HTTP to WebSocket

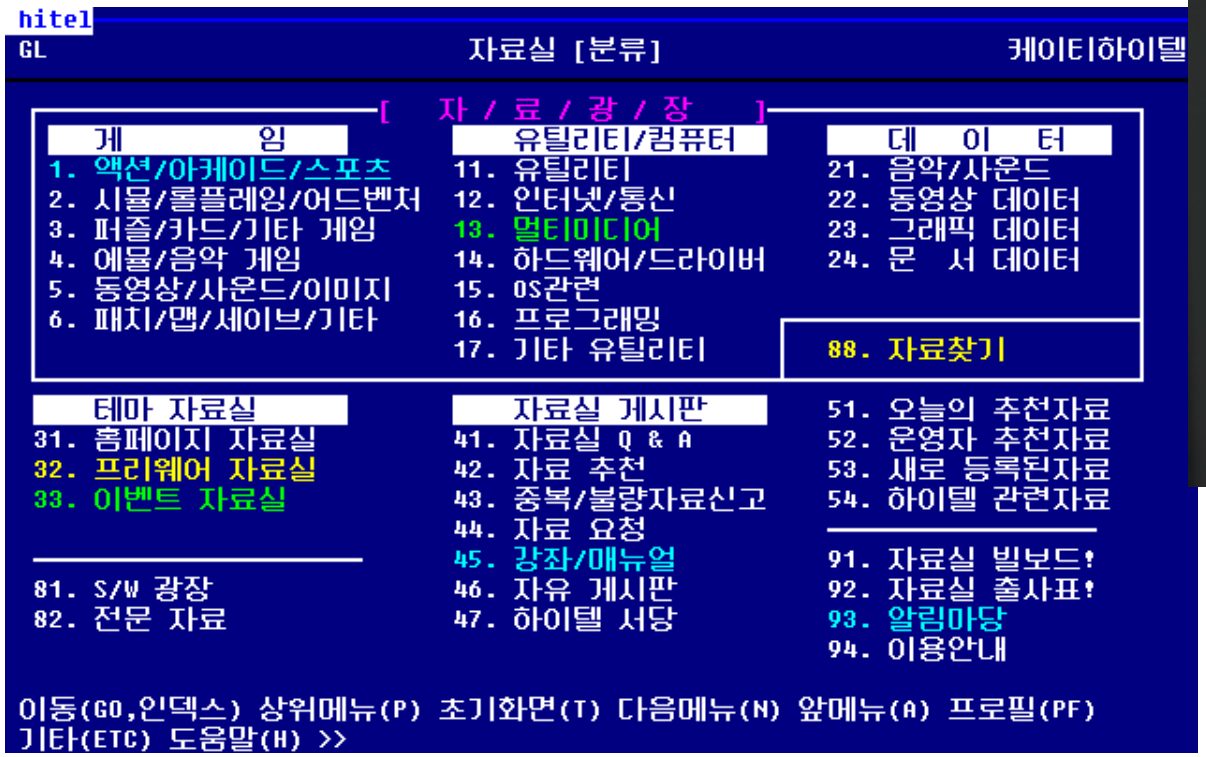
BE: 4번째 세션

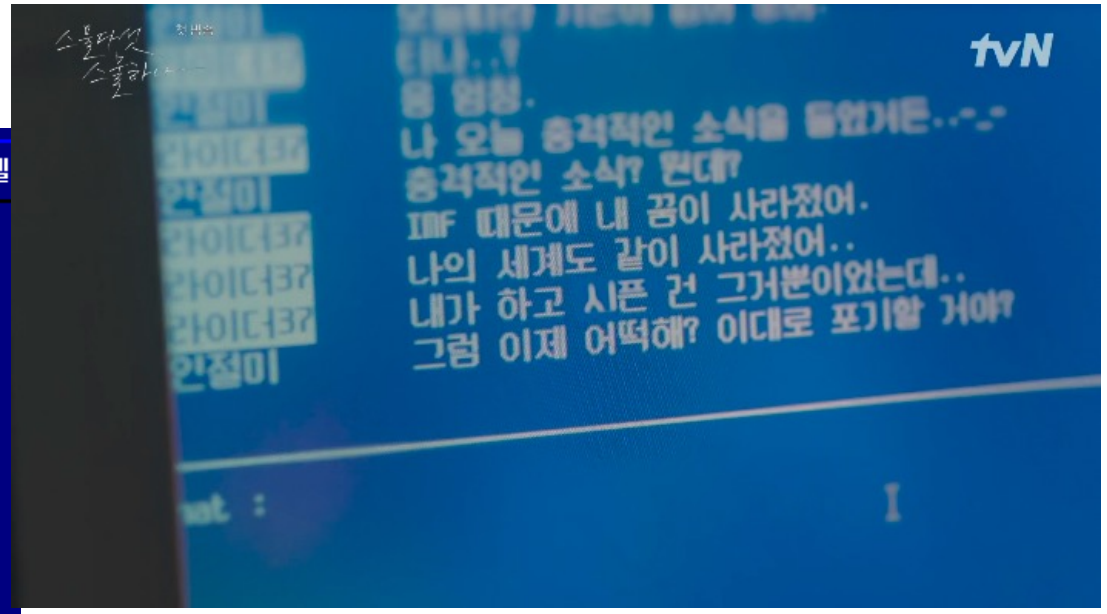
NEXT X LIKELION ㄱㅁㄱ

