

# Django – Part I

2015-05-25

SPARCS 11 undead

Greatly Inspired by SPARCS 10 hodduc

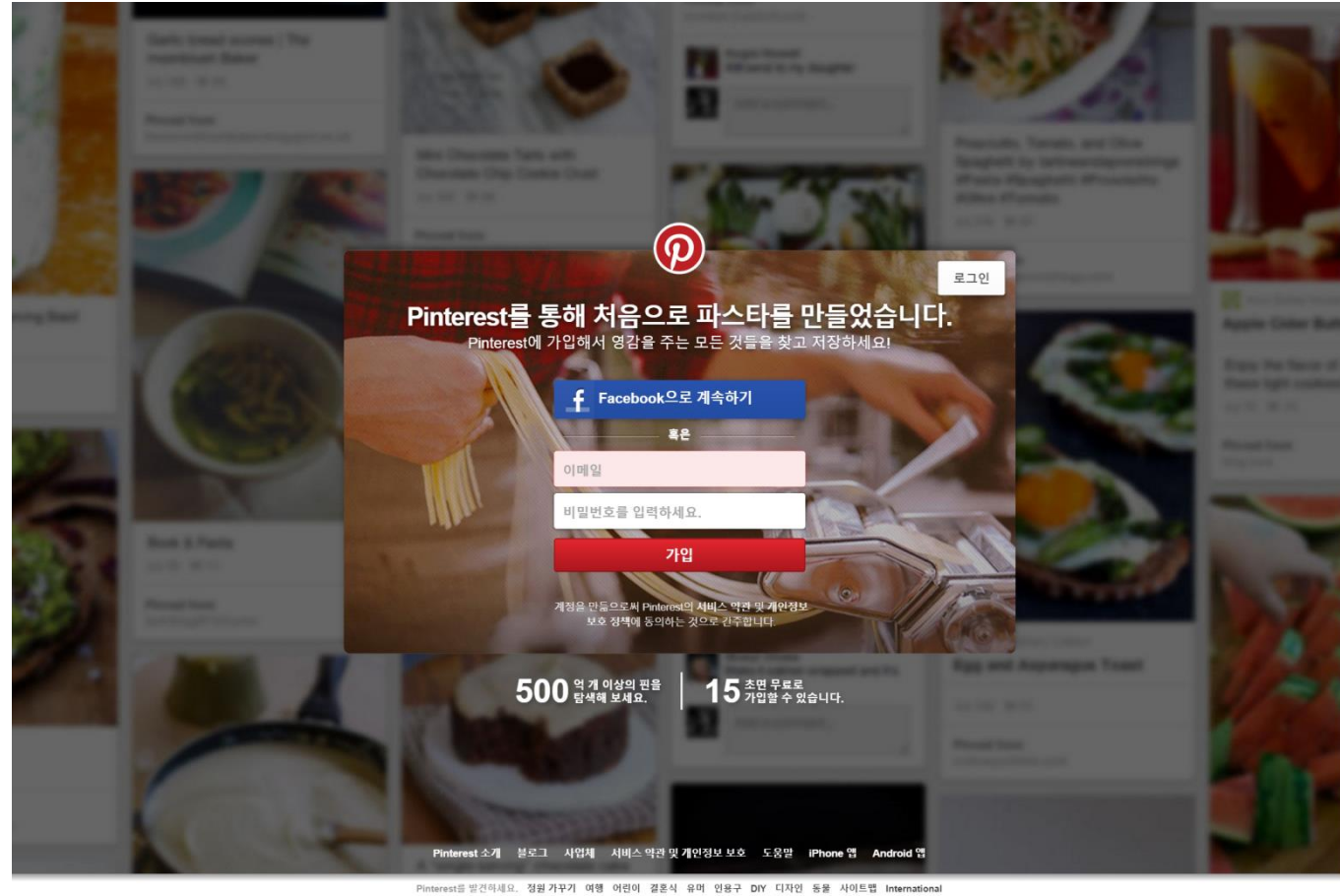
중간중간 GitHub 링크가 있으니 코드가 필요하면 참고하세요!

# 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



주변에서 일어나는 아름답고 중요한 순간을 포착해서 공유해보세요

Instagram은 친구 및 가족들과 일상을 빠르고, 멋지고, 재미있게 공유할 수 있는 앱입니다.

