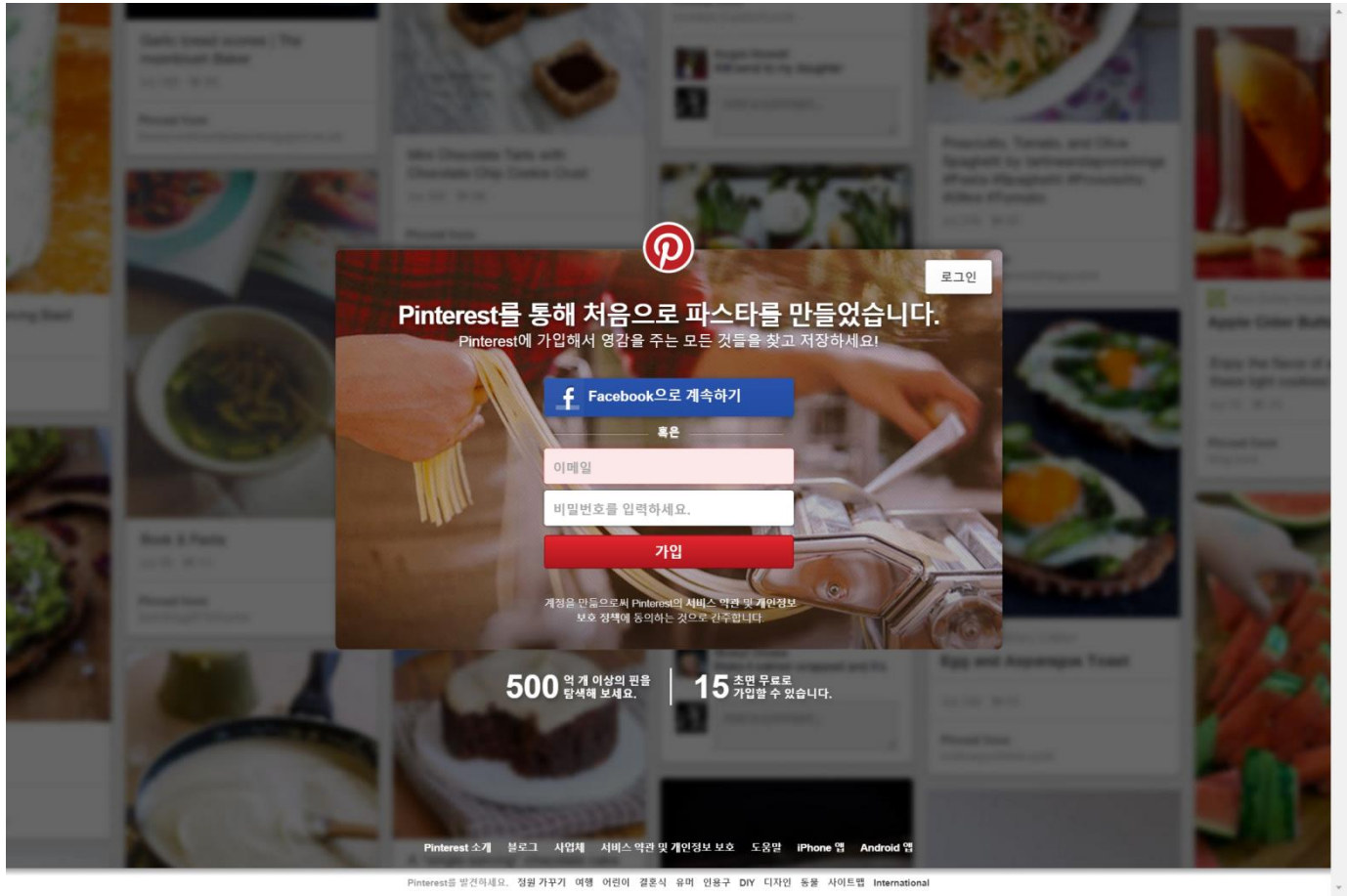
중간중간 GitHub 링크가 있으니 코드가 필요하면 참고하세요!

Django

The web framework for perfectionists with deadlines.