HTTP to Websocket

HTTP란?

Before HTTP,,,

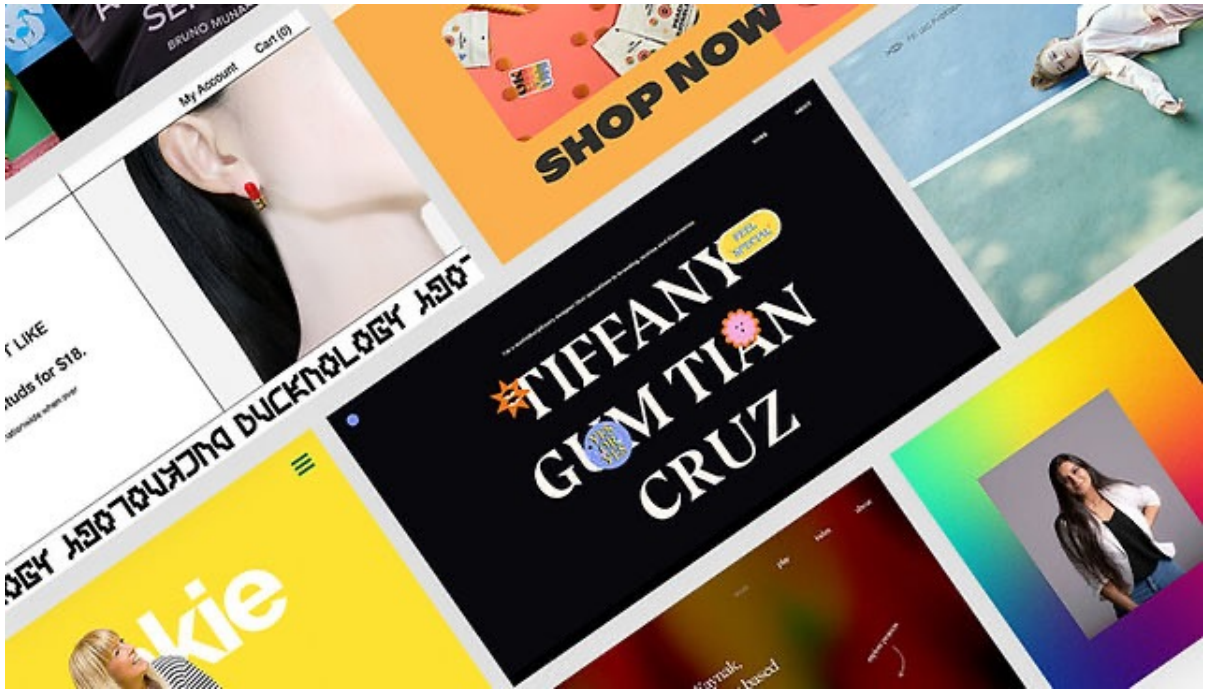




HTTP to Websocket

HTTP란?