사진이나 동영상을 찍고 원하는 느낌의 필터를 선택하여 간단히 Instagram에 게시하세요. Facebook, Twitter, Tumblr 등에 공유할 수도 있습니다. 세상을 바라보는 새로운 방법을 경험해보세요!

참, 무료라고 말씀드렸었나요?

 App Store에서 다운로드 하기

 다운로드 하기 Google play

[소개](#) [지원](#) [블로그](#) [관련 기사](#) [API](#) [채용 정보](#) [개인 정보 보호](#) [약관](#) © 2015 INSTAGRAM

# Websites Built with Django

The screenshot displays the KAIST ARA website interface. At the top, there is a navigation bar with the KAIST logo and the 'ara' sub-brand. The main menu includes '모아보기', 'KAIST', 'TALK', 'SHARE', 'HOBBY', and 'ARA'. On the left side, there is a 'MY INFO' section with links for '내 정보', '비밀번호 수정', '쪽지함', '블랙리스트', '스크랩북', and '로그아웃'. Below this is a 'Notifications' section with a red badge indicating 9 notifications. The main content area is titled 'All Articles - All articles in ARA BBS!' and contains a table of articles. The table has columns for '작성자' (Author), '게시판' (Board), '제목' (Title), '추진/조회' (Progress/Search), and '글쓴날자' (Posted Date). The first article is by 'sweetko' in the 'QandA' board, titled '국비장학생 합격하거나 정보 아시는 분 계신가요?'. The table lists 20 articles in total. At the bottom of the page, there is a search bar and a footer with the URL 'ara.kaist.ac.kr/all/496327/?page\_no=1'.

작성자	게시판	제목	추진/조회	글쓴날자
N sweetko	QandA	국비장학생 합격하거나 정보 아시는 분 계신가요?	+0-0/2	20150524
N hung85	Wanted	윈도우 폰 사용하시는 분 (Win8) 계신가요?	+0-0/17	20150524
N hung85	QandA	윈도우 폰 사용하시는 분 (Win8) 계신가요?	+0-0/9	20150524
N bbs9901	BuySell	[팝니다] 한경희 스텝다라미 (미개봉 새제품)	+0-0/49	20150524
N 니아	Garbages	원내아파트 1동 인터넷 사망했나요? [9]	+6-0/246	20150523
N 아아허허	Hobby	내일 일요일 24일 오후 8-10시 5:5교내풋살장 상대팀 구합니다!!	+0-0/26	20150523
N 승규	BuySell	맥북 프로 (13인치, 2009) - 250,000원	+0-1/277	20150523
N sh	Garbages	교분에서 보인카드 분실하신 분을 찾습니다 [1]	+0-0/73	20150523
N ccw	BuySell	HTML 정통 H5 노트북 백팩 팝니다 (가격인하)	+0-0/110	20150523
N ynhyuk	Wanted	대전에서 가천 의전 면접 준비하실분 있으신가요?	+0-2/67	20150523
N 배굴	Lostfound	[분실] 나이키 운동화	+0-0/23	20150523
N 이카이가	BuySell	쪽문 근처 원통 이어사실분 구합니다~	+0-0/91	20150523
N 유라메키	Garbages	미르관 사감실 아저씨 [9]	+34-7/778	20150523
N 통연	Notice	[논의결과 및 속기록] 2015년도 제11차 운영위원회 회의	+1-0/46	20150523
N 후	Wanted	[카풀] 5월25일 월요일 서울에서 분원 갑니다.	+0-0/70	20150523
N jys	QandA	[질문] 서측 체육관 남자사위실 옷장 열쇠는 어디있나요	+0-0/51	20150522
N ㅁㅇㅇ...	Lostfound	산다동에서 Toca 에그셰이커 습득했습니다	+0-0/52	20150522
N soulfree	Garbages	지갑 분실, 도난 추정 (장영신회관 1층 화장실, CCTV 확인)	+9-1/230	20150522
N john0501	Garbages	"YF는 순수한 봉사 단체가 아닙니다"의 반박문 [16]	+5-50/578	20150522
N 잉여햄스터	BuySell	[팝니다]탁구복 사이즈 100 팝니다	+1-1/99	20150522

