

나만의 학교 위키 만들기

1차시 - 오리엔테이션 🤝

설리번 프로젝트 Web1팀 — 만나서 반가워요 :D

목차

1. 설리번 프로젝트를 소개합니다
2. 우리의 교육은요...
3. 커리큘럼
4. 웹과 프로그래밍이란?
5. 개발환경 설치
6. 갈무리



SULLIVAN PROJECT

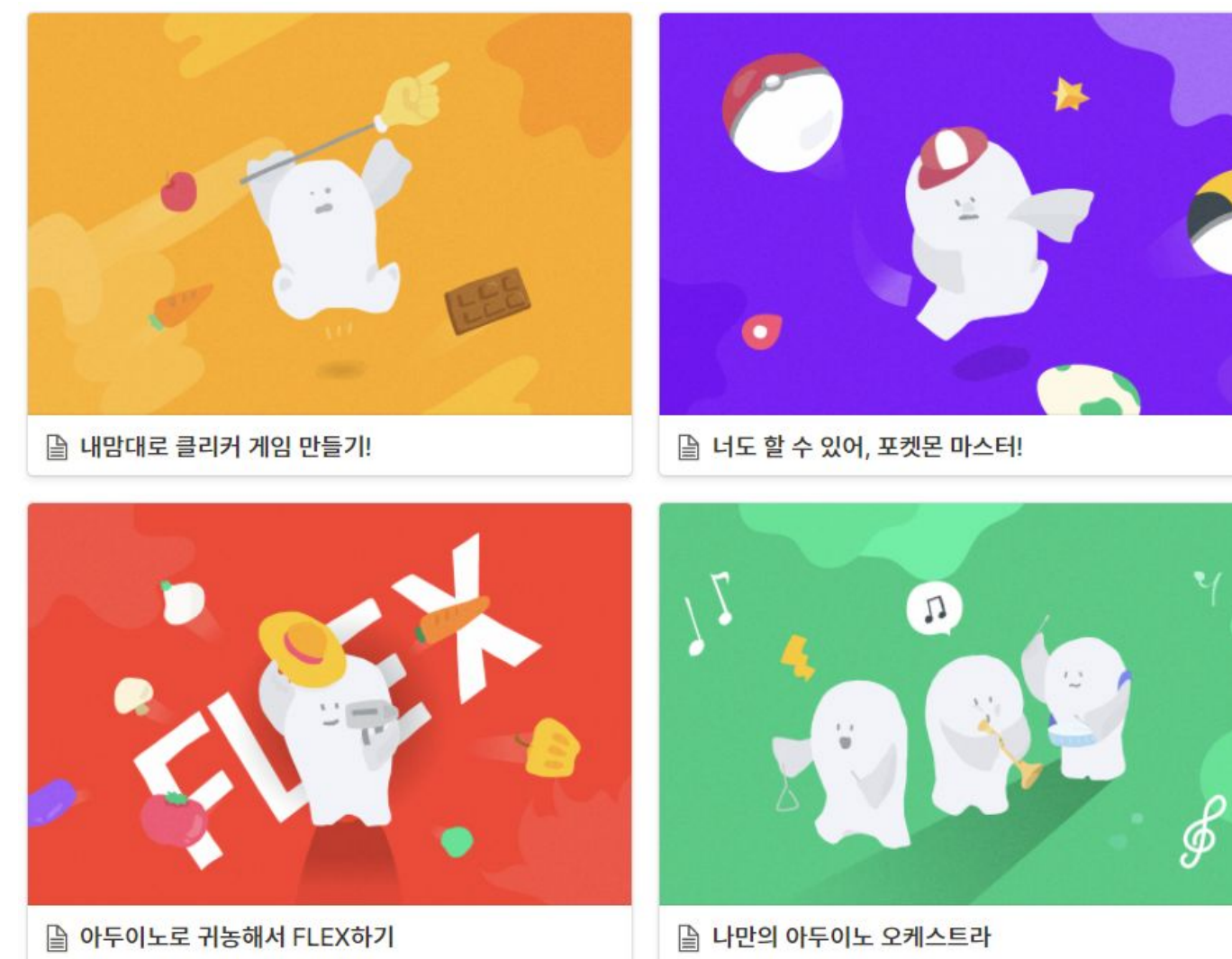
설리번 프로젝트 소개

설리번 프로젝트(Sullivan Project)란?

기술과 배움을 통해

우리 주변의 **문제**를 주도적으로 포착하고 정의하며
협력하여 **해결**할 수 있도록 돕는 교육 프로젝트

내용이 여기를 넘지
않도록
주의해주세요!



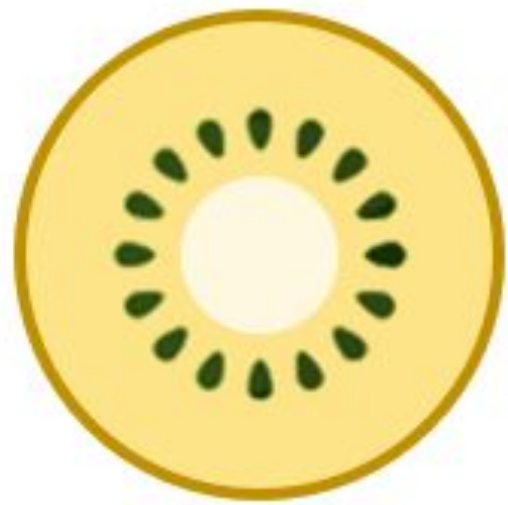
SULLIVAN PROJECT

설리번 프로젝트 소개

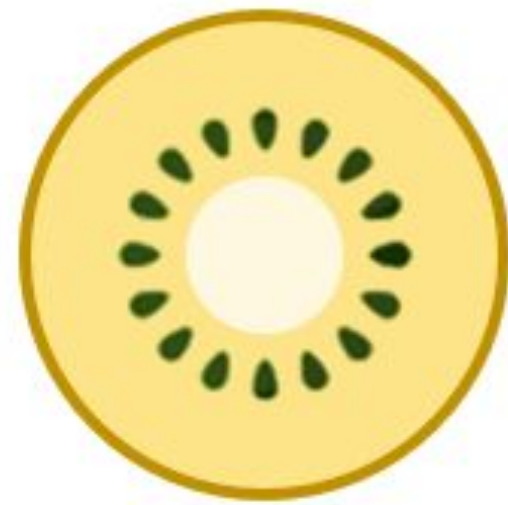
설리번 선생님 소개

만나서 반갑습니다!

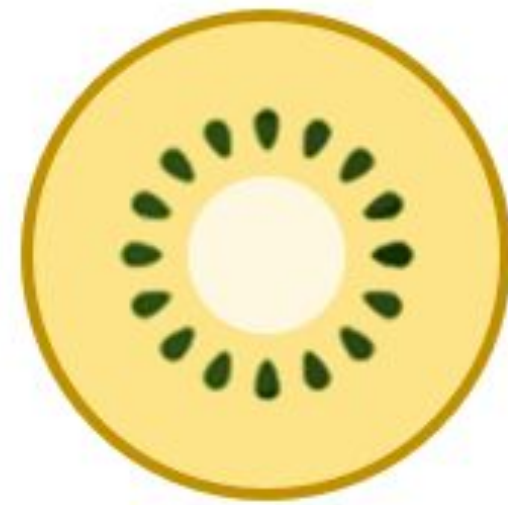
다양하고 유익한 교육적 영양소를 전달하겠습니다!



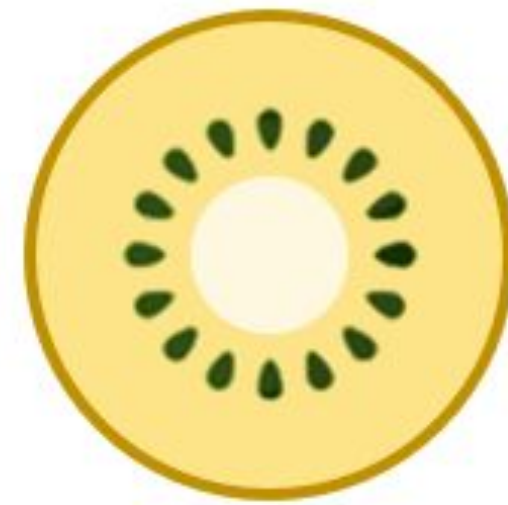
곽현지 선생님



김민지 선생님



김의경 선생님



오지영 선생님

내용이 여기를 넘지
않도록
주의해주세요!

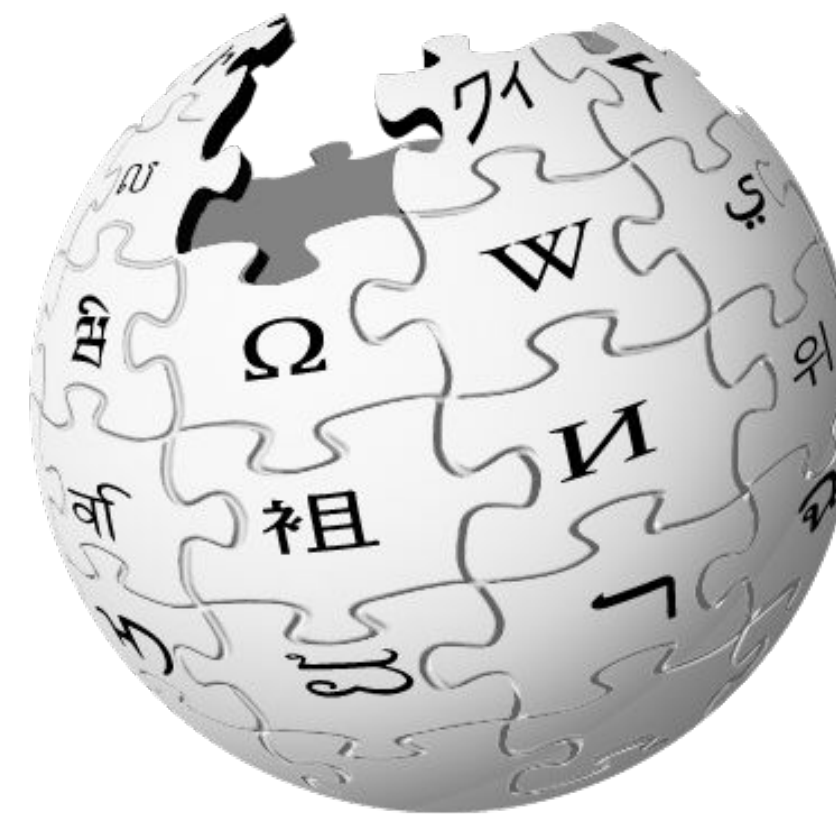
우리의 교육은요...

교육 소개 및 목표

내 손으로 만드는 우리학교 위키!

내가 만드는 우리학교 백과사전!

- 학생들의 학교 생활에 도움을 주는 **참여형** 사전
- 학생이 운영 주체가 되어 학생들 만의 문화 형성
- 정보통신 윤리의식 제고



위키백과
우리 모두의 백과사전

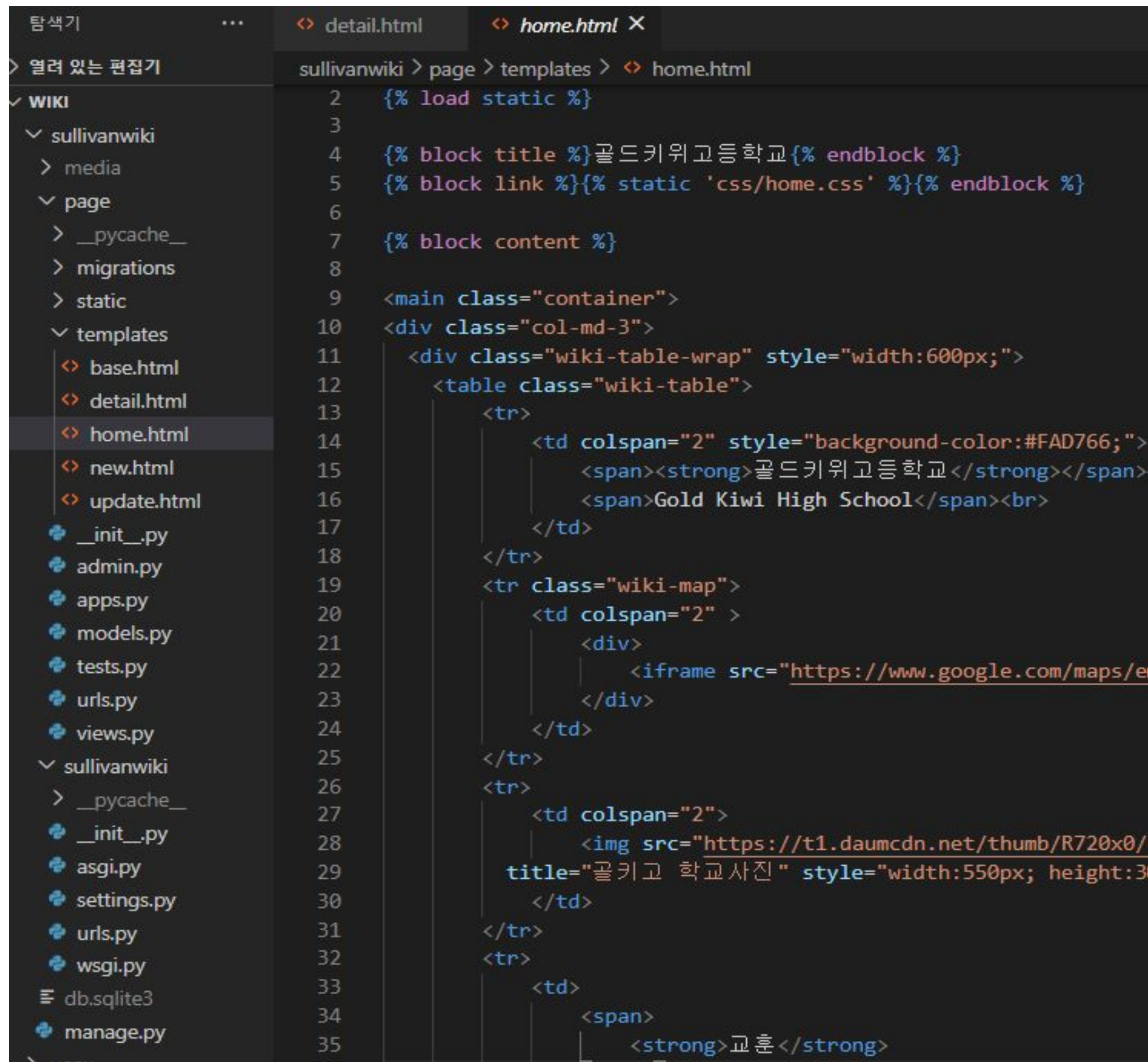
1. 웹 프로그래밍과 정보통신 윤리에 대한 이해
2. HTML, CSS, Python, Django의 기본적인 특성 및 문법 이해
3. 기본적인 위키의 틀을 바탕으로 우리학교 위키 사이트 제작

내용이 여기를 넘지
않도록
주의해주세요!