Hyper(개쩌는) Text(텍스트) Transfer(전송) Protocol(규약)



URL 및 부가 정보로 요청을 보내면,

관련된 페이지(HTML)로 응답을 내려준다.

* HTML : Hyper Text Markup Language

이제까지 django에서 했던 SSR과 흡사해보이죠?

HTTP to Websocket

HTTP란?

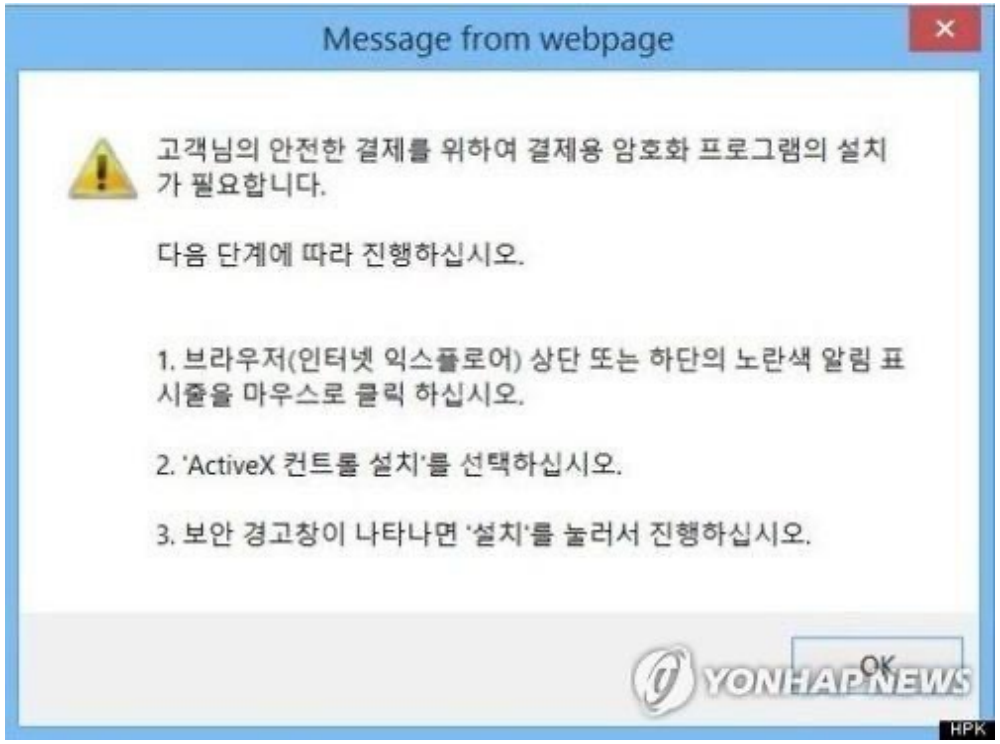
근데 매번 데이터 주고 받을 때 페이지를 이동해야해…?

개에반데

HTTP to Websocket

HTTP란?

개꼼수 active X의 등장.



순수한 웹 환경이 아닌 별도의 플러그인을 통해 데이터를 주고 받는 것

HTTP to Websocket

HTTP란?

혁신의 AJAX…

Ajax
(Asynchronous JavaScript + XML)

- Ajax는 XMLHttpRequest라는 자바스크립트 객체를 활용해 웹 서버와 비동기로 통신하고 DOM을 이용하여 웹 페이지를 동적으로 갱신하는 프로그래밍 기법
- 다시 말해, 웹 페이지가 로딩된 후 일부 데이터를 서버로부터 요청할 수 있도록 함.
- Ajax를 위해서는 XMLHttpRequest 객체를 만들어야 하지만, 편의상 fetch API와 axios 라이브러리를 사용해 데이터를 송수신함.