# Websites Built with Django

Online Timeplanner with Lectures 로그인 | English

**OTL BETA** 과목 사전 모의 시간표 만든 사람들 도움말

학과 (관심학과 설정)  
인문사회과학부

과목구분  
전체 보기


키워드

카테고리 내 검색 적용

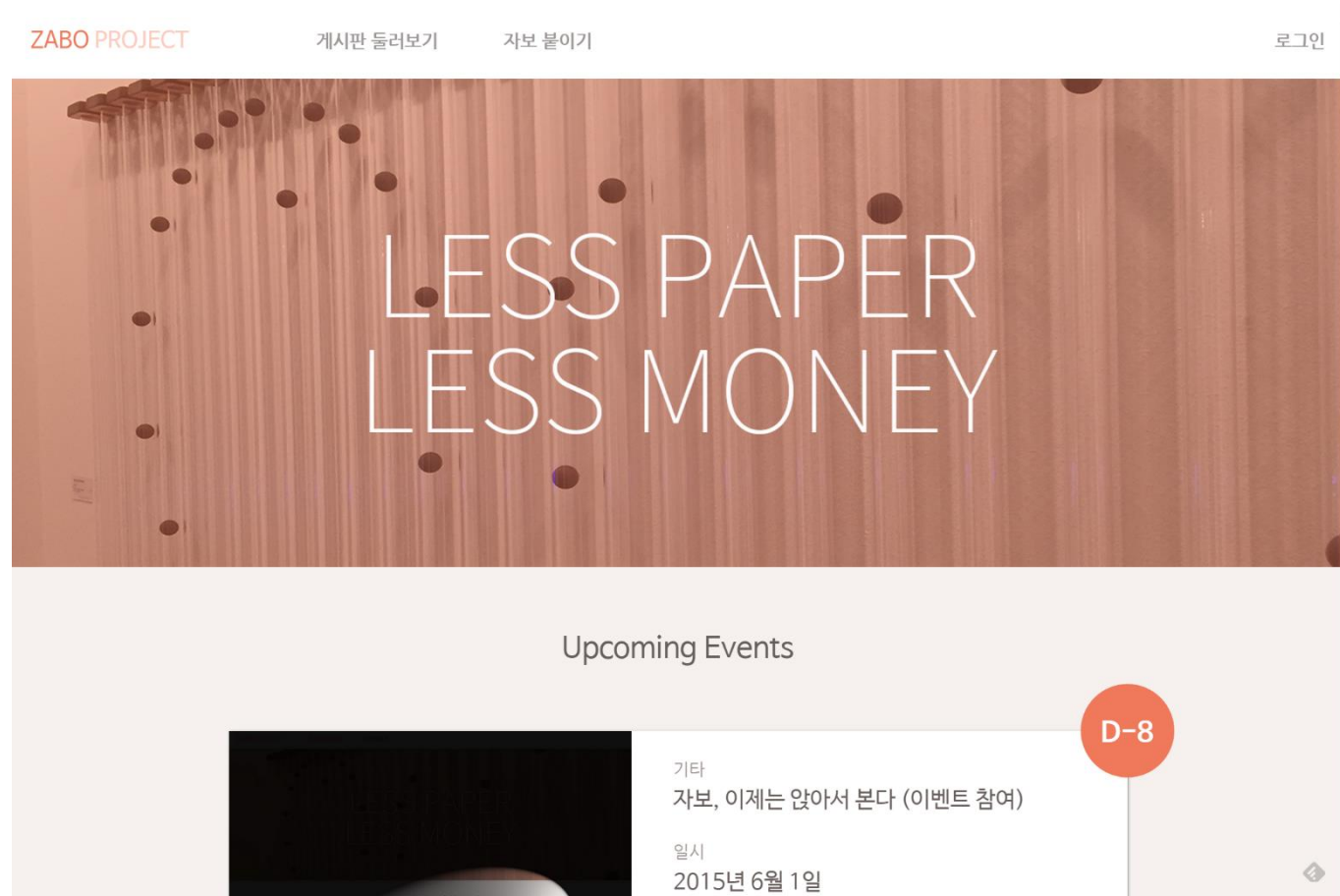
안녕하세요, 이 서비스는 SPARCS에서 제공하고 있습니다.