우리의 교육은요...

교육이 끝나면...

내용이 여기를 넘지
않도록
주의해주세요!



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a 'WIKI' folder containing 'sullivanwiki', which has subfolders 'media', 'page', 'static', and 'templates'. The 'templates' folder is expanded, showing 'base.html', 'detail.html', 'home.html' (selected), 'new.html', and 'update.html'. The code editor shows the content of 'home.html' with the following code:

```
2 {% load static %}
3
4 {% block title %}골드키위고등학교{% endblock %}
5 {% block link %}{% static 'css/home.css' %}{% endblock %}
6
7 {% block content %}
8
9 <main class="container">
10 <div class="col-md-3">
11 <div class="wiki-table-wrap" style="width:600px;">
12 <table class="wiki-table">
13 <tr>
14 <td colspan="2" style="background-color:#FAD766;">
15 <span><strong>골드키위고등학교</strong></span></td>
16 <span>Gold Kiwi High School</span><br>
17 </td>
18 </tr>
19 <tr class="wiki-map">
20 <td colspan="2" >
21 <div>
22 <iframe src="https://www.google.com/maps/e
23 </div>
24 </td>
25 </tr>
26 <tr>
27 <td colspan="2">
28 
31 </tr>
32 <tr>
33 <td>
34 <span>
35 <strong>교훈</strong>
```

차근차근 웹 프로그래밍에
대해서 공부하면서 이렇게
어렵고 복잡해보이는 코드를
쉽게 만들어 볼 수 있어요!

우리의 교육은요...

교육이 끝나면...

내용이 여기를 넘지
않도록
주의해주세요!

골드키위고등학교 [계시글 등록하기](#)

About
골드키위고등학교 위키입니다. 타인을 저격하는 글 등 부
적합한 게시글은 관리자에 의해 삭제조치 될 수 있습니
다. 그 외의 재미난 정보들은 언제나 환영입니다.)

- 목차**
- 1. 개요
 - 2. 학교 연혁
 - 3. 교훈 및 상징
 - 4. 학교 특징
 - 5. 학교 시설
 - 6. 학교 생활

바로가기

- 중 위키
- 고 위키
- 고 위키

골드키위고등학교

Gold Kiwi High School

코스토코 양평점

큰 지도 보기

교훈	키위먹고잘크자
교장	박키위
교감	김키위
홈페이지	서울위키고등학교

[계시글 모음](#)

골키고 최고 [정보수정](#) [정보삭제](#)

2020/12/27

내 맘속에 골키고가 일등이야...[자세히 알아보기](#)

안녕하세요:) [정보수정](#) [정보삭제](#)

2020/12/27

ㅎㅇㅎㅇ...[자세히 알아보기](#)

ㅎㅇ [정보수정](#) [정보삭제](#)

2020/12/27

ㅎㅇㅎㅇ...[자세히 알아보기](#)

내 손으로 직접 최종
결과물로 우리학교의
위키를 만들 수 있어요!

우리의 교육은요...

수업의 방향

- 교육 방식

- 코로나 상황 악화로 **온라인 수업(총 6차시)**으로 진행
(기간: 2021년 00월 00일 ~ 00월 00일 매주 0요일 강의 업로드)
- 강의 방식: **실시간 화상 강의 + 녹화 강의 혼합**
- 실시간 화상 과제실습 및 질의응답 프로그램 및 채팅방 운영
- **오픈 채팅방**을 통한 상시 소통 창구 운영
- **노션(Notion)** 페이지를 통해 교육 운영
(공지사항, 온라인 강의 영상, 강의 자료, 질의응답 등)

내용이 여기를 넘지
않도록
주의해주세요!

우리의 교육은요...

수업의 방향

● 소통 방식

- 상시적인 소통을 위해 카카오톡 오픈 채팅방에 가입해주세요!

<https://open.kakao.com/o/ggzkLYNc>

- 궁금한 내용, 건의하고 싶은 내용 등 선생님들과 소통해요! (칼답!)
- 본인확인을 위해 꼭 **자신의 이름**으로 들어와주세요!

- **노션(Notion)**에서 온라인 강의 영상, 교육 자료 등을 확인해요!

- 노션을 통해 교육 전반의 안내 사항 및 자료들이 게시될거예요!
- 노션 가입 후 우리의 교육 페이지를 수시로 확인해주세요!

<https://www.notion.so/wiki-73158be5f56d47d8b2daa36f83af820d>

내용이 여기를 넘지
않도록
주의해주세요!

우리의 교육은요...

커리큘럼

내용이 여기를 넘지
않도록
주의해주세요!

HTML/CSS 이해하기 02

- 기본적인 웹 언어인 HTML의 문법과 특징에 대한 이해
- 웹 문서의 전반적인 스타일을 다루는 CSS에 대한 이해
- [과제] 자기소개 페이지 만들기

01 Orientation

- 교육 취지 및 교육 목표 소개
- 소통 채널 안내
- 웹이란 무엇인가?
- 개발 환경 설치 및 사용할 언어 소개
- [과제] 정보통신윤리에 대해 생각해보기

03 Django 이해하기

- 파이썬 기반 웹프레임 워크인 Django에 대해서 알아보고 친해져봐요!

우리의 교육은요...

커리큘럼

내용이 여기를 넘지
않도록
주의해주세요!

Django 실전연습 05

- 본격적으로 Django를 사용하며
지난 시간에 구상한 위키 페이지를
직접 만들어보아요!
- [과제] 우리학교 위키 페이지를
완성해보아요!

04 나만의 위키 구상하기

- 어떻게 위키 페이지를 만들지
구상해보아요!
- Django 개념 이해하기(static, method)
- [과제] 위키 페이지 디자인하기

06 결과 발표회

- 내가 만든 우리학교 위키 페이지에
대해서 발표하는 시간
- 세상에 하나밖에 없는 나만의 위키를
다른 사람들과 공유해보아요.

그럼 웹이 뭐예요?

웹과 프로그래밍 이해하기 🖐️

웹과 프로그래밍이란?

웹 프로그래밍이란?

- 웹(Web)이란?

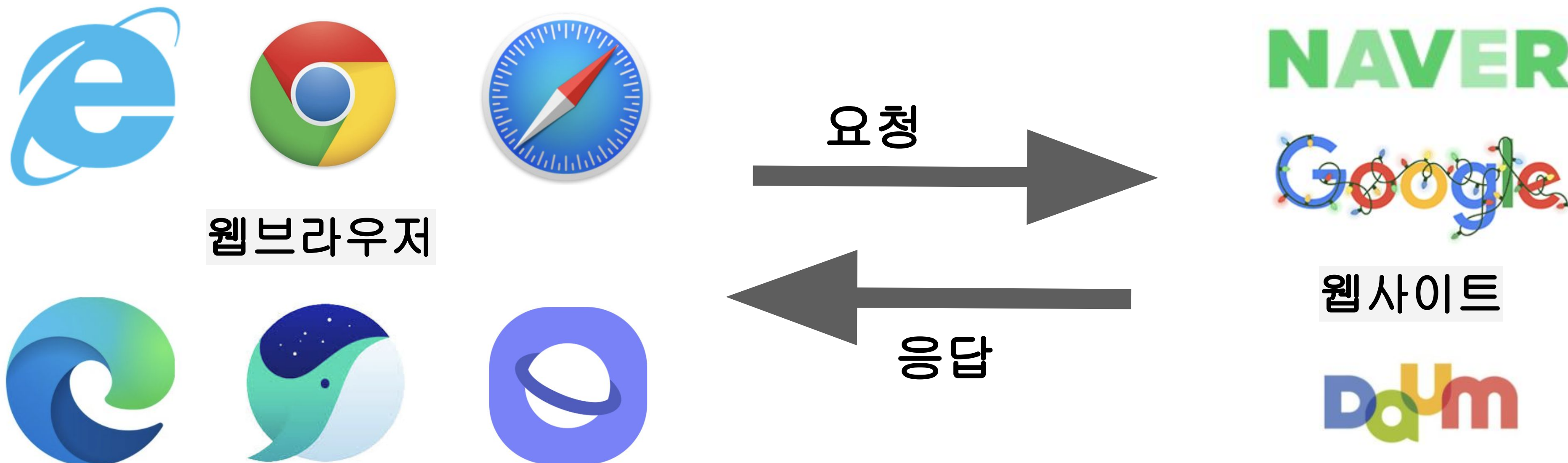
- ‘거미줄’ => ‘세상의 크기만한 거미줄’
- 인터넷에 연결된 컴퓨터들을 통해 사람들이 정보를 공유할 수 있는 전 세계적인 정보 공간
- 인터넷은 컴퓨터 네트워크의 망 자체를 의미하며, 웹은 인터넷 상에서 동작하는 하나의 서비스

내용이 여기를 넘지
않도록
주의해주세요!

웹과 프로그래밍이란?

웹 프로그래밍이란?

- 웹(Web)의 동작 원리
 - 웹브라우저와 웹서버가 협력해서 웹이 동작하게 됨!
 - 서버(server): 서비스 제공(고객에게 **응답**)
 - 클라이언트(client): 서비스 이용하는 고객(서버에게 **요청**)



내용이 여기를 넘지
않도록
주의해주세요!

웹과 프로그래밍이란?

웹 프로그래밍이란?

내용이 여기를 넘지
않도록
주의해주세요!



글로벌 아이돌 육성 프로그램, 프로듀스 101

1. 아이돌 연습생 101명이 출연
2. 팀을 이루어서 춤과 노래 연습
3. 공연 및 국민 프로듀서들의 투표
4. 탈락자 선출, 새로운 팀과 함께 새로운 공연
5. 최종 Wanna One 멤버 11명 선출, 그리고 데뷔

SULLIVAN PROJECT



웹과 프로그래밍이란?

웹 프로그래밍이란?

- 컴퓨터에서 프로그램(program)이란?
 - 여러 가지 데이터(data)에 대해 정해진 **절차대로** 특별한 처리를 수행하여 의사 결정에 사용할 수 있는 정보 (information)를 얻기 위해 컴퓨터에게 내리는 명령을 모아놓은 것
 - 즉, 특정한 문제를 해결하기 위해 컴퓨터에게 내리는 **명령들의 집합** (온라인 메신저, 계산기 등)

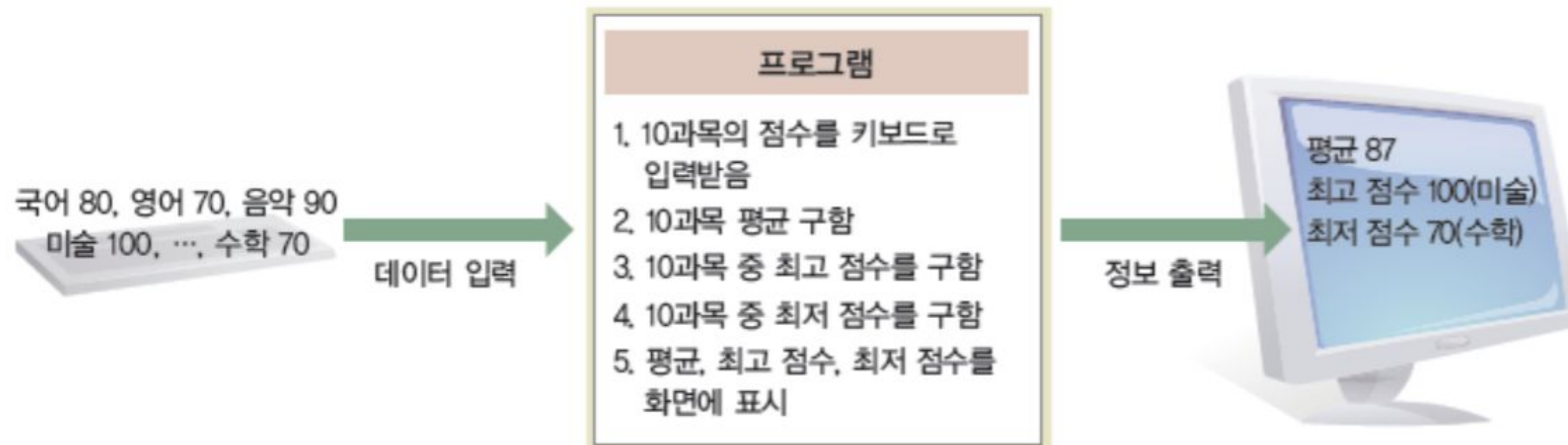
내용이 여기를 넘지
않도록
주의해주세요!