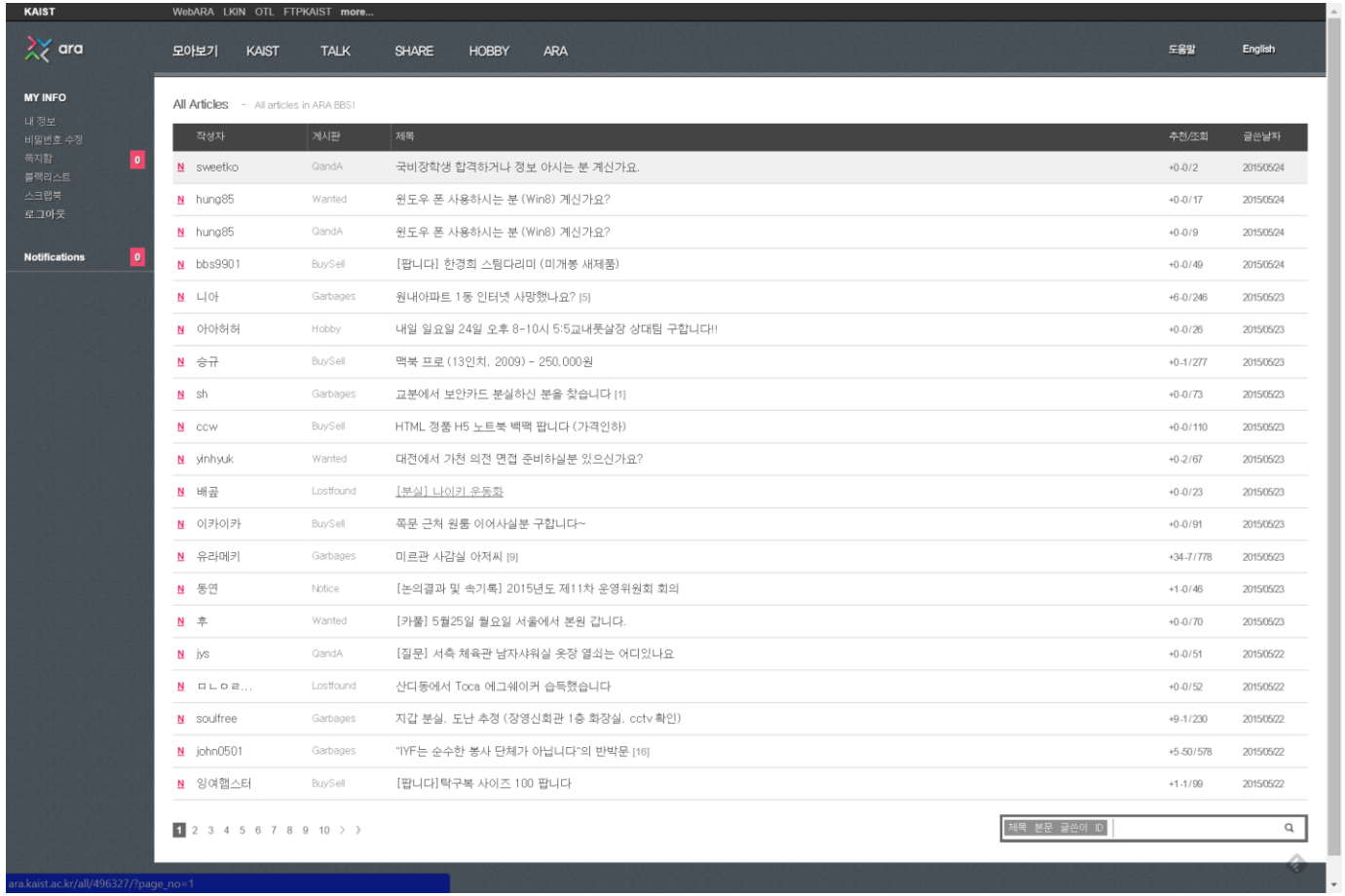
Django makes it easier to build better Web apps more quickly and with less code.