\* 강의시간이 없는 과목들은 제외됩니다.  
(예: 졸업연구, 논문연구)

2015 가을 시간표 1 시간표 2 시간표 3 초기화 | 인쇄

	강의장소	시험시간	기타	경쟁률	강의실 장소																																												
1 교양필수					 <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <tr> <th colspan="2">학점</th> <th colspan="2">AU</th> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <th colspan="4">과목평가</th> </tr> <tr> <th>학점</th> <th>불널함</th> <th colspan="2">남는거</th> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <th colspan="4">시험시간표</th> </tr> <tr> <td>월</td> <td></td> <td></td> <td></td> </tr> <tr> <td>화</td> <td></td> <td></td> <td></td> </tr> <tr> <td>수</td> <td></td> <td></td> <td></td> </tr> <tr> <td>목</td> <td></td> <td></td> <td></td> </tr> <tr> <td>금</td> <td></td> <td></td> <td></td> </tr> </table>	학점		AU		0	0	0	0	과목평가				학점	불널함	남는거		0	0	0	0	시험시간표				월				화				수				목				금			
학점		AU																																															
0	0	0	0																																														
과목평가																																																	
학점	불널함	남는거																																															
0	0	0	0																																														
시험시간표																																																	
월																																																	
화																																																	
수																																																	
목																																																	
금																																																	
2 English Presentation & Discussion																																																	
3 English Presentation & Discussion																																																	
4 English Presentation & Discussion																																																	
5 English Presentation & Discussion																																																	
6 English Presentation & Discussion																																																	
7 English Presentation & Discussion																																																	
8 English Presentation & Discussion																																																	
9 English Presentation & Discussion																																																	
10 English Presentation & Discussion																																																	
11 Advanced English Listening																																																	
12 Advanced English Listening																																																	
13 Advanced English Listening																																																	
14 Advanced English Listening																																																	
15 Advanced English Listening																																																	
16 Advanced English Listening																																																	

# 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

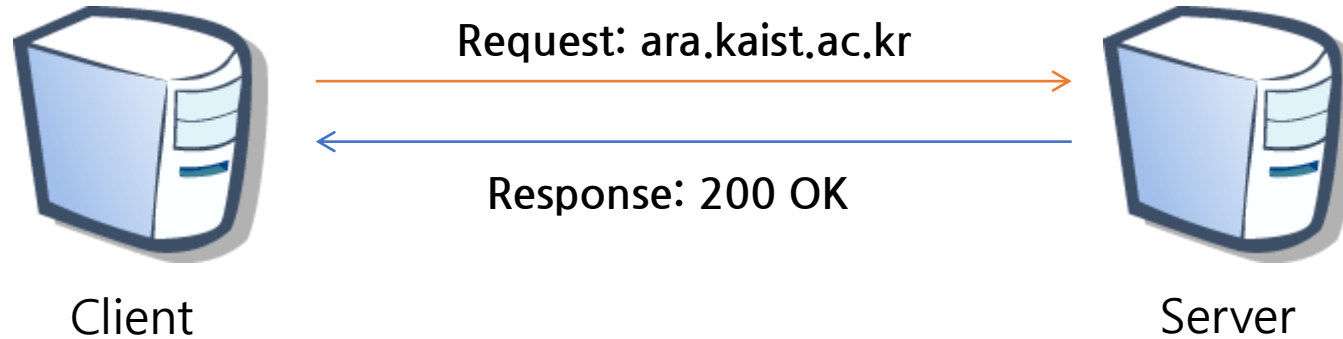
- 웹이란 무엇일까?

# 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
```



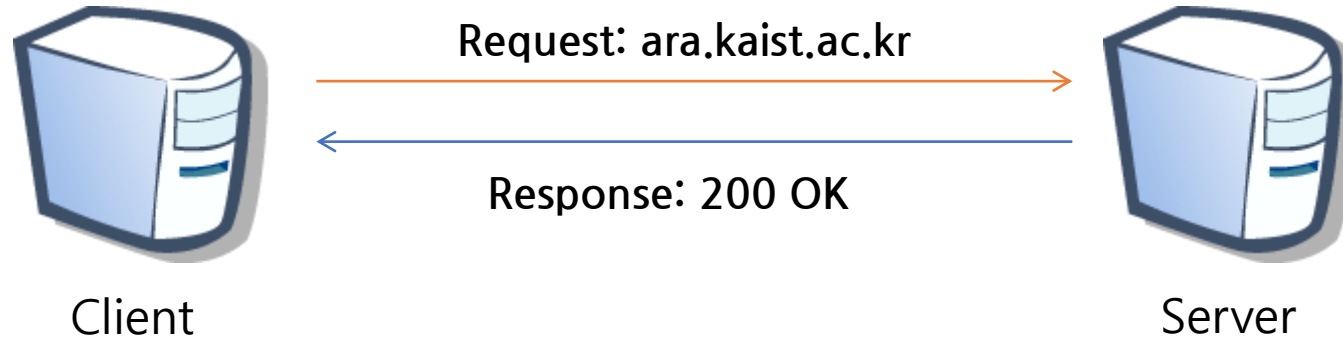
▼ **Response Headers** [view parsed](#)