Session 13 NEXT X LIKELION

Session 13에서 다루었습니다.

HTTP to Websocket

websocket

그렇게 세월은 흘러 흘러…

도착한 곳은 2014년,

html 5가 출시되면서 **websocket**이 등장한다…

Websocket

websocket

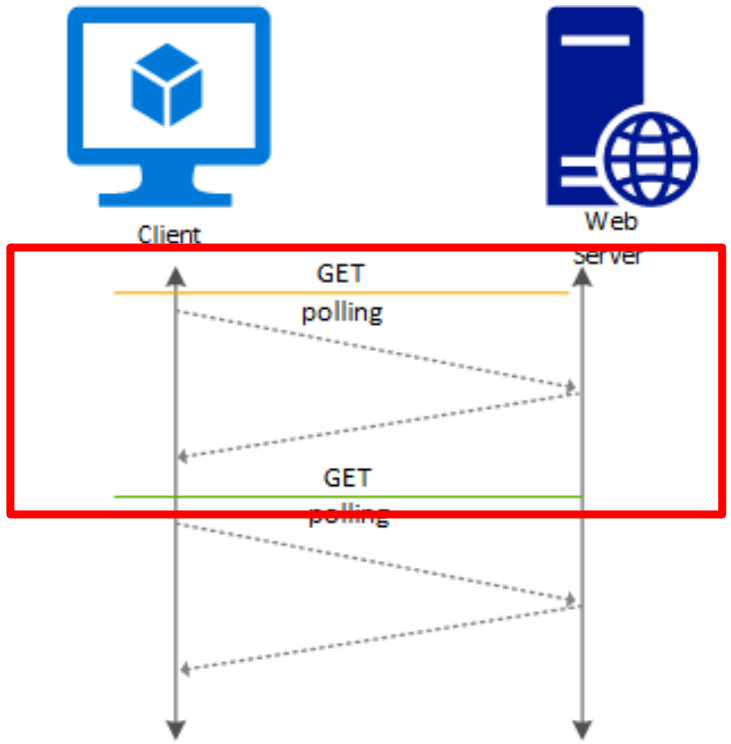
HTTP의 문제,

”클라이언트의 요청이 없으면 서버로부터의 응답이 불가능하다”

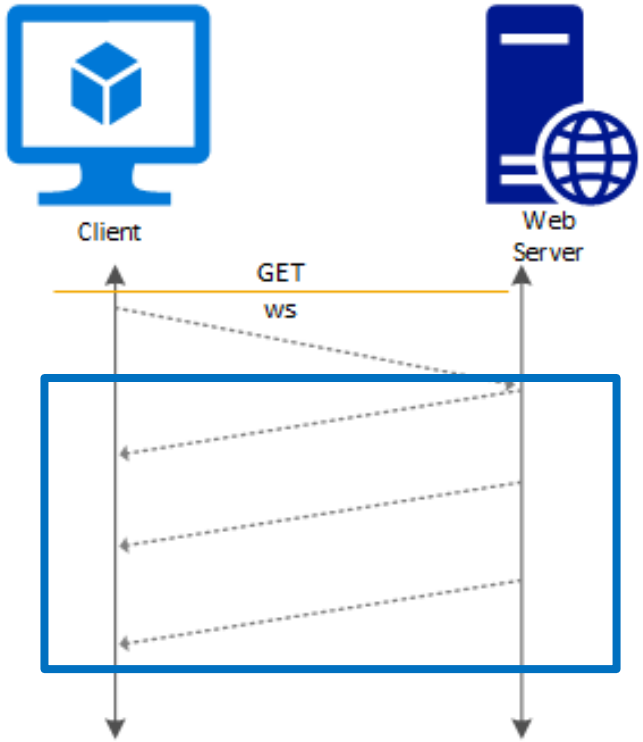
를 해결하기 위해 등장한 새로운 형태의 프로토콜(통신 규약)