웹과 프로그래밍이란?

웹 프로그래밍이란?

- 프로그래밍(programming)이란?
 - 프로그램을 만드는 것, 개발(development)
 - 프로그래밍 언어를 사용하여 어떤 문제를 해결하기 위한 절차, 방법, 명령어들의 집합인 알고리즘(algorithm) 개발

컴퓨터 프로그램: 10과목의 점수로부터 평균, 최소 점수, 최대 점수 구하기



내용이 여기를 넘지
않도록
주의해주세요!

웹과 프로그래밍이란?

웹 프로그래밍이란?

- 웹프로그래밍(web programming)이란?
 - 인터넷 상에서 사용할 수 있는 웹사이트를 개발하는 일
 - 이때, 웹프로그래밍을 위한 언어 및 개발 환경 필요
 - 프로그래밍 언어: 컴퓨터에게 명령을 전하는 언어
 - 개발 환경: 에디터, 컴파일러, 디버그 등
- 프로그래밍에 필요한 툴이 하나의 인터페이스로 통합되어 사용 할 수 있는 개발 환경 필요

내용이 여기를 넘지
않도록
주의해주세요!

그럼 이제부터 개발환경에 대해서 알아보까요?

개발환경 VS CODE 설치하기 🖱️

본격적으로 시작하기 전에

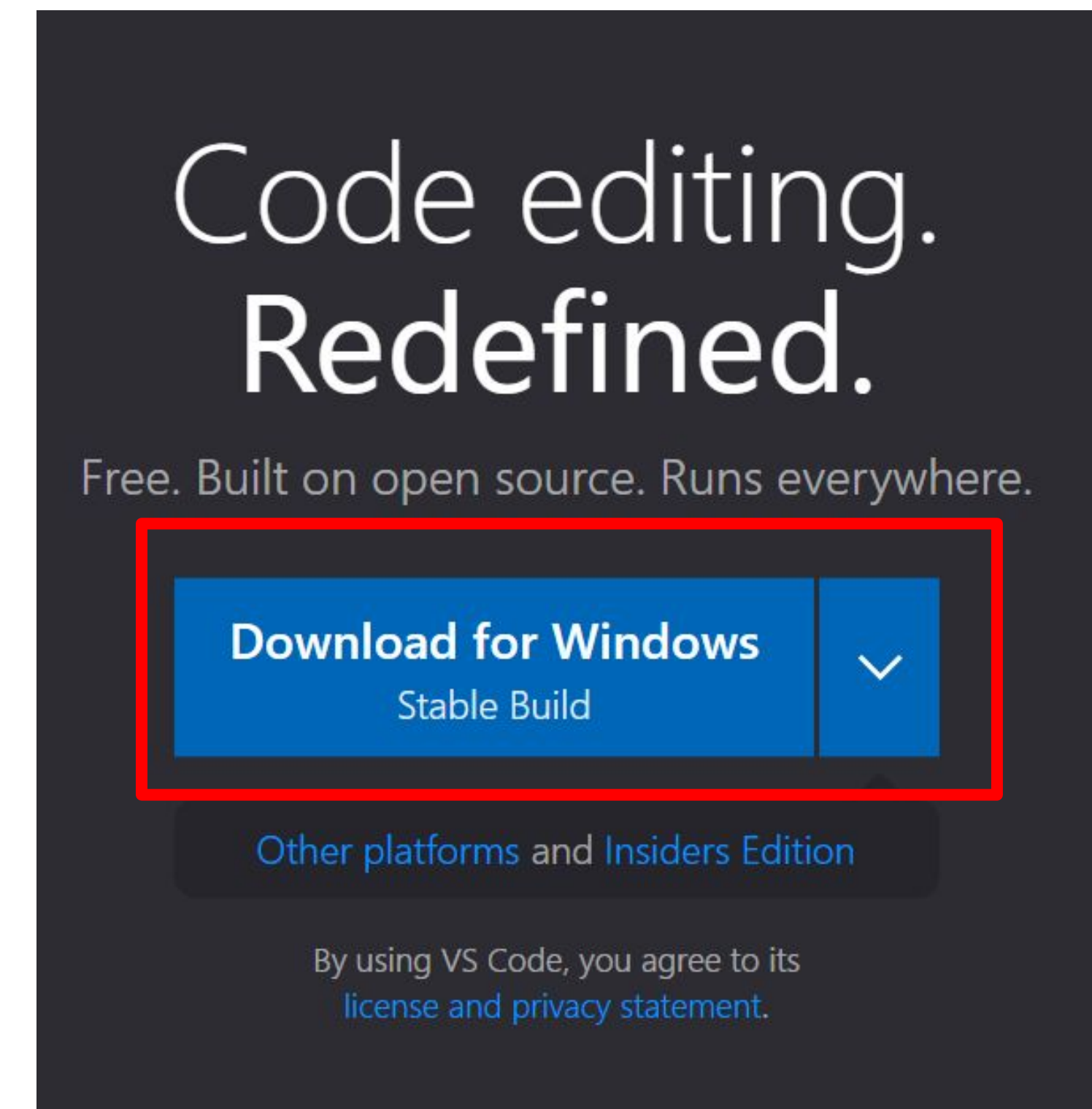
개발환경을 설치해 볼까요?

나만의 위키를 만들기 위해서는 약간의 준비 과정이 필요해요.

우리가 더 편하게 개발할 수 있도록 도와주는 편집기

Visual Studio Code (VSCode)를 설치해 보도록 하겠습니다.

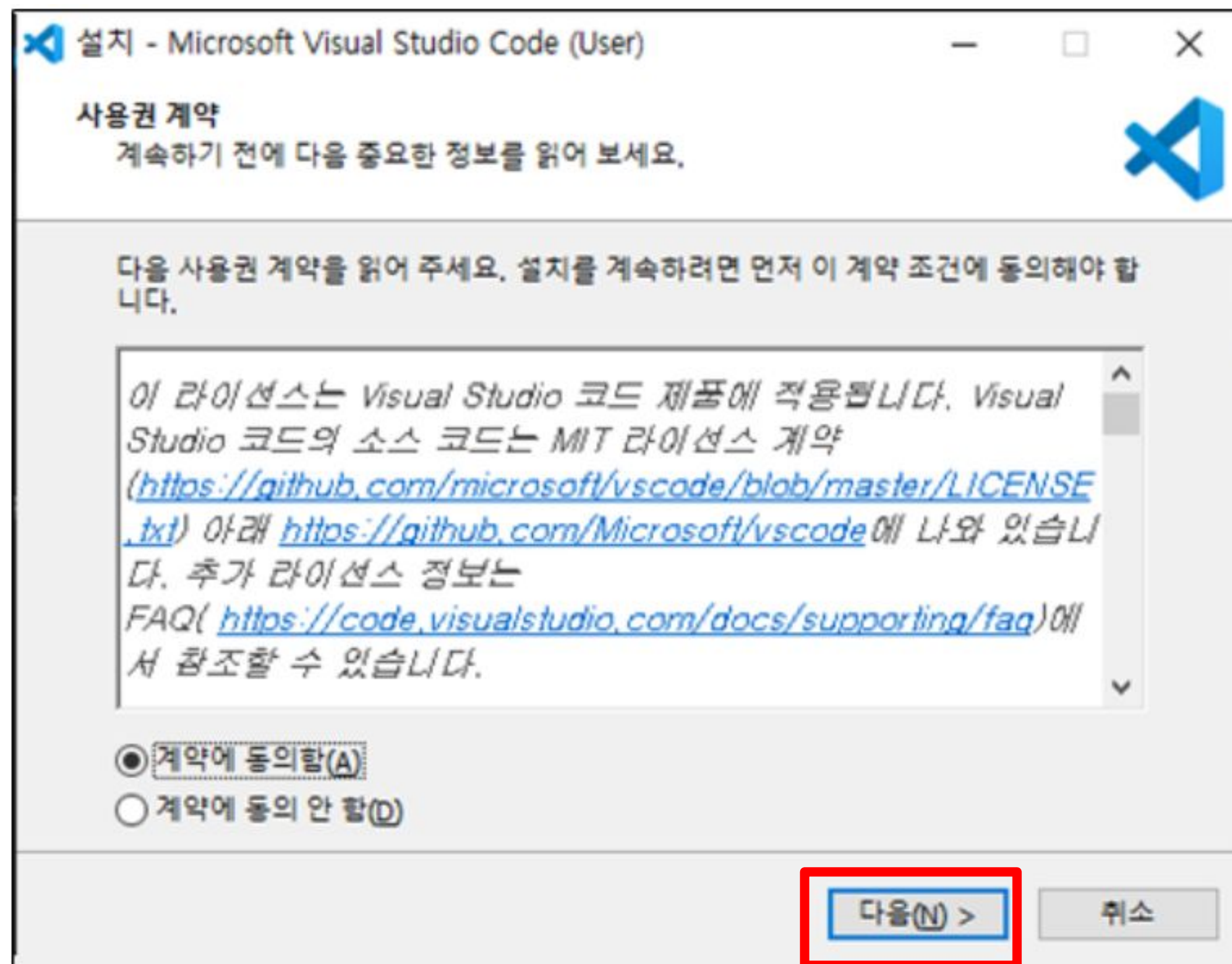
<https://code.visualstudio.com/> 에서 다운로드를 해줍니다.



Visual Studio Code 설치하기

다운받은 파일을 실행하면 설치가 시작됩니다.

“계약에 동의함”을 선택한 뒤 **다음** 버튼을 눌러주세요.

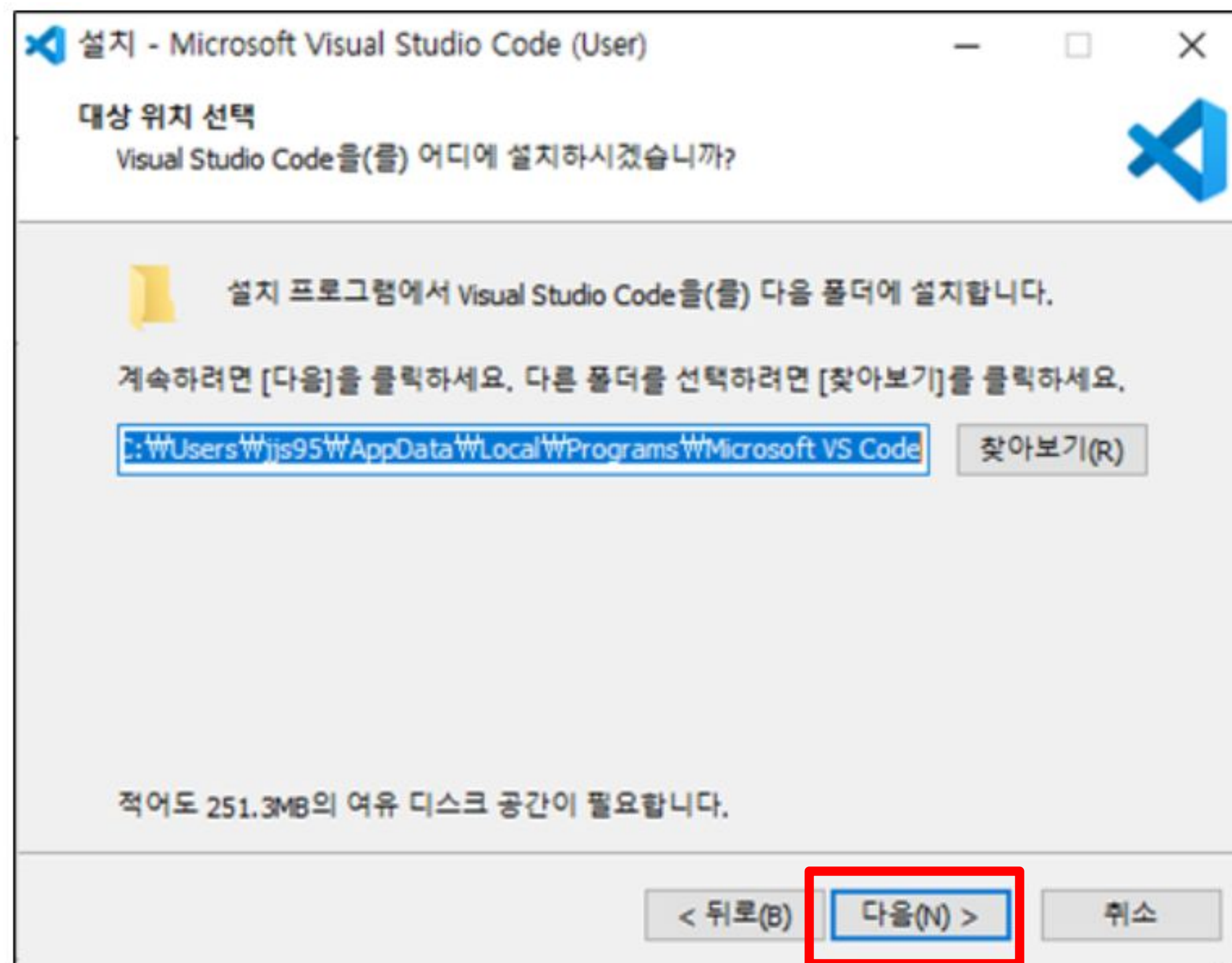


내용이 여기를 넘지
않도록
주의해주세요!