```
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
```



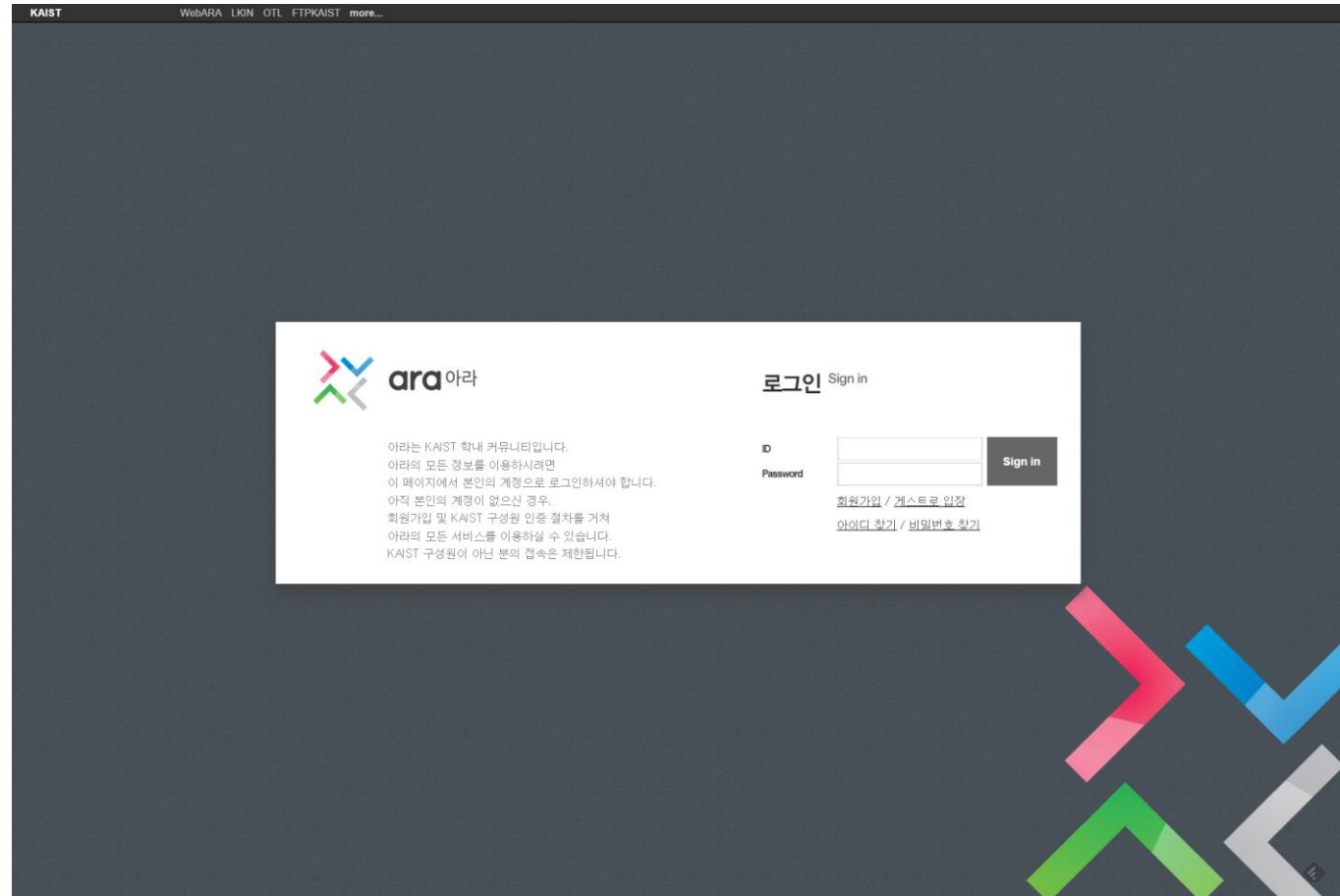
```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
2
3 <html xmlns="http://www.w3.org/1999/xhtml" lang="ko" xml:lang="ko">
4   <head>
5     <title>아라 - 로그인</title>
6     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
7     <meta name="description" content="KAIST 학내 커뮤니티, 아라">
8     <link rel="stylesheet" href="/media/style_new/layout.css" type="text/css" media="screen" />
9     <link rel="stylesheet" href="/media/style_new/stealBlue.css" type="text/css" media="screen" />
10    <link rel="shortcut icon" href="/media/image/favicon.ico" type="image/x-icon" />
11    <!--<link rel="stylesheet" href="/media/style_new/print.css" type="text/css" media="print" />-->
12    <script type="text/javascript" src="/media/thirdparty/jquery-latest.min.js"></script>
13    <script type="text/javascript" src="/media/js/layout.js"></script>
14    <!--[if (gte IE 5.5)&(lte IE 8)]>