Websocket

websocket



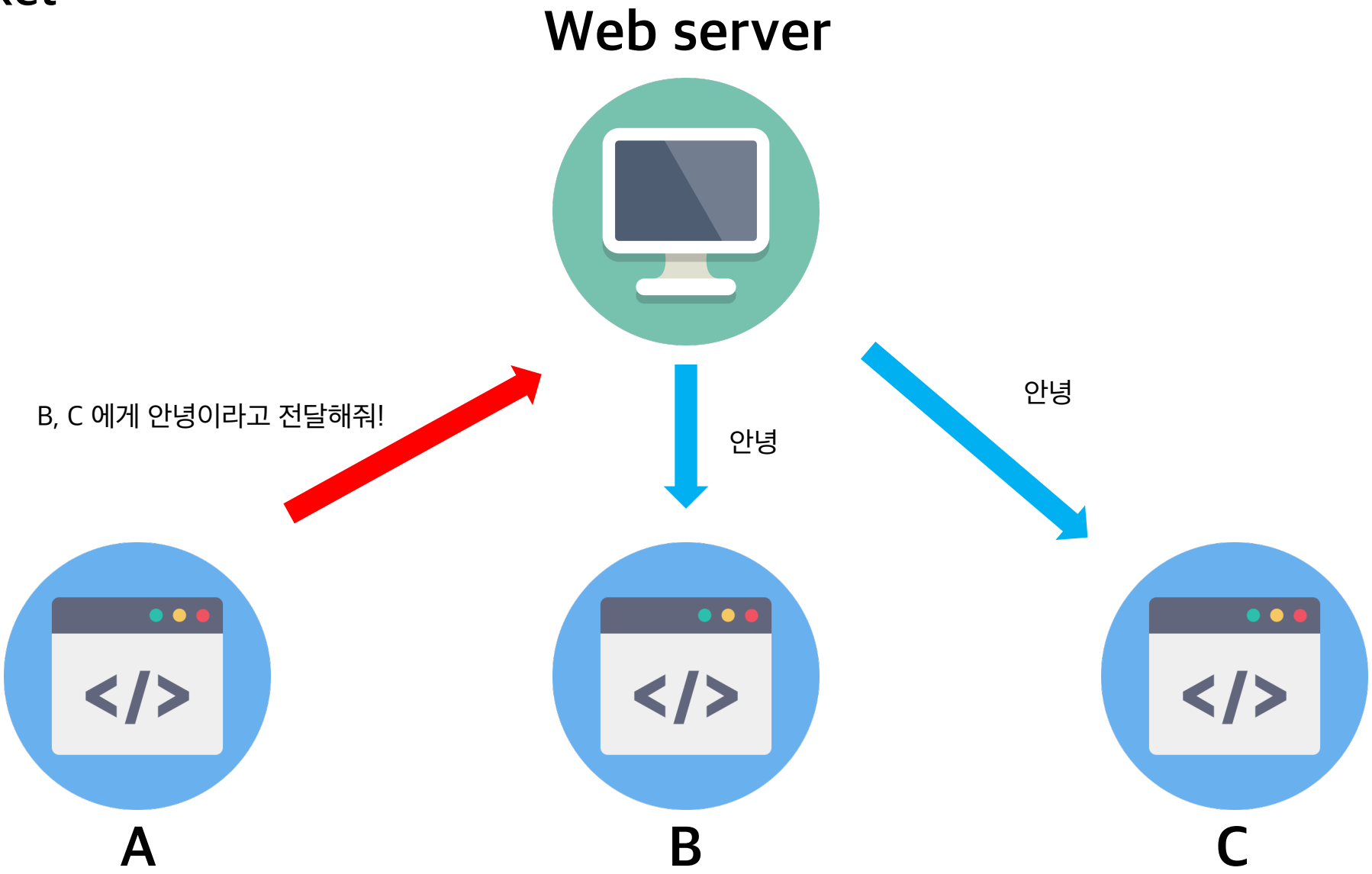
클라이언트의 요청이 존재할때만,
서버의 응답을 받을 수 있음.



최초 websocket 연결 후,
서버에서 필요한 데이터를 내려줄 수 있음.