Visual Studio Code 설치하기

설치 경로를 설정하는 옵션입니다.

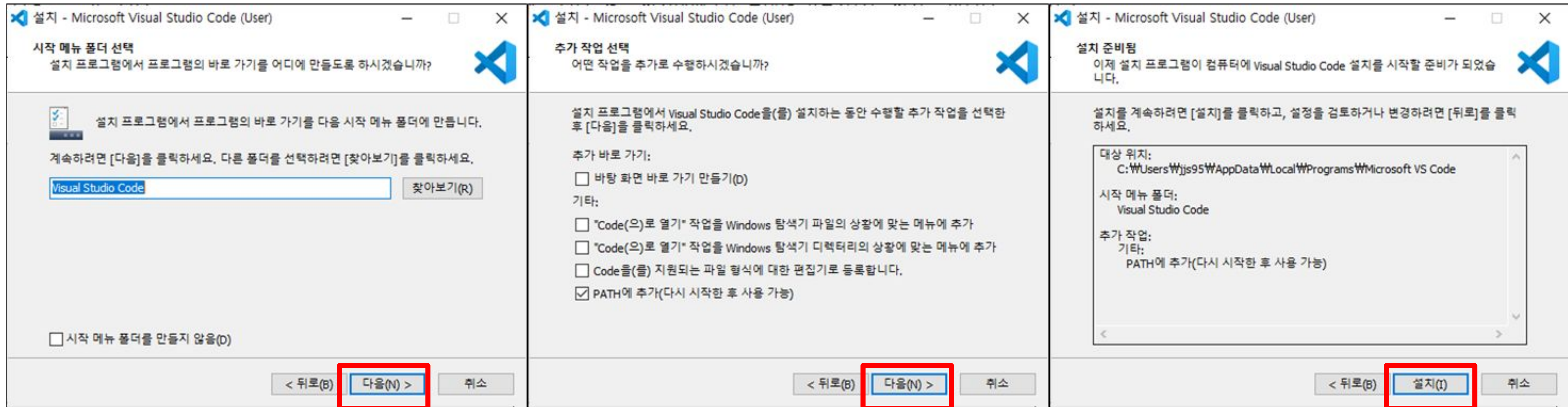
아무 것도 건드리지 않고 **다음** 버튼을 눌러주세요.



내용이 여기를 넘지
않도록
주의해주세요!

Visual Studio Code 설치하기

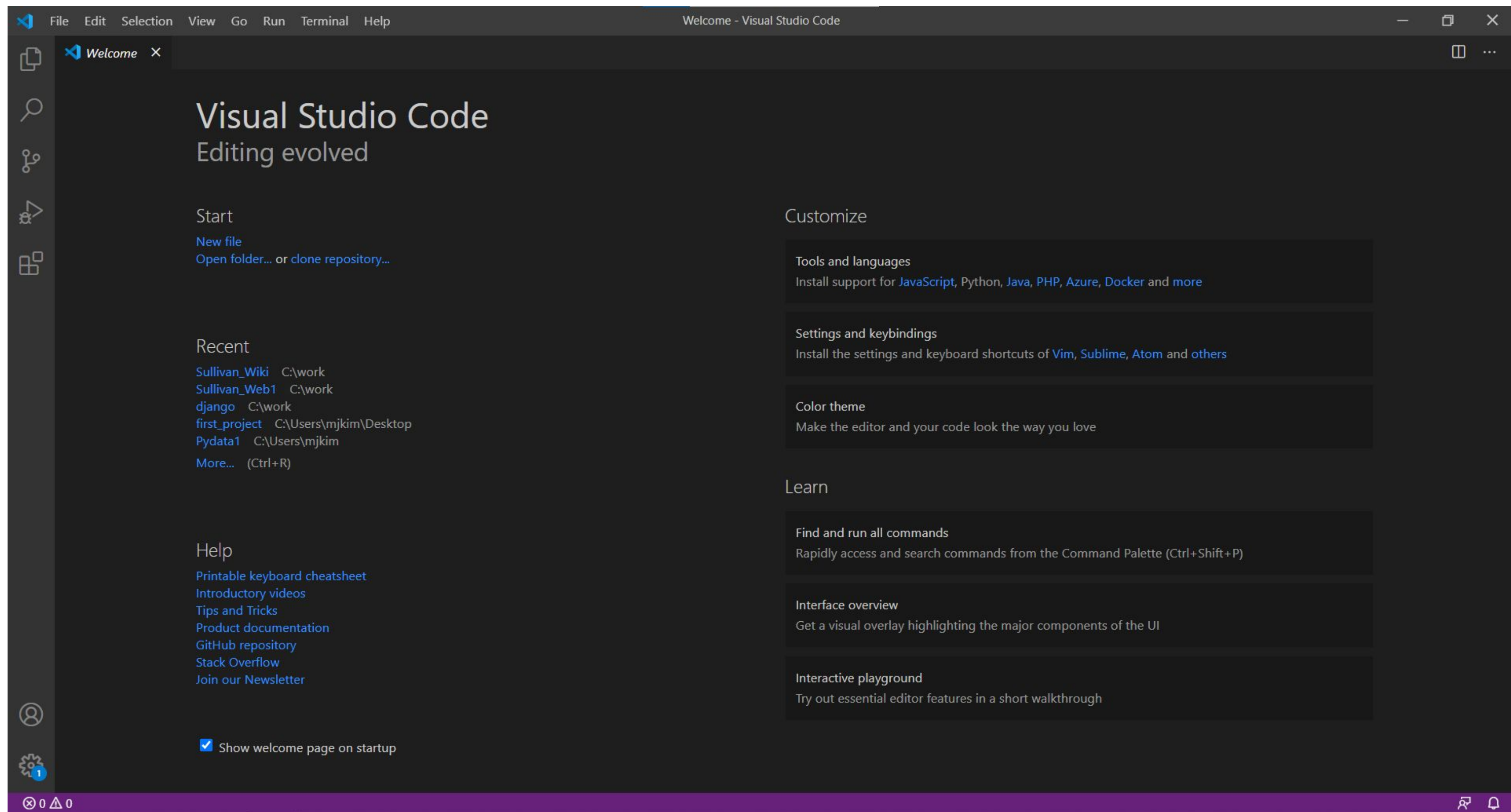
기타 옵션들도 모두 기본값으로 설정하려고 합니다.
계속해서 **다음** 버튼을 누르면 설치가 완료됩니다.



개발환경 설치

Visual Studio Code 설치하기

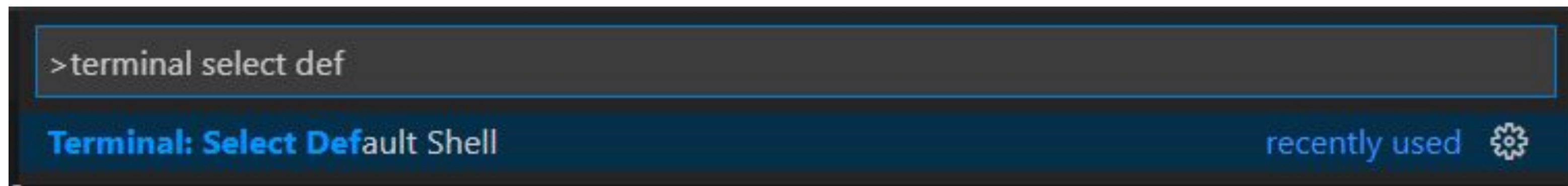
Visual Studio Code가 실행된다면 성공!



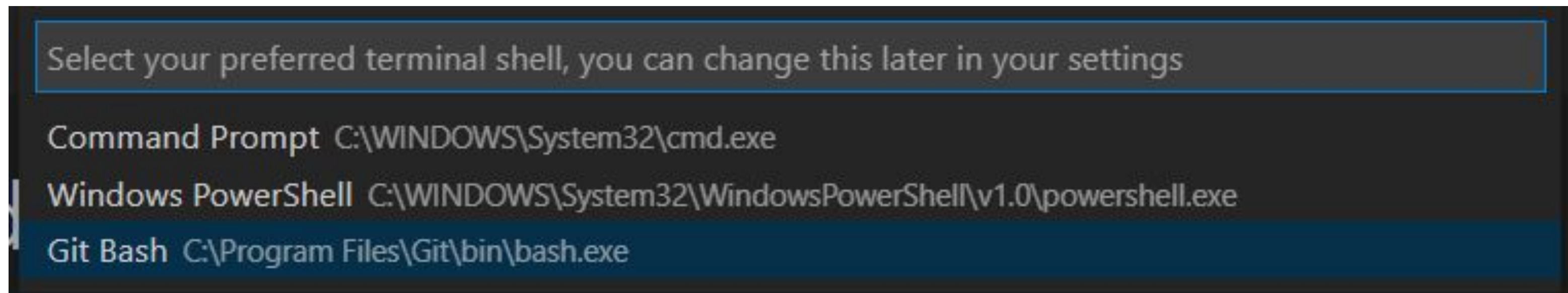
개발환경 설치

Visual Studio Code에서 개발 환경 설정하기

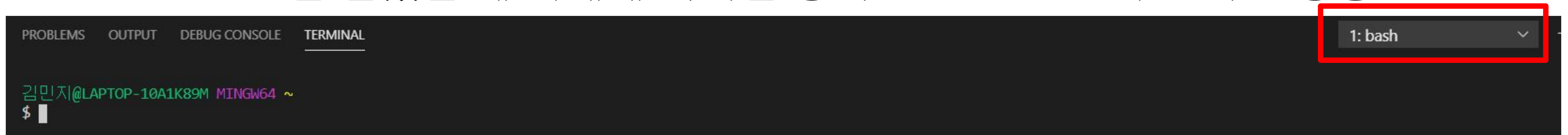
F1 키 → Terminal: Select Default Shell을 클릭한 뒤



Git Bash를 눌러줍니다.



Ctrl + Shift + ` 를 눌렀을 때 아래에 터미널 창이 뜨고 **1:bash**가 뜬다면 성공!




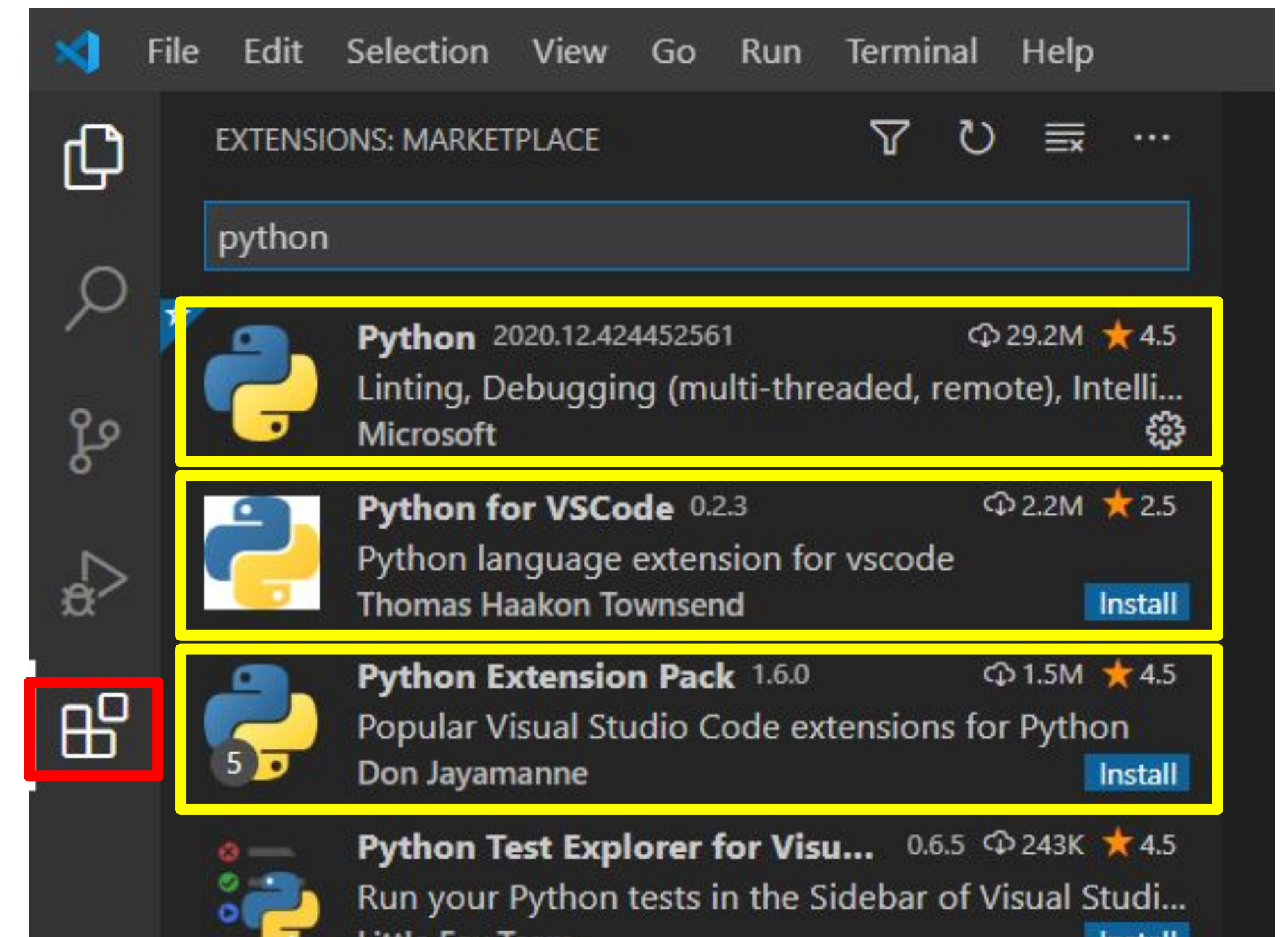
Visual Studio Code에서 파이썬 설치하기

위키의 주요 기능인 글 추가/수정/삭제 기능을 구현하기 위해서는
파이썬(Python)을 설치해야 합니다.