15      <script type="text/javascript" src="/media/js/ie-css3.js"></script>
16    <![endif]-->
17  </head>
18
19  <body id="noMenu" onload="setWidth();" onresize="setWidth();">
```

# Know What You Know

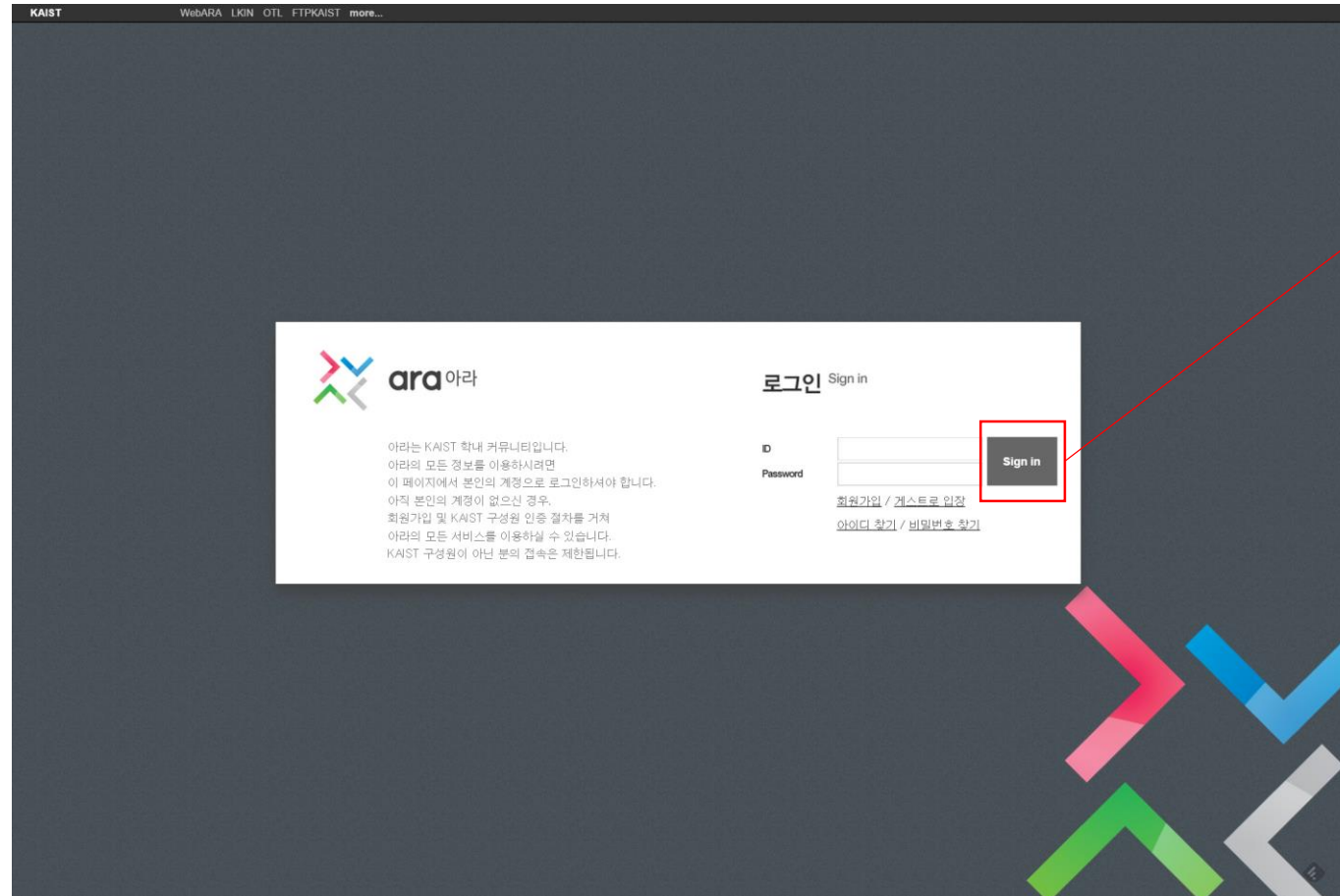
## Front end programming

- 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 쓰시다.  
또또칸 애들이 이미 잘 만들어뒀으니까요.

Let the Django Begin

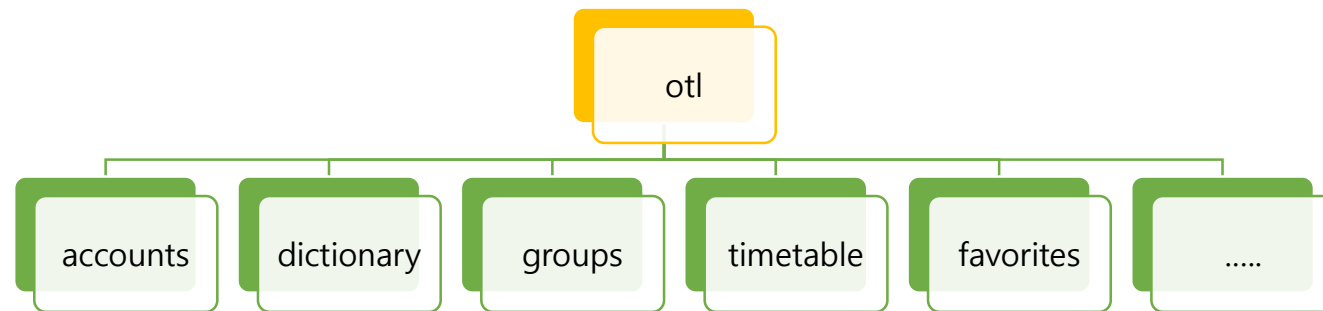
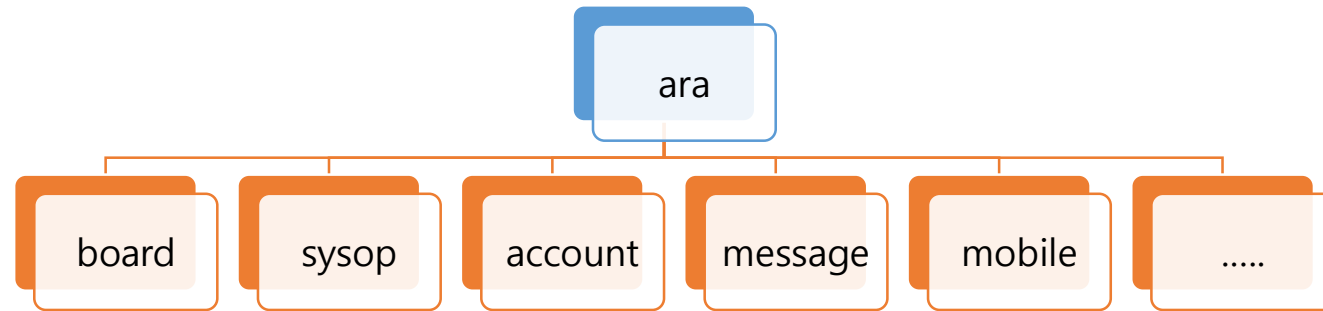
# Basic Structure

Project and Applications



# Basic Structure

Django 웹 어플리케이션은 project와 app으로 구성되어 있다.



# 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

It worked!

Congratulations on your first Django-powered page.

Of course, you haven't actually done any work yet. Next, start your first app by running `python manage.py startapp [app_label]`.

You're seeing this message because you have `DEBUG = True` in your Django settings file and you haven't configured any URLs. Get to work!

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!")
```