Websocket

websocket



Websocket

websocket

그래서 그거 어떻게 하는데요…?

Websocket

Django channels

Django Channels 모듈을 사용해서

간단하게 websocket에 대해

구현해보겠습니다.

Websocket

Django channels

```
pipenv shell
pip install django
django-admin startproject websocket

//channels 모듈 설치
python -m pip install -U channels

//channels 모듈 설치 확인
python3 -c 'import channels; print(channels.__version__)’

python manage.py startapp chat
```

Websocket

Django channels

```
/chat
        /templates
                index.html
        consumers.py
        routing.py
        urls.py
        views.py
        ...
/websocket
        asgi.py
        settings.py
        urls.py
        wsgi.py
        asgi.py
```

Websocket

Django channels

websocket/settings.py

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

```
ASGI_APPLICATION = 'socketProject.wsgi.application'
```

Websocket

Django channels

chat/urls.py

```python
from django.urls import path

from . import views

urlpatterns = [
    path('', views.index, name='index'),
]
```

chat/views.py

```python
from django.shortcuts import render

# Create your views here.

def index(request):
    return render(request, 'chat/index.html')
```

websocket/urls.py

```python
from django.contrib import admin
from django.urls import path

from django.urls import include, path

urlpatterns = [
    path('admin/', admin.site.urls),
    path('chat/', include('chat.urls')),

]
```

Websocket

Django channels

websocket/asgi.py & wsgi.py

```
import os

from channels.auth import AuthMiddlewareStack
from channels.routing import ProtocolTypeRouter, URLRouter
from channels.security.websocket import
AllowedHostsOriginValidator
from django.core.asgi import get_asgi_application

os.environ.setdefault("DJANGO_SETTINGS_MODULE",
"mysite.settings")
django_asgi_app = get_asgi_application()

import chat.routing

application = ProtocolTypeRouter({
  "http": django_asgi_app,
   'websocket': AuthMiddlewareStack(
      URLRouter(
         chat.routing.websocket_urlpatterns
      )
   ),

})
```

Websocket

Django channels

chat/consumers.py

```python
from channels.generic.websocket import WebsocketConsumer
import json

class ChatConsumer(WebsocketConsumer):
    def connect(self):
        self.accept()

    def disconnect(self, close_code):
        pass
```

consumers.py는

클라이언트 접속 시,

websocket 통신을 할 수 있는 **인스턴스를 생성**하는 역할을 함.

해당 인스턴스에는 클라이언트가 websocket 내에서 할 수 있는

기능(함수)들이 정의되어 있음.

*현재는 connect와 disconnect만 가능.

Websocket

Django channels

chat/routing.py

```python
from django.urls import re_path

from . import consumers

websocket_urlpatterns = [
    re_path('chat/public', consumers.ChatConsumer.as_asgi()),
]
```

routing.py는

클라이언트 접속 시,

어떤 websocket **인스턴스를 생성하고 접근할지를 결정함.**

Django의 urls.py와 흡사함.

Websocket

Django channels

자 이제 channels를 통해 websocket으로 데이터를 받아보자!