파이썬은 비전공자도 쉽게 배울 수 있는 프로그래밍 언어입니다.

Visual Studio Code에서 파이썬을 편하게 설치할 수 있어요 :)

왼쪽 맨 아래  기호를 클릭한 뒤
검색창에 **python**이라고 입력하면 나오는
3개의 프로그램을 **모두 설치**해주시면 됩니다.



개발환경 설치

Visual Studio Code에서 파이썬 설치하기

터미널 창에서 **python --version**을 입력했을 때
아래와 같이 버전이 정상적으로 나온다면
파이썬이 성공적으로 설치된 것입니다.

```
$ python --version  
Python 3.9.0
```

드디어 본격적인 개발을 위한 준비 과정을 마쳤어요!

다음 시간부터는 직접 웹 페이지를 만들어보도록 하겠습니다👍

노션과 웹하고 친해지기

이번 시간의 간단한 과제를 알려드릴게요!

다음 시간까지 준비할 내용!

1. 노션(Notion) 페이지에 간단한 자기소개와 우리학교 위키에 구현하고 싶은 기능에 대해서 글을 남겨주세요 😊
2. 위키 개발에 있어서 생각해 볼 수 있는 **정보통신 윤리문제** 3가지를 고민해봐요! (무엇이 문제가 될 수 있으며, 그 문제를 어떻게 위키에서 다룰 것인지 등)

나만의 학교 위키 만들기

2차시 - HTML과 CSS를 알아 보아요 🙌

설리번 프로젝트 Web1팀 — 발표 관련 정보 (일시, 자료 등)

목차

1. HTML에 대해 알아보아요!

- a. HTML 문서의 기본 구조
- b. 요소와 속성이란 무엇일까요?
- c. 핵심 기능 소개 (텍스트, 목록, 표, 하이퍼링크, 이미지 삽입)

2. CSS에 대해 알아보아요!

- a. CSS를 시작하려면 (선언 방법)
- b. 웹 페이지 꾸며보기 (폰트, 색상 등 기본적인 속성)

3. [실습] 자기소개 페이지를 만들어보아요!

수업 시작 전 워밍업 🕶️

웹 페이지는 어떻게 만들어질까요?

HTML과 CSS를 왜 배워야 할까요?

종이에 글씨를 쓰기 위해서는 무엇이 가장 필요할까요?
바로 글씨를 쓸 **필기구**입니다.
또한 글씨의 크기나 색깔을 마음대로 바꿀 수도 있겠죠?

여기서 종이가 웹 페이지라면
필기구는 **HTML**이며, 글씨를 꾸며주는 도구는 **CSS**입니다.
HTML은 웹 사이트를 구성하는 기본 언어이며,
CSS는 **HTML**로 만들어진 웹사이트를 꾸며주는 언어입니다.
위키라는 웹 사이트를 만들기 위해 꼭 필요한 존재랍니다!



2차시 - HTML과 CSS를 알아보아요

HTML에 대해 알아보아요!

HTML로 웹 페이지를 만들기 위해서는

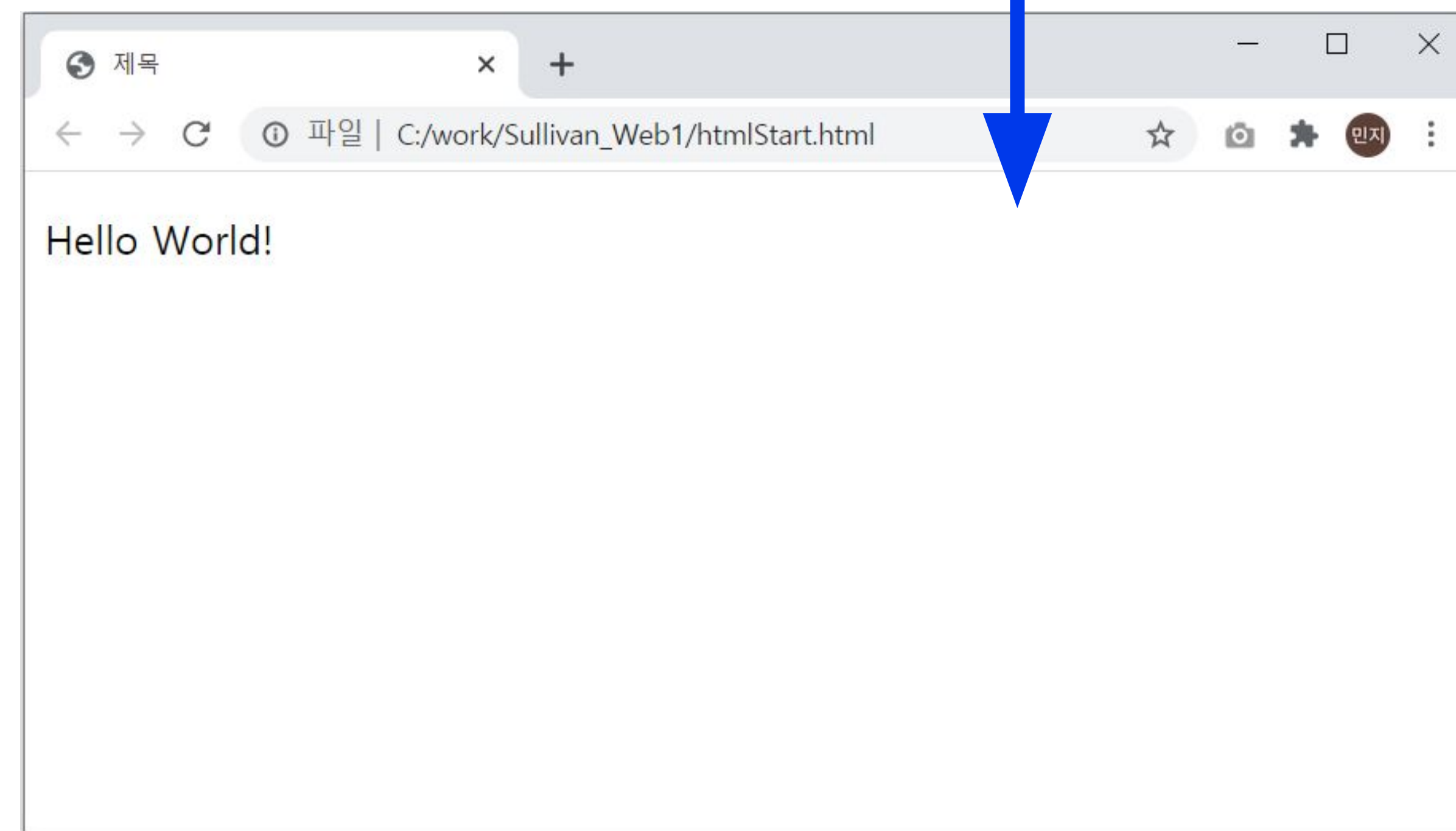
1. 문서 형식을 선언하고
2. <head>, <body>, <title>과 같은 요소들을 입력합니다.

파일명 **.html**로 저장하고 브라우저에서 열어보면 방금 만든 웹 페이지가 나오게 됩니다.

“요소”가 무엇인지는 지금 몰라도 괜찮아요.
우선, HTML로 웹 페이지를 만드는 방법부터 확실히 알아봅시다!

```
1  <!DOCTYPE HTML> ← 문서 형식 선언
2  <html>
3      <head>
4          <title>제목</title>
5      </head>
6      <body>
7          <p>Hello World!</p>
8      </body>
9  </html>
```

HTML 기본 구조



HTML에 대해 알아보아요!

방금 보신 HTML 언어의 특징은 무엇일까요? 모두 < > 기호를 사용한다는 점입니다.

각각의 덩어리들은 모두 **요소**라고 부릅니다.

즉, 요소들이 모여 하나의 HTML 문서가 완성됩니다.

```
<title>제목</title>
```

요소는 **시작 태그**로 열어 **종료 태그**로 닫습니다.

(종료 태그가 없는 경우도 있습니다.)

```
<p align="center">Hello World!</p>
```

태그 사이에 들어가 있는 Hello World!는 **내용**이고,

태그 안에 있는 단어들은 **속성**입니다.

HTML에 대해 알아보아요!

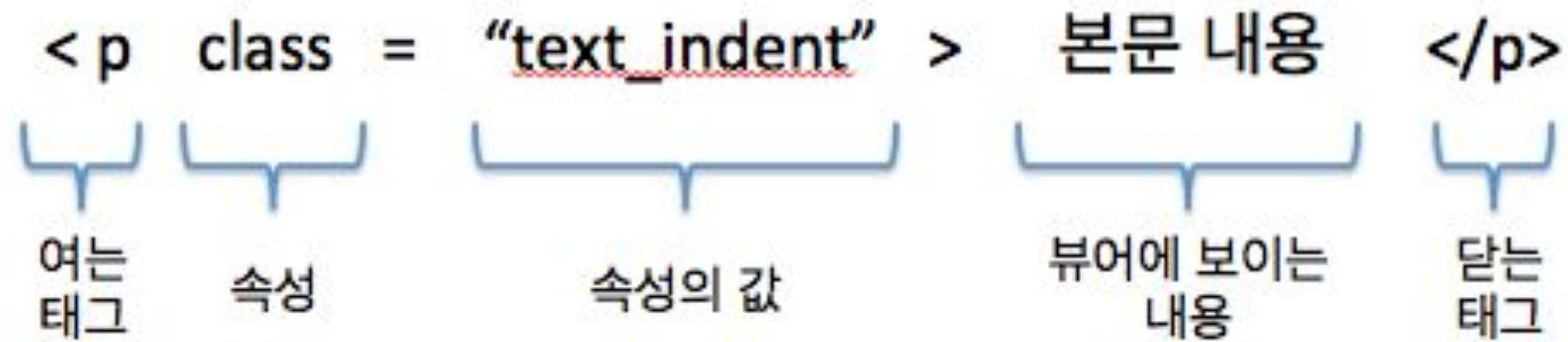


사진 출처: <https://www.epubguide.net/115>

HTML에 대해 알아보아요!

위키 제작에 필요한 핵심 요소

1. 텍스트 입력

- 제목: <h1> ~ <h6>
- 단락: <p>
- 줄바꿈:

- 가로줄: <hr>
- 강조:

```
<h1>제목 h1</h1>
<h2>제목 h2</h2>
<h3>제목 h3</h3>
<h4>제목 h4</h4>
<h5>제목 h5</h5>
<h6>제목 h6</h6>
```

```
<p>종이에 글씨를 쓰기 위해서는 무엇이 가장 필요할까요?
  바로 글씨를 쓸 필기구입니다.
  또한 글씨의 크기나 색깔을 마음대로 바꿀 수도 있겠죠?
</p>
<p>여기서 종이가 웹 페이지라면
  필기구는 <strong>HTML</strong>이며, 글씨를 꾸며주는 도구는 <strong>CSS</strong>입니다.
  <br>
  HTML은 웹 사이트를 구성하는 기본 언어이며,
  CSS는 HTML로 만들어진 웹사이트를 꾸며주는 언어입니다.
  위키라는 웹 사이트를 만들기 위해 꼭 필요한 존재입니다!
</p>

<hr>
```

제목 h1

제목 h2

제목 h3

제목 h4

제목 h5

제목 h6

종이에 글씨를 쓰기 위해서는 무엇이 가장 필요할까요? 바로 글씨를 쓸 필기구입니다. 또한 글씨의 크기나 색깔을 마음대로 바꿀 수도 있겠죠?

여기서 종이가 웹 페이지라면 필기구는 **HTML**이며, 글씨를 꾸며주는 도구는 **CSS**입니다.

HTML은 웹 사이트를 구성하는 기본 언어이며, CSS는 HTML로 만들어진 웹사이트를 꾸며주는 언어입니다. 위키라는 웹 사이트를 만들기 위해 꼭 필요한 존재입니다!

HTML에 대해 알아보아요!