Websites Built with Django

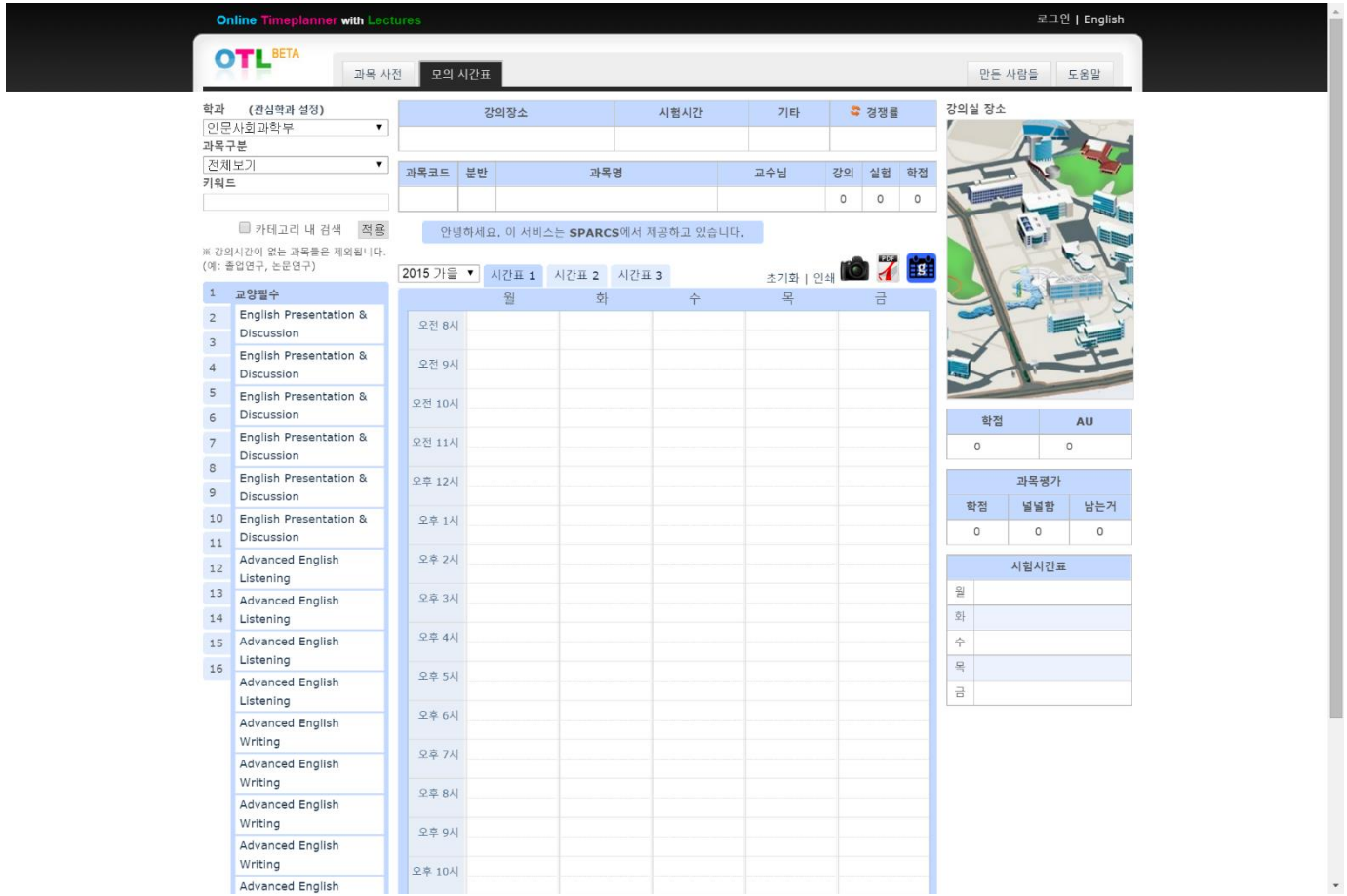


Websites Built with Django

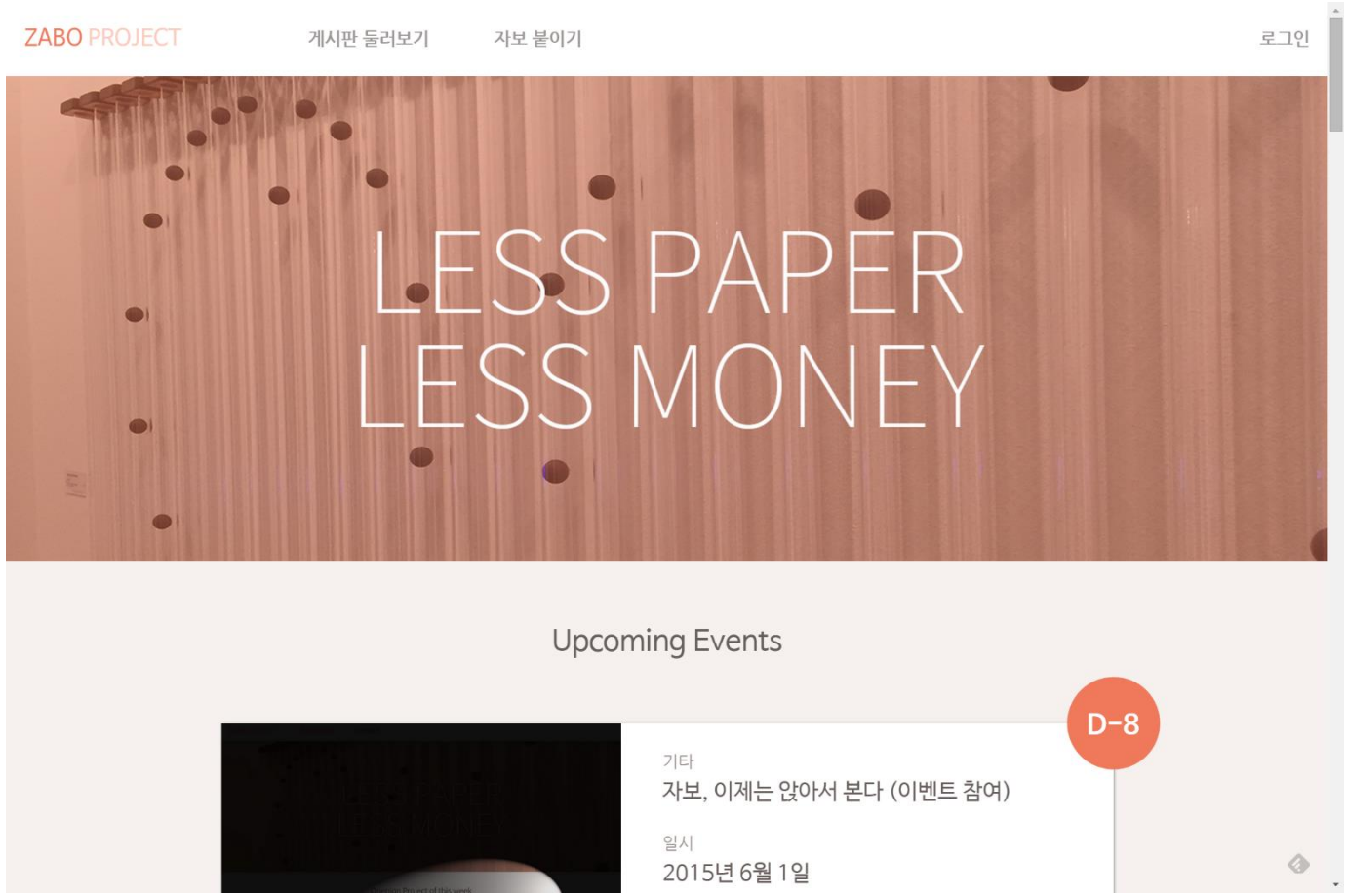
Websites Built with Django



Websites Built with Django



Websites Built with Django



Websites Built with Django

HotFrameworks | Top Frameworks | Rankings | Languages | FAQ

Rankings

Framework	Github Score	Stack Overflow Score	Overall Score
ASP.NET		100	100
Ruby on Rails	95	98	96
AngularJS	100	92	96
ASP.NET MVC		93	93
Django	89	92	90
Meteor	95	75	85
Express	92	77	84
Laravel	90	79	84
CodeIgniter		84	84
Spring	79	88	83
Ember.js	89	77	83
Symfony	85	82	83
JSF		81	81
Flask	89	71	80

Django as a Web Framework

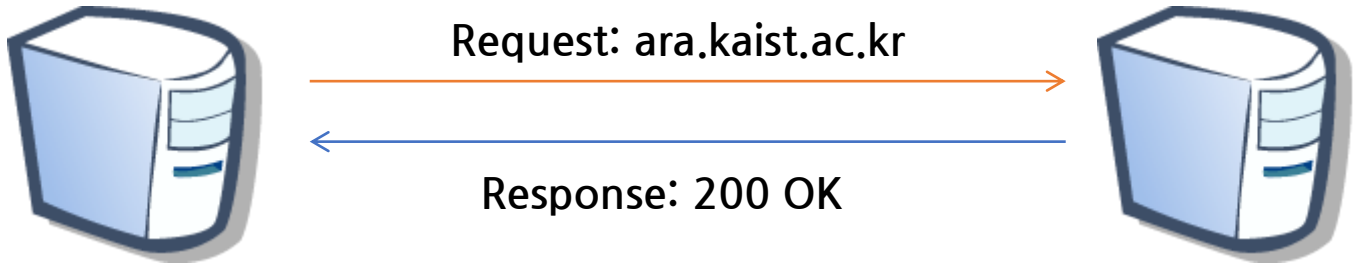
- 웹이란 무엇일까?