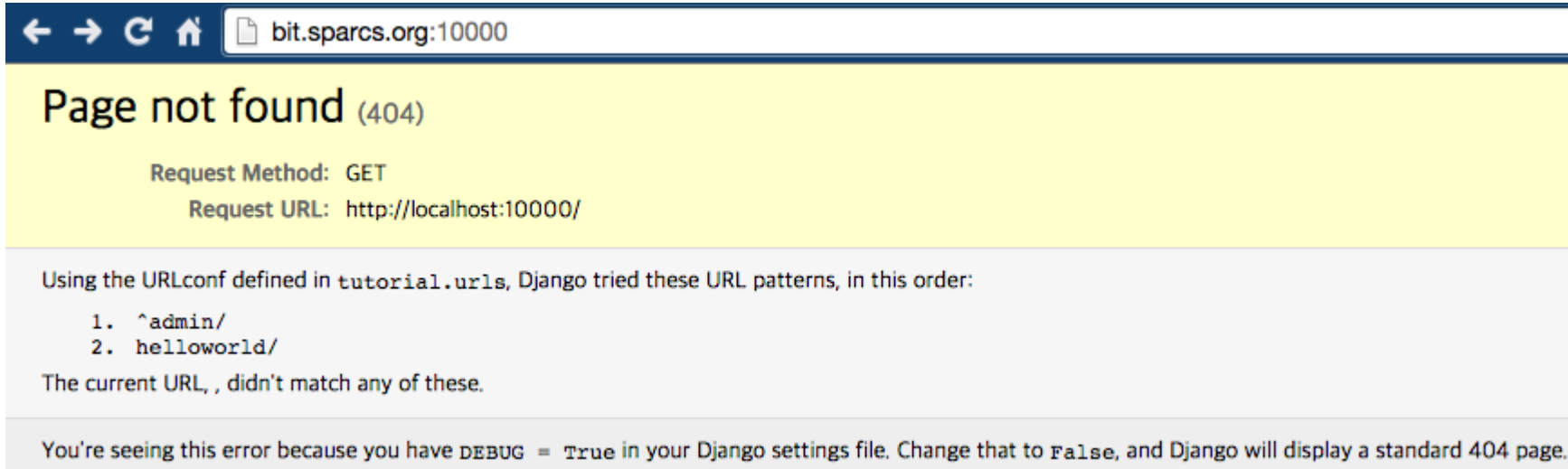
# Map a URL for a View Function

```
(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
```