위키 제작에 필요한 핵심 요소

2. 목록

- 순서 있는 목록
- 순서 없는 목록
 - 하위 요소

```
<ul>
  <li>순서 있는 목록</li>
  <li>순서 있는 목록</li>
  <li>순서 있는 목록</li>
</ul>
<ol>
  <li>순서 없는 목록</li>
  <li>순서 없는 목록</li>
  <li>순서 없는 목록</li>
</ol>
```

- 순서 있는 목록
- 순서 있는 목록
- 순서 있는 목록

1. 순서 없는 목록
2. 순서 없는 목록
3. 순서 없는 목록

HTML에 대해 알아보아요!

위키 제작에 필요한 핵심 요소

3. 표 <table>

- 행 <tr>
- 제목 <th>
- 열 <td>
- 합치기
 - 위아래 합치기: rowspan
 - 좌우 합치기: colspan

```
<table border="1">
  <tr>
    <th>이름</th>
    <th>학교</th>
    <th>학년</th>
  </tr>
  <tr>
    <td>김ㅇㅇ</td>
    <td>키위중학교</td>
    <td>3학년</td>
  </tr>
  <tr>
    <td>이ㅁㅁ</td>
    <td>골드여자고등학교</td>
    <td>2학년</td>
  </tr>
  <tr>
    <td>박ㅎㅎ</td>
    <td>설리번고등학교</td>
    <td>1학년</td>
  </tr>
</table>
```

이름	학교	학년
김ㅇㅇ	키위중학교	3학년
이ㅁㅁ	골드여자고등학교	2학년
박ㅎㅎ	설리번고등학교	1학년

HTML에 대해 알아보아요!

위키 제작에 필요한 핵심 요소

4. 하이퍼링크 <a>

- 다른 사이트로 이동 (a href)
- 페이지 내 특정 위치로 이동 (id)

```
<a href="https://www.naver.com">네이버</a>  
<a href="htmlStart.html">실습 홈페이지</a>
```

```
<a href="#sullivan">시작점</a>  
<a id="sullivan">목적지</a>
```


2차시 - HTML과 CSS를 알아보아요

HTML에 대해 알아보아요!

위키 제작에 필요한 핵심 요소

5. 이미지 삽입

```

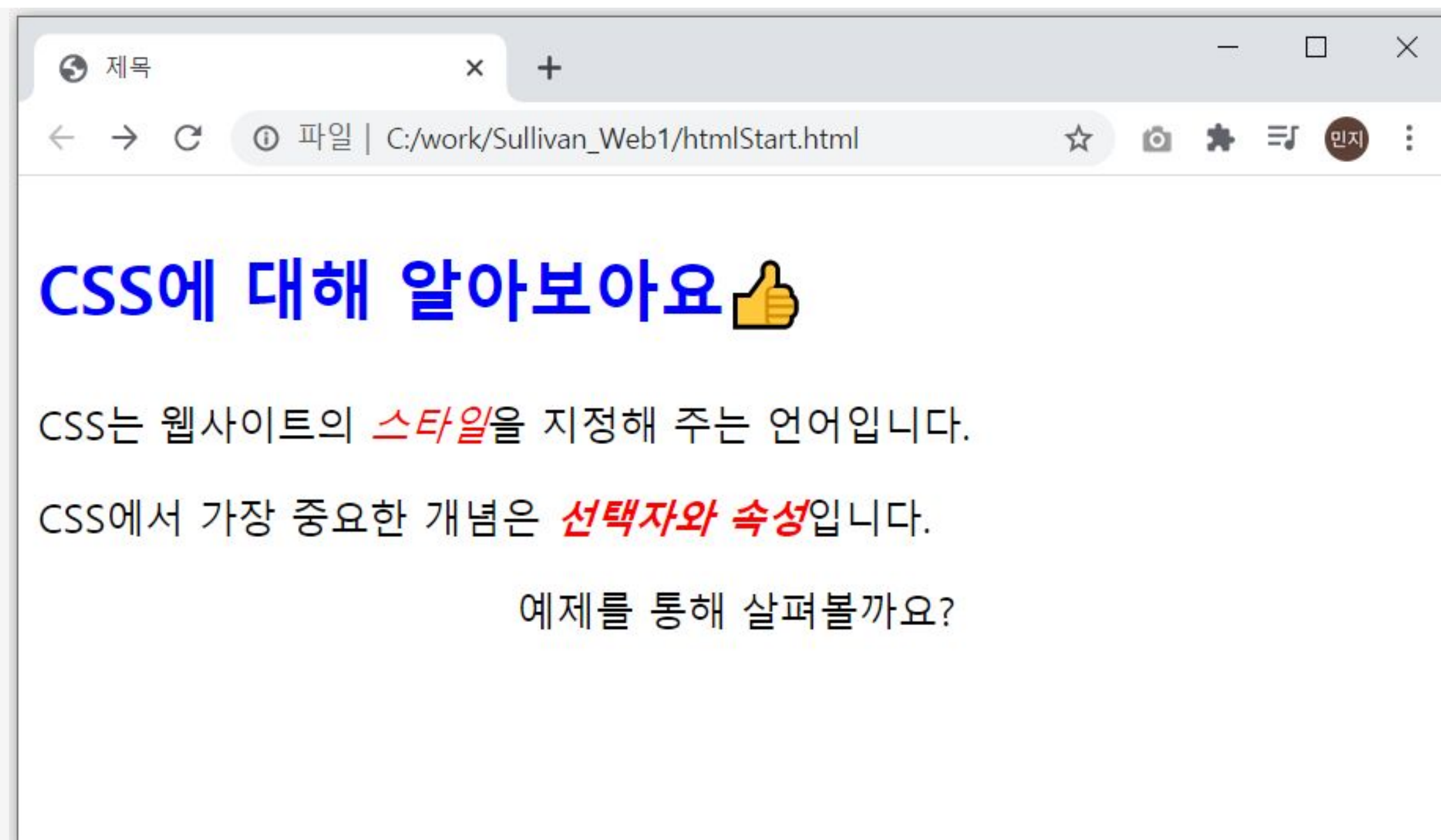
```

2차시 - HTML과 CSS를 알아보아요

CSS에 대해 알아보아요!

글자 크기와 색상을 어떻게 바꾸었을까요?
힌트: **style**으로 시작하는 부분에 집중!

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>제목</title>
    <style type="text/css">
      h3 {font-size: 20pt; color: blue;}
      .style {color: red; font-style: italic;}
      #example {text-align: center;}
    </style>
  </head>
  <body>
    <h3>CSS에 대해 알아보아요👍</h3>
    <p>CSS는 웹사이트의 <span class="style">스타일</span>을 지정해 주는 언어입니다.</p>
    <p>CSS에서 가장 중요한 개념은 <strong class="style">선택자와 속성</strong>입니다.</p>
    <p id="example">예제를 통해 살펴볼까요?</p>
  </body>
</html>
```



CSS에 대해 알아보아요!

CSS는 웹사이트의 스타일을 지정해 주는 언어입니다.

즉, **HTML**의 태그(요소)에 스타일 효과를 주는 언어입니다.

CSS를 사용하기 위해서는 **HTML**처럼 직접 선언을 해주어야 합니다.

- 내부에서 선언 (<head> 태그 안)

<style type="text/css"> ... </style>

- 외부 스타일시트 연결

<link rel="stylesheet" type="text/css" href="외부 스타일시트 주소">

CSS에 대해 알아보아요!

CSS의 기본 문법은 선택자 {속성:값; 속성:값;...} 입니다.



여기서 **선택자**는 무엇일까요?

우리가 효과를 주고 싶은 태그를 선택해야 하는데,
이 때 꼭 알아야 하는 중요한 개념입니다.

CSS에 대해 알아보아요!

선택자는 어떤 요소에 스타일을 적용할지 알려주는 방식입니다. 선택자의 유형에는 여러 가지가 있어요. 각각의 유형에 대해 잘 알아두면, 더욱 편리하게 스타일을 설정할 수 있습니다.

- **태그** 선택자: HTML에서 정의되는 태그를 선택하여 스타일 지정
- **클래스** 선택자: 어떠한 태그든 선언한 클래스명만 적용해 주면 해당 스타일이 적용됨
- **아이디** 선택자: 클래스 선택자와 유사하지만 문서 내에서 한 번만 사용할 수 있음

CSS에 대해 알아보아요!

1. 태그 선택자 `p { color: blue; margin: 5px; }`

선택자 속성 속성값

2. 클래스 선택자

`<p class="class_name">` `.class_name { color: blue; margin: 5px; }`

태그선택자 클래스 선언 클래스명 클래스선택자 속성 속성값

3. 아이디 선택자

`<p id="id_name">` `#id_name { color: blue; margin: 5px; }`

id 선언 id명 id선택자 속성 속성값

CSS에 대해 알아보아요!

내용이 여기를 넘지
않도록
주의해주세요!

위키 제작에 필요한 핵심 CSS 속성을 알아보아요!

- 폰트: font-size, color, text-align, font-family, text-decoration
- 배경: background-color, background-image
- 목록: list-style-type
- 표: width, table-layout
- 박스 모델: margin, padding, border, display
- 콘텐츠: width, height, top, right, bottom, left, float

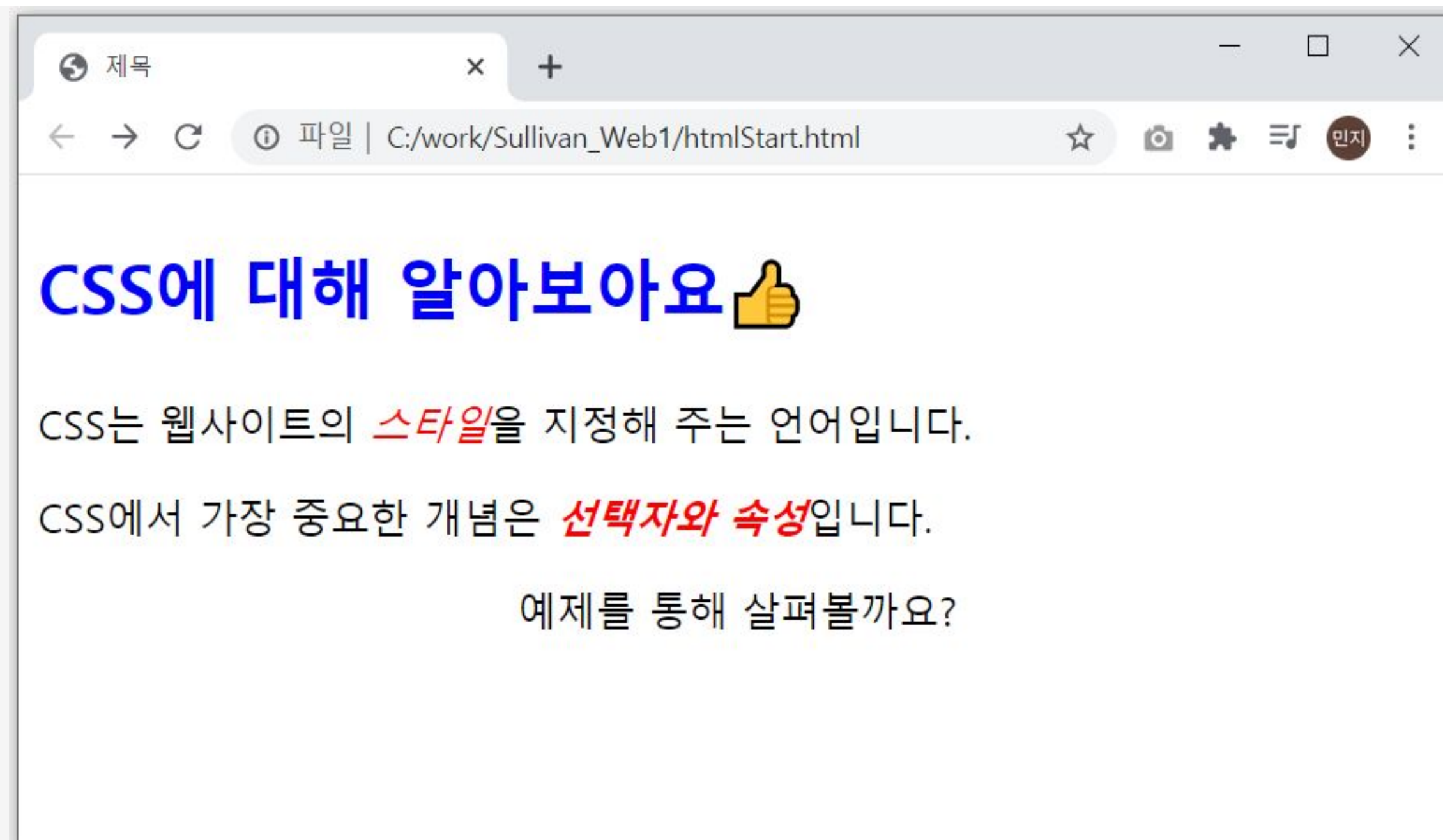
* 한눈에 보기 쉽게 표로 정리할 예정입니다.

2차시 - HTML과 CSS를 알아보아요

CSS에 대해 알아보아요!

조금 전에 보여드렸던 사진을 다시 살펴봅시다.
어떤 선택자와 속성을 사용했는지 알아보아요!

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>제목</title>
    <style type="text/css">
      h3 {font-size: 20pt; color: blue;}
      .style {color: red; font-style: italic;}
      #example {text-align: center;}
    </style>
  </head>
  <body>
    <h3>CSS에 대해 알아보아요👍</h3>
    <p>CSS는 웹사이트의 <span class="style">스타일</span>을 지정해 주는 언어입니다.</p>
    <p>CSS에서 가장 중요한 개념은 <strong class="style">선택자와 속성</strong>입니다.</p>
    <p id="example">예제를 통해 살펴볼까요?</p>
  </body>
</html>
```



지금까지 배운 내용을 복습해 볼까요?