Django as a Web Framework

- 웹이란 무엇일까?
  - HTTP request와 HTTP response의 집합

▼ **Request Headers** view parsed

```
GET / HTTP/1.1
Host: ara.kaist.ac.kr
Connection: keep-alive
Cache-Control: max-age=0
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.202
Safari/535.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,sdch
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4
Accept-Charset: windows-949,utf-8;q=0.7,*;q=0.3
Cookie: sessionid=e839651b5b2a24ed2ad92a44fedb8c34
If-Modified-Since: Sat, 08 Oct 2011 21:14:22 GMT
```

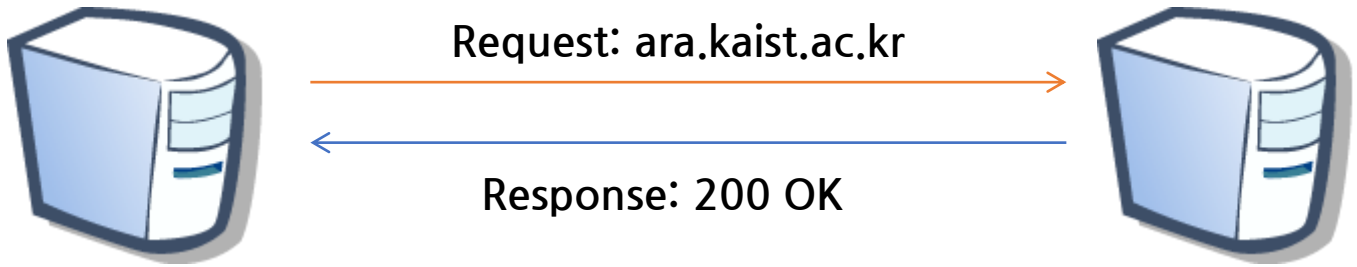


▼ **Response Headers** view parsed

```
HTTP/1.1 200 OK
Server: nginx/1.1.0
Date: Sat, 08 Oct 2011 21:14:24 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Content-Language: en
Expires: Sat, 08 Oct 2011 21:14:54 GMT
Vary: Accept-Language, Cookie
Last-Modified: Sat, 08 Oct 2011 21:14:24 GMT
Cache-Control: max-age=30
Set-Cookie: sessionid=e839651b5b2a24ed2ad92a44fedb8c34; Path=/
Content-Encoding: gzip
```

▼ **Request Headers** view parsed

```
GET / HTTP/1.1
Host: ara.kaist.ac.kr
Connection: keep-alive
Cache-Control: max-age=0
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/535.1 (KHTML, like Gecko) Chrome/14.0.835.202
Safari/535.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip,deflate,sdch
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4
Accept-Charset: windows-949,utf-8;q=0.7,*;q=0.3
Cookie: sessionid=e839651b5b2a24ed2ad92a44fedb8c34
If-Modified-Since: Sat, 08 Oct 2011 21:14:22 GMT
```