Websocket

Django channels

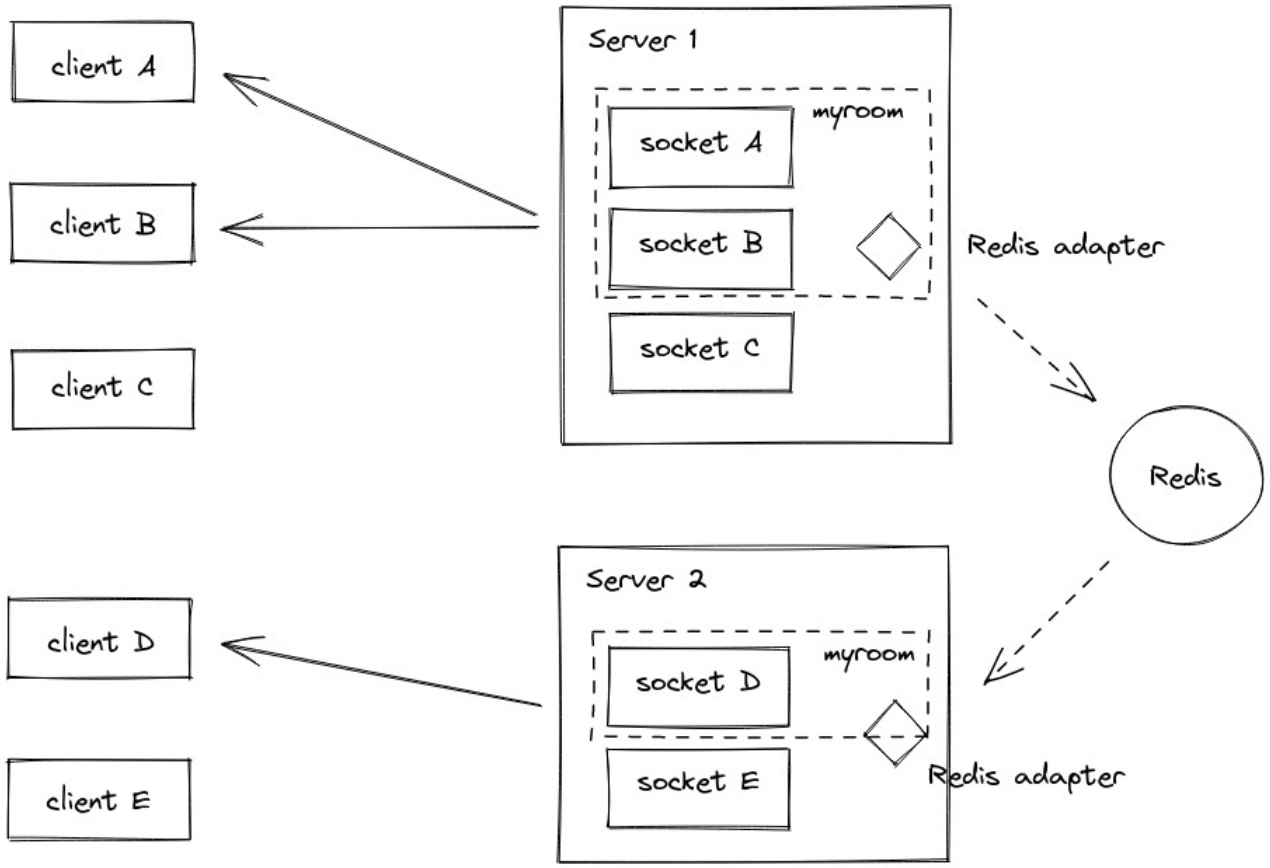
chat/templates/index.html

```
<html>
<head>
    <meta charset="utf-8"/>
    <title>Chat Rooms</title>
</head>
<body>
<div id="connection">not connected</div><br>
<div id="chat-log"></div> <br>
<input id="chat-sender" type="text">
<input id="chat-message-input" type="text"> <br>
<input id="chat-message-submit" type="button" value="Send">

<script>
    let socket = new WebSocket('ws://127.0.0.1:8000' + '/chat/public');
    socket.onopen = function open() {
        setTimeout(()=>{
            document.querySelector('#connection').innerHTML = "connected"
        }, 3000)
    };

    socket.onmessage = (e) => {
        const data = JSON.parse(e.data);
        document.querySelector('#chat-log').innerHTML += (data.sender + " : " + data.message + '\n');
    }

    document.querySelector('#chat-message-submit').onclick = () => {
        const messageInputDom = document.querySelector('#chat-message-input');
        const message = messageInputDom.value;
        const senderInputDom = document.querySelector('#chat-sender');
        const sender = senderInputDom.value
        socket.send(JSON.stringify({
            'sender' : sender,
            'message' : message
        }))
    }
```

Websocket

Additional

Room, namespace, 그리고 redis



과제

1. Django Channels Tutorial 도전

https://channels.readthedocs.io/en/stable/tutorial/part_1.html