자기소개 페이지 만들기 🖐️✂️

2차시 - HTML과 CSS를 알아보아요

실전! 자기소개 페이지 만들기

지금까지 배운 HTML과 CSS의 개념을
활용하여 자기소개 페이지를 만들어
보아요!

내용과 스타일 설정은 자유입니다!
어렵다면 옆에 있는 예시를 참고해도
좋아요 😊

자기소개

프로필



1. 이름: 김민지(KIM MINJI)
2. 나이: 21세
3. 학교: 골드키위대학교 2학년

활동 이력

시기	활동명
2014.05 -	네이버 블로그 운영
2019.03 - 2020.02	인액터스 14기 홍보부 팀장
2019.09	2019-1학기 성적우수장학금
2019.11	교내 Citizenship Fair 장려상
2020.03 - 2020.08	IT공학전공 학회
2020.03 - 2020.09	경력개발처 소속 봉사 동아리 부원

나의 성향

- Holland 직업적성검사: C(관습형), I(탐구형)
- MBTI 유형: **ISFJ**

앞으로의 목표

- 전공 공부를 열심히 하여 매학기 학점 3.5 이상 얻기
- 관심 있는 활동이라면 주저하지 않고 지원하기
- 글 쓰고 말하는 실력 키우기 - 한 달에 2권 독서, 발표 동아리 들어가기, 매일 뉴스 보기

나만의 학교 위키 만들기

3차시 - Django를 알아 보아요 🙌

설리번 프로젝트 Web1팀 — 발표 관련 정보 (일시, 자료 등)

목차

1. Django & Framework 이해하기

- a. Django
- b. Framework

2. MTV 패턴이란?

3. 가상환경 & PIP 이해하기

- a. 가상환경
- b. PIP

4. Django 프로젝트 시작하기

- a. 가상환경 생성
- b. Django 설치
- c. 프로젝트 생성
- d. App 만들기

5. [실습] Django에서 Home 출력하기

Django & Framework

Django는 Python으로 작성된

오픈 소스 웹 어플리케이션 프레임워크입니다.

인스타그램, 핀터레스트 등 다양한 곳에서 **Django**를 사용하고 있습니다.

Django의 특징: Python 기반, **MTV 패턴**, Admin 기능 등

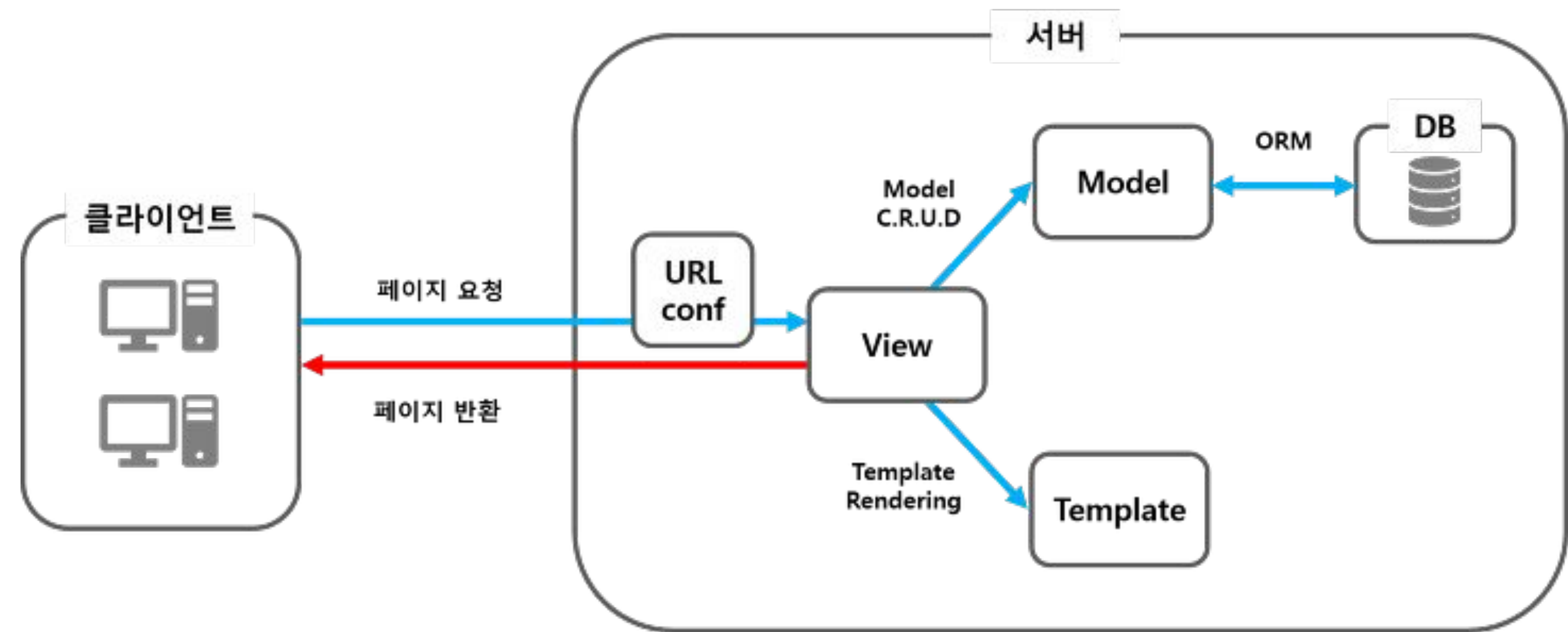
Framework란?

기본적으로 필요한 기능을 갖춰, 원하는 기능 구현에 집중하도록 도와주는 개발 환경입니다.

MTV 패턴이란?

Model - Template - View

- URLconf: URL 경로에 맞춰 View와 Template 연결
- **Template**: 사용자에게 보여지는 환경
- **View**: 웹 요청을 받고, 전달받은 데이터를 처리해서 가공
- **Model**: 데이터베이스에 저장되는 데이터



3차시 - Django를 알아보아요

가상환경 & PIP

<가상환경>

자신이 원하는 Python 환경 구축을 위해 필요한 모듈만
담아 놓는 바구니

<PIP>

Python으로 작성된 패키지 소프트웨어를 관리하는
패키지 관리 시스템

내용이 여기를 넘지
않도록
주의해주세요!

지금까지 배운 내용을 통해
프로젝트를 생성해볼까요?

Django 프로젝트 시작하기 🖋️

3차시 - Django를 알아보아요

Django 프로젝트 시작하기

Django 프로젝트를 시작하려면 아래와 같은 절차를 따라야 해요.

1. 가상환경 생성
2. Django 설치
3. 프로젝트 생성
4. App 만들기

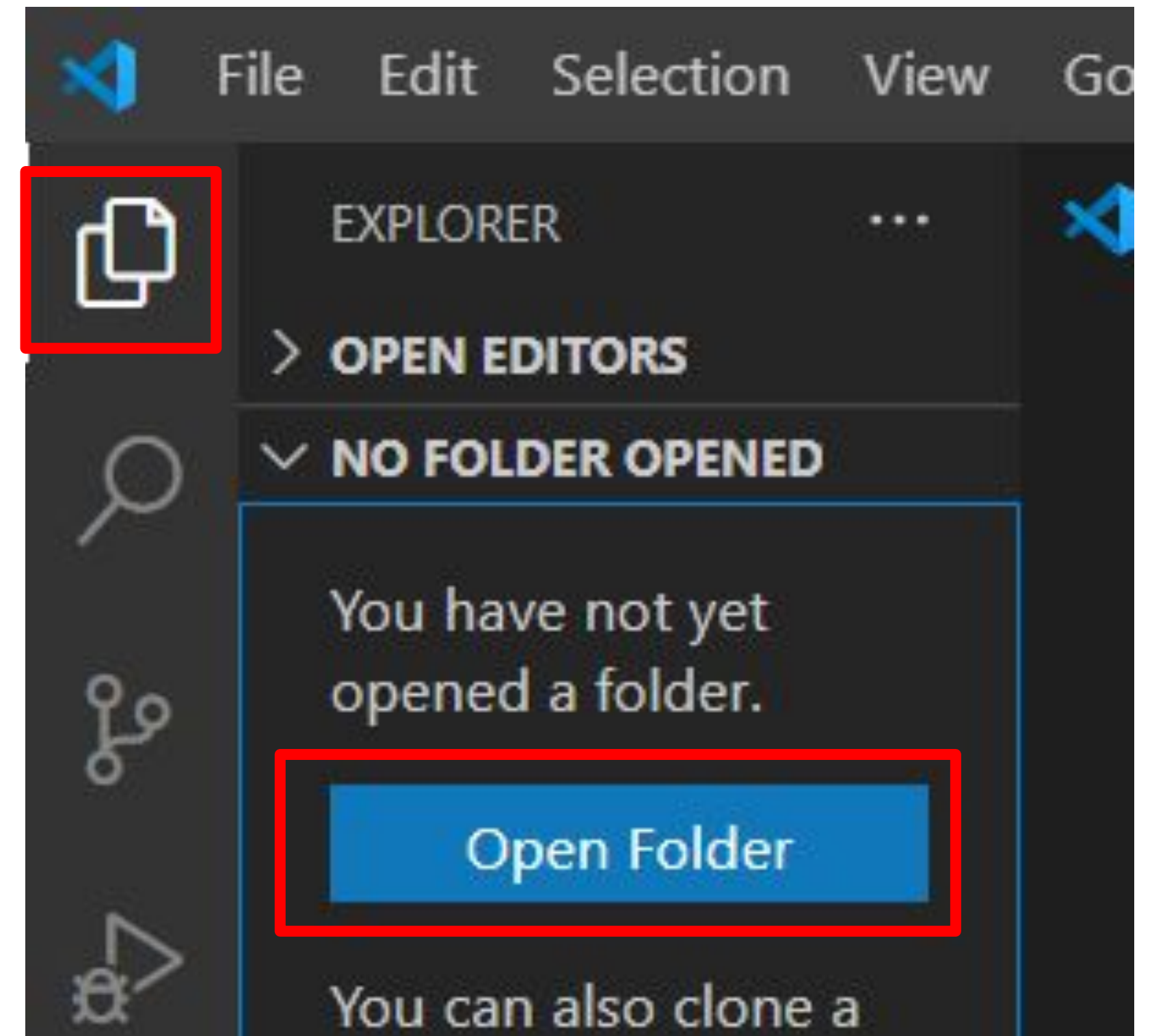
조금 어렵더라도 차근차근 따라오면 충분히 만들 수 있어요!

Django 프로젝트 시작하기

0. 준비하기

vscode를 실행하고 Open Folder를 눌러
프로젝트를 진행할 빈 폴더를
불러옵니다.

Ctrl + Shift + ` 을 눌러 터미널 창을
열어줍니다.



Django 프로젝트 시작하기

1. 가상환경 생성

```
python -m venv myvenv
```

“myvenv”라는 이름의 가상환경을 만들어봅시다.

```
source myvenv/Scripts/activate
```

 (Windows)

```
source myvenv/bin/activate
```

 (Mac)

위의 명령어를 통해 가상환경을 활성화합니다.

3차시 - Django를 알아보아요

Django 프로젝트 시작하기

2. Django 설치

```
pip install django
```

가상환경을 활성화한 상태에서 Django를 설치해줍니다.

```
$ pip install django
Collecting django
  Downloading https://files.pythonhosted.org/packages/c6/b7/63d23df1e311ca0d90f41352a9efe7389ba353df95deea5676652e615420/Django-3.0.3-py3-none-any.whl (7.5MB)
    |████████████████████| 7.5MB 939kB/s
Collecting asgiref~=3.2 (from django)
  Downloading https://files.pythonhosted.org/packages/a5/cb/5a235b605a9753ebcb2730c75e610fb51c8cab3f01230080a8229fa36adb/asgiref-3.2.3-py2.py3-none-any.whl
Collecting pytz (from django)
  Downloading https://files.pythonhosted.org/packages/e7/f9/f0b53f88060247251bf481fa6ea62cd0d25bf1b11a87888e53ce5b7c8ad2/pytz-2019.3-py2.py3-none-any.whl (509kB)
    |████████████████████| 512kB 1.1MB/s
Collecting sqlparse>=0.2.2 (from django)
  Downloading https://files.pythonhosted.org/packages/ef/53/900f7d2a54557c6a37886585a91336520e5539e3ae2423ff1102daf4f3a7/sqlparse-0.3.0-py2.py3-none-any.whl
Installing collected packages: asgiref, pytz, sqlparse, django
Successfully installed asgiref-3.2.3 django-3.0.3 pytz-2019.3 sqlparse-0.3.0
```

3차시 - Django를 알아보아요

Django 프로젝트 시작하기

3. 프로젝트 생성

```
django-admin startproject mywiki
```

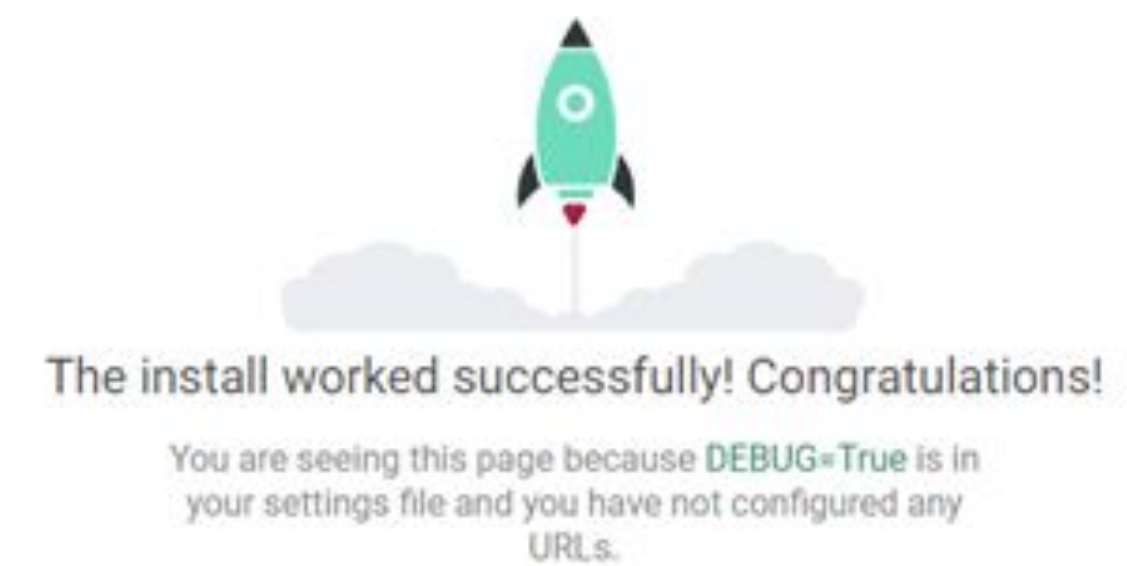
“mywiki”라는 이름의 프로젝트를 만듭니다.
(이름은 임의로 설정하였습니다)

```
cd mywiki
```

“mywiki” 프로젝트 폴더로 이동해준 후

```
python manage.py runserver
```

서버를 실행해 봅시다.