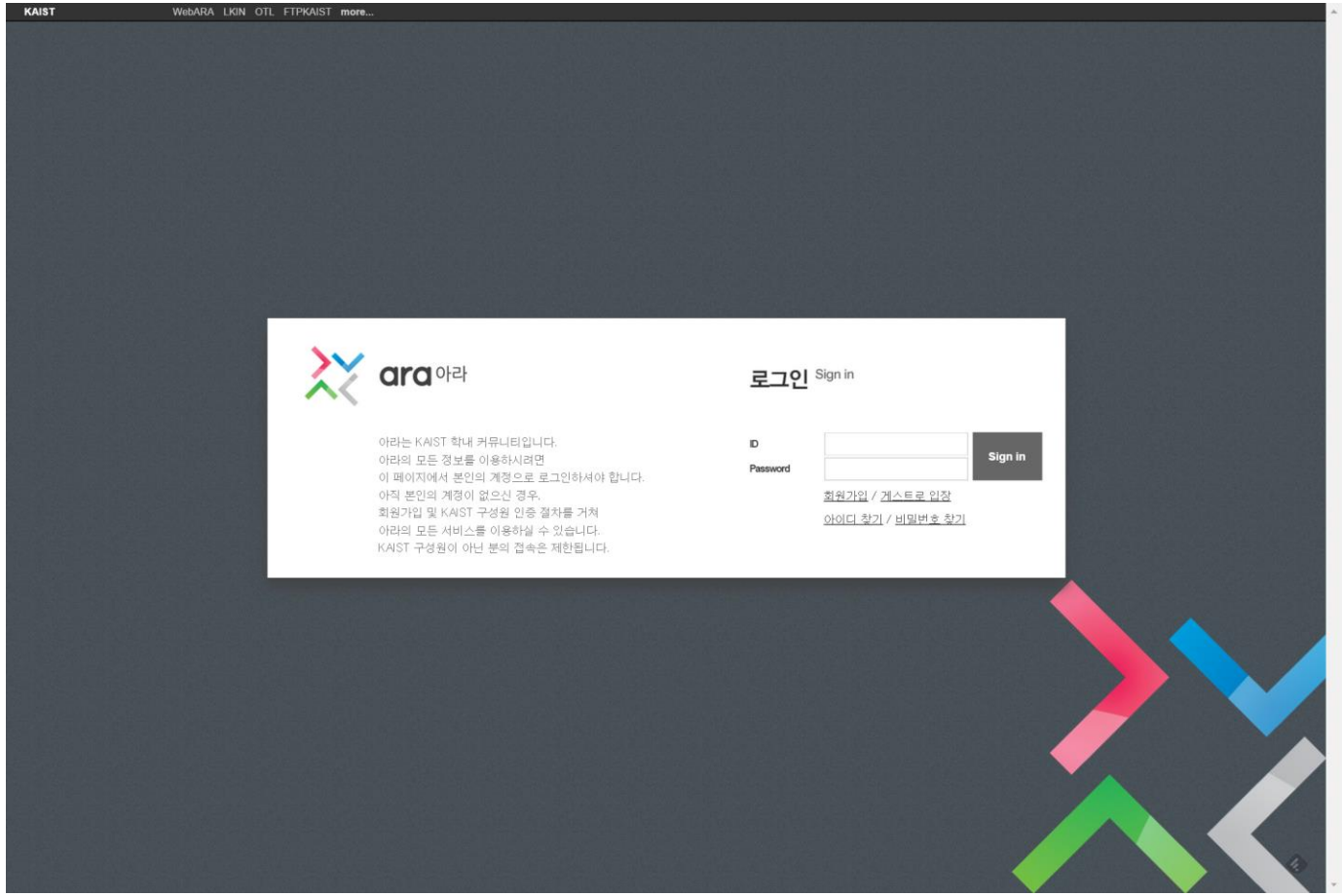

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="ko" xml:lang="ko">
    <head>
        <title>아라 - 로그인</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <meta name="description" content="KAIST 학내 커뮤니티, 아라">
        <link rel="stylesheet" href="/media/style_new/layout.css" type="text/css" media="screen" />
        <link rel="stylesheet" href="/media/style_new/stealBlue.css" type="text/css" media="screen" />
        <link rel="shortcut icon" href="/media/image/favicon.ico" type="image/x-icon" />
        <!--<link rel="stylesheet" href="/media/style_new/print.css" type="text/css" media="print" />-->
        <script type="text/javascript" src="/media/thirdparty/jquery-latest.min.js"></script>
        <script type="text/javascript" src="/media/js/layout.js"></script>
        <!--[if (gte IE 5.5)&(lte IE 8)]>
            <script type="text/javascript" src="/media/js/ie-css3.js"></script>
        <![endif]-->
    </head>

    <body id="noMenu" onload="setWidth();" onresize="setWidth();">
```

Know What You Know

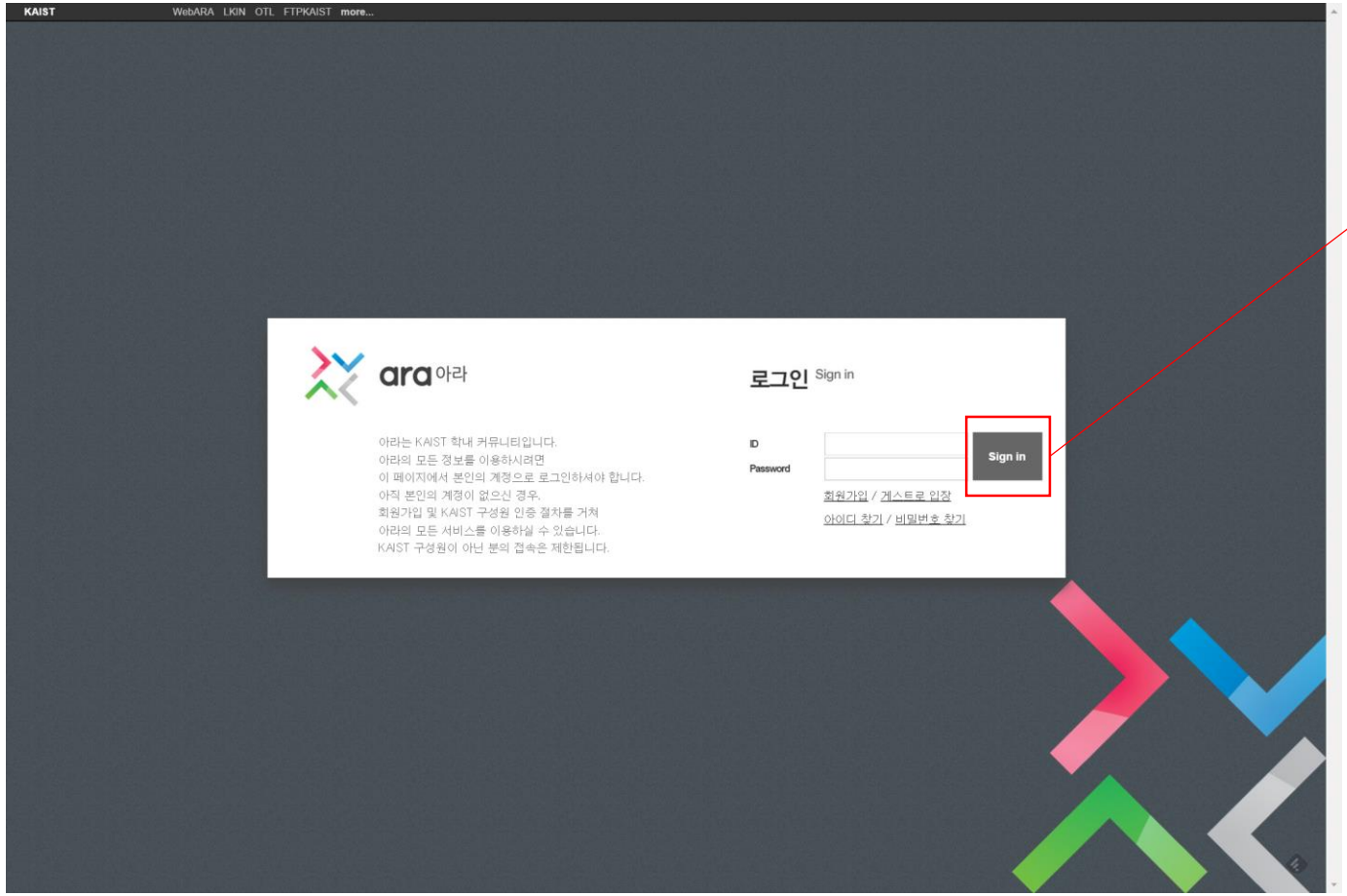
Front end programming

- Web Application
  - HTML5
  - CSS3
  - JavaScript
    - jQuery
  - Bootstrap (a ready-to-serve HTML5/CSS3/JavaScript)
- Mobile Application
  - Android
  - iOS

User Scenario



User Scenario



User Clicks.

What then?

Know What You Don’t Know

Back end programming

- Server
  - Processing incoming requests (from user)
  - Return response (to user)

- Database
  - Save user interactions
  - Logging

Know What You Don’t Know

Back end programming

- Server
  - **Processing incoming requests (from user)**
  - **Return response (to user)**

- Database
  - Save user interactions
  - Logging

Give an Output for an Input!

How?