The screenshot shows a web browser window with the address bar containing "bit.sparcs.org:10000". The main content area has a yellow background and displays the following text:

Page not found (404)

Request Method: GET  
Request URL: http://localhost:10000/

Using the URLconf defined in `tutorial.urls`, Django tried these URL patterns, in this order:

1. `^admin/`
2. `helloworld/`

The current URL, , didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

# Run Server Again

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



← → ↻ 🏠 bit.sparcs.org:10000/helloworld

Hello, World!



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 가세요..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 모듈화합니다.
- /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 개가 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
```



← → ↻ 🏠 bit.sparcs.org:10000/helloworld/introduce/

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

[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
```

← → ↻ 🏠 bit.sparcs.org:10000/intro/

/intro/스팍스/(자기 학번)/(자기 이름)으로 가주세요!

← → ↻ 🏠 bit.sparcs.org:10000/helloworld/introduce/

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

← → ↻ 🏠 bit.sparcs.org:10000/helloworld

Hello, World!

**나는 자바계의 김정일이다! / K.H.L.**

으아ㅏ아ㅏㅏㅏㅏㅏㅏㅏㅏ  
대포동 발사!!

</board/garbage/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

# Dynamic URL

- URL pattern은 정규식으로 표현한다.
  - 모르면 공부하세요... 뭘 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
```



← → ↻ 🏠 bit.sparcs.org:10000/intro/아제로스/줄진서버/샤망쥬%20(죽음의기사,%20100)/

나는 아제로스 줄진서버의 샤망쥬 (죽음의기사, 100)이다!

Next Time: All About Template