```
#include <iostream>
using namespace std;
int main()
{
	cin >> input;
	(do something…)
	cout << output;

	return 0;
}
```

How?

- CGI

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   char *data;
   long m,n;
   printf("%s%c%c\n", "Content-Type:text/html;charset=iso-8859-1",13,10);
   printf("<TITLE>Multiplication results</TITLE>\n");
   printf("<H3>Multiplication results</H3>\n");
   data = getenv("QUERY_STRING");
   if(data == NULL)
     printf("<P>Error! Error in passing data from form to script.");
   else if(sscanf(data,"m=%ld&n=%ld",&m,&n)!=2)
     printf("<P>Error! Invalid data. Data must be numeric.");
   else
     printf("<P>The product of %ld and %ld is %ld.",m,n,m*n);
   return 0;
}
```

Don’t Reinvent the Wheel

있는 건 그냥 가져다 씁시다.
또또칸 애들이 이미 잘 만들어뒀으니까요.

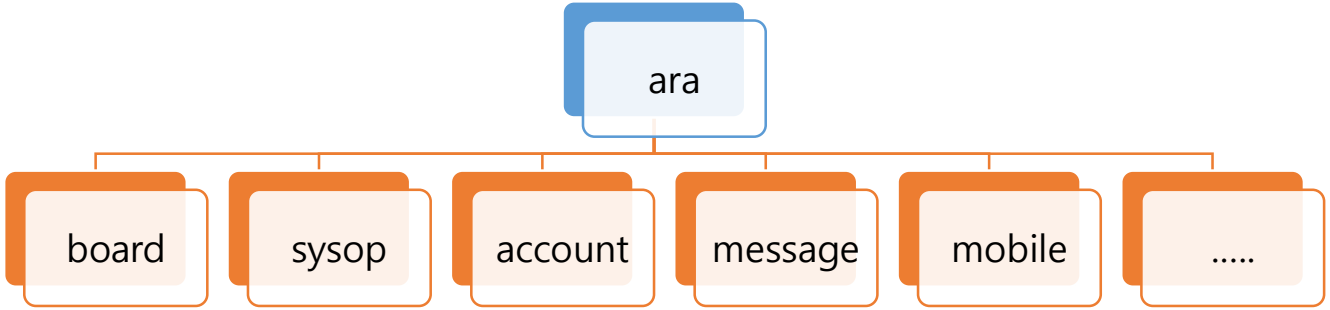
Let the Django Begin

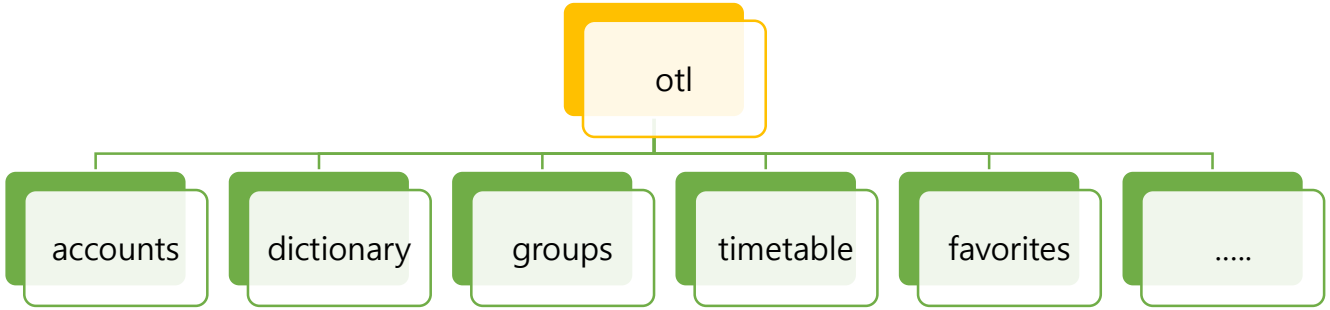
Basic Structure

Project and Applications

Basic Structure

Django 웹 어플리케이션은 project와 app으로 구성되어 있다.


ara
board
sysop
account
message
mobile
.....


otl
accounts
dictionary
groups
timetable
favorites
.....

Practice: Project ‘tutorial’

With ‘helloworld’, ‘intro’ apps

Before You Start Any Python Project…

```
~/ $ virtualenv env
~/ $ source env/bin/activate

# If a requirements.txt exists
~/ $ pip install -r requirements.txt
```

```
(env) ~/ $
```

Before You Start Any Python Project…

```
~/ $ virtualenv env
~/ $ source env/bin/activate

# If a requirements.txt exists
~/ $ pip install -r requirements.txt
```

```
(env) ~/ $
(env) ~/ $ deactivate          # When you are done
```

Start a Django Project

```
(env) ~/ $ pip install django

# Django version 1.8.2 installed as of 2015-05-24.

(env) ~/ $ django-admin startproject tutorial
(env) ~/ $ cd tutorial
(env) ~/tutorial $ ls
manage.py tutorial
```

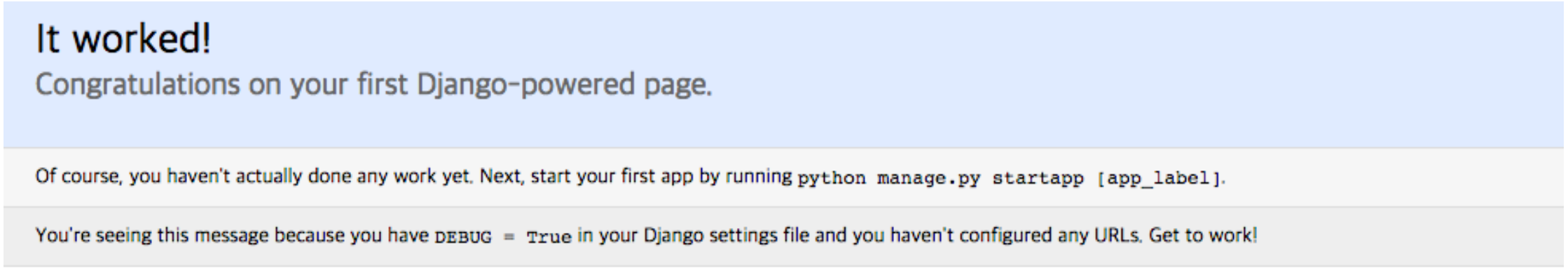
Start a Django Project

```
env/                        # Python virtual environment directory
  ..
tutorial/                   # Django project directory
  manage.py                 # Django project manage utility
  tutorial/                 # Django default package directory
    __init__.py             # Empty file: python package identifier
    settings.py             # File for Django settings
    urls.py                 # File for URL mapping
    wsgi.py                 # WSGI entry point
```

Start a Django Project

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER


# PORT_NUMBER in range 0 ~ 65535
# 0 ~ 9999 preoccupied by system.
# (Which means, DON’t USE THEM!!)
```

Start a Django Project



View Code: Github.com

First App: Hello, World!

Creating a Django App

```
(env) ~/tutorial $ python manage.py startapp helloworld
(env) ~/tutorial $ ls
helloworld manage.py tutorial
(env) ~/tutorial $ ls helloworld
__init__.py  admin.py  migrations  models.py  tests.py
views.py
```

Add App to Project

```
(env) ~/tutorial $ vi tutorial/settings.py
```

```
INSTALLED_APPS = (
    …
    ‘django.contrib.staticfiles’,
    ‘helloworld’,
)
```

Writing a View File

```
(env) ~/tutorial $ vi helloworld/views.py
```

```
from django.http import HttpResponse

# Create your views here.


def helloworld(request):
    return HttpResponse(“Hello, World!”)
```

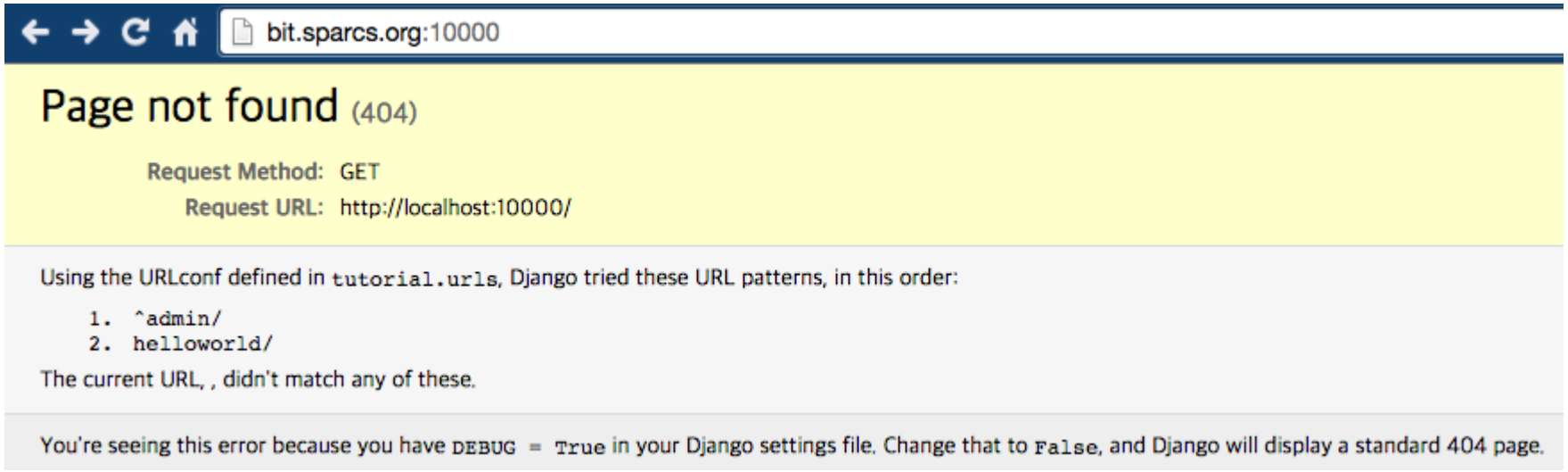
Map a URL for a View Function

```
(env) ~/tutorial $ vi tutorial/urls.py
```

```
…
urlpatterns = [
      url(r’^admin/’, include(admin.site.urls)),
      url(r’^helloworld/’, ‘helloworld.views.helloworld’),
]
```

Run Server Again

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER
```



Run Server Again

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER
```

View Code: Github.com

Understanding urls.py

Role of urls.py

- 이정표로써 urls.py
  - Request가 발생했다!
  - 여긴 어디? 난 누구?

  - urls.py: 당황하지 말고 어디어디로 가서 이 함수를 실행시켜 보세요.

Role of urls.py

```
url(r'^helloworld/', 'helloworld.views.helloworld'),
```

- "'helloworld/' 요청이 들어왔네요?
  helloworld app 안의 views.py를 열어보시고 helloworld()라는 함수를 실행시키세요!"

Application of urls.py

- 여러 앱으로 구성된 프로젝트에서 작업하고 있다면?

- /main/view/ : 메인 앱의 뷰 함수에 뷰로 가세요
- /board/list/ : 보드 앱의 뷰 함수에 리스트로 가세요
- /board/new/ : 보드 앱의 뷰 함수의 리스트로 가세..요 bb
- /board/edit/ : 보드 앱의 뷰 함수의 에딧으로 가세요 ㅠㅠㅠ
- /session/login/ : 세션 앱의 뷰 함수의...
- ... : 으앙 아몰랑 ㅠㅠ

Application of urls.py

- 불쌍한 urls.py짱을 위해 코드를 모듈화합시다.

- /main/view/ : 메인 앱이시네요? 메인 앱의 담당자한테 rr
- /board/list/ : 보드 앱이시네요? 보드 앱의 담당자한테 rr
- /board/new/ : 너도 보드 앱한테 가세요
- /board/edit/ : 너도 보드 앱한테 가시고 그 다음은 아몰랑
- /session/login/ : 세션 앱의 담당자한테 가면 걔가 알려줄거임
- /wrong_request/ : 넌 누구세요? 안사요 관심 없어요

URL Routing

```
(env) ~/tutorial $ cp tutorial/urls.py helloworld/
(env) ~/tutorial $ ls helloworld/
__init__.py  admin.py  migrations  models.py
tests.py  urls.py  views.py
```

URL Routing

```
(env) ~/tutorial $ vi tutorial/urls.py
```

```
urlpatterns = [
    url(r’^admin/’, include(admin.site.urls)),
    url(r’^helloworld/’, ‘helloworld.views.helloworld’),
]
```

URL Routing

```
(env) ~/tutorial $ vi tutorial/urls.py
```

```
urlpatterns = [
      url(r’^admin/’, include(admin.site.urls)),
      url(r’^helloworld/’, ‘helloworld.views.helloworld’),
      url(r’^helloworld/’, include(‘helloworld.urls’)),
]
```

URL Routing

```
(env) ~/tutorial $ vi helloworld/urls.py
```

```
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^helloworld/', 'helloworld.views.helloworld'),
]
```

URL Routing

```
(env) ~/tutorial $ vi helloworld/urls.py
```

```
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^helloworld/', 'helloworld.views.helloworld'),
    url(r'^$', 'helloworld.views.helloworld'),
]
```

Add Another View Function

```
(env) ~/tutorial $ vi helloworld/views.py
```

```
def helloworld(request):
    return HttpResponse(“Hello, World!”)



def introduce(request):
    return HttpResponse(“This is Shavakan, 24 years old, from Zul’jin, Azeroth.”)
```

Add Another View Function

```
(env) ~/tutorial $ vi helloworld/urls.py
```

```
urlpatterns = [
    url(r'^$', 'helloworld.views.helloworld'),
    url(r'^introduce/', 'helloworld.views.introduce'),
]
```

Add Another View Function

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER
```

View Code: Github.com

Add Another App

```
(env) ~/tutorial $ python manage.py startapp intro
(env) ~/tutorial $ vi tutorial/settings.py
```

```
INSTALLED_APPS = (
    …
    ‘django.contrib.staticfiles’,
    ‘helloworld’,
)
```

Add Another App

```
(env) ~/tutorial $ python manage.py startapp intro
(env) ~/tutorial $ vi tutorial/settings.py
```

```
INSTALLED_APPS = (
    …
    ‘django.contrib.staticfiles’,
    ‘helloworld’,
    ‘intro’,
)
```

Add Another App

```
(env) ~/tutorial $ cp tutorial/urls.py intro/
(env) ~/tutorial $ vi tutorial/urls.py
```

```
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^helloworld/', include('helloworld.urls')),
]
```

Add Another App

```
(env) ~/tutorial $ cp tutorial/urls.py intro/
(env) ~/tutorial $ vi tutorial/urls.py
```

```
urlpatterns = [
    url(r'^admin/', include(admin.site.urls)),
    url(r'^helloworld/', include('helloworld.urls')),
    url(r'^intro/', include('intro.urls')),
]
```

Add Another App

```
(env) ~/tutorial $ vi intro/views.py
```

```
# -*- coding: utf-8 -*-
from django.http import HttpResponse



def home(request):
      return HttpResponse('/intro/스팍스/(자기 학번)/(자기 이름)으로 가주세요!')
```

Add Another App

```
(env) ~/tutorial $ vi intro/urls.py
```

```
urlpatterns = [
    ~~url(r'^admin/', include(admin.site.urls)),~~
    ~~url(r'^helloworld/', include('helloworld.urls')),~~
    url(r'^$', 'intro.views.home'),
]
```

Add Another App

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER
```

/intro/스팍스/(자기 학번)/(자기 이름)으로 가주세요!

This is Shavakan, 24 years old, from Zul'jin, Azeroth.

Hello, World!

나는 자바계의 김정일이다! / K.H.L.

으아ㅏ아ㅏㅏㅏㅏㅏㅏㅏ
대포동 발사!!

/board/garbages/123456

아라 개발팀에 카와이한 디자이너님이 계시다던데 사실인가요 / 호떡

진실은 저 너머에

/board/qandA/654321

여자친구와 데이트 후기 / qwertyasdf

/board/love/1048576

매번 바뀌는 URL **규칙**

참고: 픽션입니다 ^^

Dynamic URL

- URL pattern은 정규식으로 표현한다.
  - 모르면 공부하세요... 뭔 줄 알고 필요할 때 찾아 쓸 수 있는 정도면 됨

```
…
urlpatterns = patterns(‘’,
…
  (r'^([\w \[\]\.]+)/([\d]+)/$', ‘warara.board.views.read')
…
)
```

Dynamic URL

- 만약 /garbage/3334343/ 이라면
  read(request, ‘garbage’, ‘3334343’) 함수 호출과 동일하게 동작

```
…
urlpatterns = patterns(‘’,
…
  (r'^([\w \[\]\.]+)/([\d]+)/$', ‘warara.board.views.read')
…
)
```

Write View Function for Dynamic URL

```
(env) ~/tutorial $ vi intro/views.py
```

```
# -*- coding: utf-8 -*-
from django.http import HttpResponse


…
def me(request, city, town, name):
    s = u"나는 %s %s의 %s이다!" % (city, town, name)
    return HttpResponse(s)
```

Write Dynamic URL Handler

```
(env) ~/tutorial $ vi intro/urls.py
```

```
urlpatterns = [
      url(r’^$’, ‘intro.views.home’),
      url(r'^([^/]+)/([^/]+)/([^/]+)/', 'intro.views.me'),
]
```

Run Server With Dynamic URL

```
(env) ~/tutorial $ python manage.py runserver
0.0.0.0:PORT_NUMBER
```

View Code: Github.com

Next Time: All About Template