 **Django Documentation**
Topics, references, & how-to's

 **Tutorial: A Polling App**
Get started with Django

 **Django Community**
Connect, get help, or contribute

<http://127.0.0.1:8000> 을 실행했을 때
위와 같은 화면이 뜬다면
Django 프로젝트가 성공적으로 만들어졌다는 뜻입니다.

서버 실행을 종료하려면 **Ctrl+C** 를 입력해 주세요!

Django 프로젝트 시작하기

4. App 만들기

```
python manage.py startapp page
```

“page”라는 이름의 App을 만듭니다.
(이름은 임의로 설정하였습니다)

< Project와 App의 관계 >

- 하나의 프로젝트 = 하나의 웹 사이트
- 프로젝트 안의 기능들을 각각의 App으로 관리

Django에서 Home 출력하기

1. settings.py

- 프로젝트에 App의 존재 알려주기

2. Templates

- User에게 보여줄 HTML 파일 만들기

3. views.py

- 요청이 들어오면 HTML 파일 열어주는
함수 정의

4. urls.py

- url 요청을 views와 연결하기

사용자 정의 함수 예제

Django 프로젝트 실습

1. App 만든 것을 알려주기

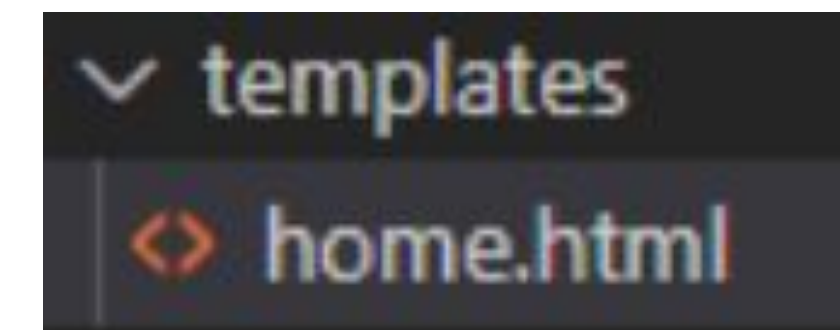
settings.py의 INSTALLED_APPS에 생성한 App을 알려줍니다.(예시: App 이름 → page)

`'page.apps.PageConfig',`

list 형식으로 받기 때문에 뒤에 ,를 반드시 붙여주어야 합니다.

2. home.html 만들기

앱 폴더 안에 templates 폴더를 만들고 그 안에 home.html 파일을 만듭니다.



Django 프로젝트 실습

3. HTML 파일을 열어주는 함수 정의하기

```
def home(request):  
    return render(request, 'home.html')
```

request를 통해 요청이 들어오면 home.html 파일을 반환합니다.

4. URL에 경로 추가해주기

```
path("", views.home, name = 'home'),
```

path를 통해 어떤 주소로, 어떤 함수를 실행하고 어떻게 관리할지를 정합니다.

```
from . import views
```

urls.py와 views.py의 경로를 연결하기 위해 import 해야합니다.

나만의 학교 위키 만들기

4차시 - Django 기본 개념 익히기

설리번 프로젝트 Web1팀 — 발표 관련 정보 (일시, 자료 등)

목차

1. URL & Template

- a. URL
- b. URL Include
- c. Template
- d. Template 상속

2. Static & Media

- a. Static
- b. Media
- c. 실습

3. Model

- a. Model 이란?
- b. 실습
- c. Model과 Database 연동
- d. Admin 확인

3주차 과제 리뷰

내가 만들고 싶은 위키에 대해 발표하기

만들고 싶은 웹페이지 구성이나 기능에 대해 얘기해보아요 😊
발표는 부담 갖지 말고 짧게 **1~2분 내외**로 해주시면 됩니다.

내용이 여기를 넘지
않도록
주의해주세요!

URL & Template

< URL >

웹사이트를 방문할 때마다 브라우저의 주소창에서 볼 수 있음.

즉, URL은 웹주소이다. eg> 127.0.0.1:8000

→ url include를 사용하면, 고정적인 url을 쉽게 관리하거나 앱별로 url을 관리할 수 있다.

```
from django.contrib import admin
from django.urls import path, include
from page import views
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('page.urls')),
] + static(settings.MEDIA_URL, document_root = settings.MEDIA_ROOT)
```

< Template >

파이썬을 HTML로 바꿔주어, 빠르고 쉽게 동적인 웹 사이트를 만들 수 있게 도와줌.

-> 템플릿 상속이란, 말 그대로 html 문서 중 기본 뼈대가 되는 문서를 기본

템플릿으로 정하고, 이는 공통의 코드이므로 다른 문서에서 기본 템플릿의 코드가 필요하면 상속하여 가져다 쓸 수 있다.

```
{% extends 'base.html' %}
```

Static & Media

< Static >

개발 리소스로서의 정적인 파일 (css, js, image 등)

-> static 파일은 `setting.STATICFILES_DIRS`에 지정된 경로에 두게 됨.

-> 다수의 디렉토리에 저장된 static 파일은 `collectstatic` 명령을 통해서 `setting.STATIC_ROOT`에 지정한 경로로 모아서 서비스에 사용됨.

< Media >

유저가 업로드 한 모든 정적인 파일 (image, pdf 등)

-> 프로젝트 단위로 저장/서빙

Static & Media 실습

1. static 폴더, 파일 생성

- App 폴더 내에 static 폴더 만들기
- static 폴더 내에 css, images 폴더 만들기
- css 폴더 안에 home.css 등 html파일의 css 파일 만들기
- images 폴더 안에 자주 사용할 logo.png 등의 파일 넣기

2. static 설정

settings.py 폴더 맨 하단에 경로 지정

```
STATICFILES_DIRS = [
```

```
    os.path.join(BASE_DIR, 'page', 'static')
```

```
]
```

```
STATIC_ROOT = os.path.join(BASE_DIR, 'static')
```

4차시 - Django 기본 개념 익히기

Static & Media 실습

3. static 파일 모으기

```
python manage.py collectstatic
```

4. Media 설정

```
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

```
MEDIA_URL = '/media/'
```

5. urls.py에서 import

```
from django.conf import settings
```

```
from django.conf.urls.static import static
```

```
urlpatterns = [
```

```
] + static(settings.MEDIA_URL, document_root = settings.MEDIA_ROOT)
```


Static & Media 실습

5. urls.py에서 import

```
from django.conf import settings
```

```
from django.conf.urls.static import static
```

```
urlpatterns = [
```

```
] + static(settings.MEDIA_URL, document_root = settings.MEDIA_ROOT)
```

6. HTML에서 사용

HTML 파일 내에서 상단에 아래 코드를 사용합니다.

```
{% load static %}
```

Model

< Model이란? >

뷰(view) 함수에서 데이터베이스에 어떤 작업을 요청할 때는 **SQL** 구문이 필요
즉, **DB** 테이블과 매핑
데이터에 접속하고 관리하도록 도와주는 객체

Model 실습

1. models.py에 모델 생성

```
class Post(models.Model):
```

```
    title = models.CharField(max_length=50, default='')
```

```
    update_date = models.DateTimeField(default=timezone.now)
```

```
    text = models.TextField()
```

```
    image = models.ImageField(upload_to = 'images/', blank = True, null = True)
```

```
    def __str__(self):
```

```
        return self.title
```

Model 실습

2. Model과 Database 연동

DB가 이해할 수 있도록 번역해줍니다.

```
python manage.py makemigrations
```

번역한 내용을 DB에 적용시켜줍니다.

```
python manage.py migrate
```


Admin 확인

웹 서비스 관리를 위한 admin 기능을 사용해봅시다.

```
python manage.py createsuperuser
```

터미널창에 위의 명령어를 입력해 관리자 계정을 생성하고,
실행된 url에 /admin을 입력해 접속합니다.

```
from django.contrib import admin
from django.urls import path, include
from page import views
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('page.urls')),
] + static(settings.MEDIA_URL, document_root = settings.MEDIA_ROOT)
```

Admin에게 Model 알려주기

admin.py 파일에 아래의 코드를 입력해줍니다.

```
from .models import Post
```

```
admin.site.register(Post)
```

Python을 이용하여 위키 기능 구현하기

자 이제 본격적으로 위키 만들기 시작~!

나만의 학교 위키 만들기

5차시 - Django 심화

설리번 프로젝트 Web1팀 — 발표 관련 정보 (일시, 자료 등)

목차

1. QuerySet & Method

a. QuerySet

b. Method

2. PK, Path Converter, get_object_or_404

a. PK

b. Path Convertor

c. get_object_or_404

3. Detail 페이지 구현

a. 자세히 보기 기능

b. +a 기능

4. [실습] 위키 구현하기

QuerySet & Method

< QuerySet >

Database에서 전달받은 객체들의 모음(list)

* DB(SQL)에서는 row에 해당한다.

< Method >

객체 지향 프로그래밍 언어에서 클래스 내부에서 정의된 함수

```
from django.shortcuts import render, redirect, get_object_or_404
from .models import Post

# Create your views here.
def home(request):
    posts = Post.objects.all()
    return render(request, 'home.html', {'posts':posts})
```

→ posts에 objects 메소드를 사용하여 모델의 객체 불러오고, 이때 전달받은 객체를 **쿼리셋**이라고 함.

```
{% for post in posts %}
<article class="blog-post">
  <h2 class="blog-post-title">{{ post.title }}
  <div class="btn-group">
    <a href = "{% url 'update' post.id %}" class = "btn btn-sm btn-outli
    <a href = "{% url 'delete' post.id %}" class = "btn btn-sm btn-outli
  </div>
</h2>
<p class="blog-post-meta">{{ post.update_date|date:"Y/m/d" }}</p>
<span>{{ post.text|truncatechars:20 }}<a href="{% url 'detail' post.id %}">.
```

→ posts 안에 있는 목록의 모든 객체를 반복.
쿼리셋 메소드 사용하여 특별한 기능 수행

PK, Path Converter, get_object_or_404

< PK : Primary Key >

Model을 통해 생성된 객체를 구분해주는 역할
각 상황에 몇 번째 블로그 객체를 호출하는지 알기 위함.

< Path Converter >

그 상황에 맞는 url은 .../blog/{객체번호}처럼 객체에 따라서 달라짐.
여러 객체의 url을 효율적으로 다룰 수 있도록 도와주는 도구

< get_object_or_404 >

존재하지 않는 객체 번호를 호출할 경우, 에러 페이지를 호출.

Detail 페이지 구현

게시글을 자세히 보는 기능을
만들어봅시다!

→ 스스로 해보다 막히면 언제든지
편하게 선생님들께 질문해주세요 ✨

```
def detail(request, post_id):  
    post = get_object_or_404(Post, pk = post_id)  
    return render(request, 'detail.html', {'post':post})
```

5차시 - Django 심화

+a 기능

가장 많은 학생들이 원했던 oo
기능을 구현해봅시다!

→ 스스로 해보다 막히면 언제든지
편하게 선생님들께 질문해주세요 ✨

예시 코드 이미지

위키 구현하기

우리학교 위키 페이지 만들기

- 4주차에 구현한 기능을 바탕으로 만들어보아요!
- 2주차에 배웠던 **HTML, CSS**를 이용해 예쁜 위키 페이지를 만들어봅시다!

→ 설리번 프로젝트는 스스로 구현하는 것을 목표로 하고 있지만, 충분한 시도에도 구현이 어렵다면 언제든지 편하게 선생님들께 질문해주세요 😊

내용이 여기를 넘지
않도록
주의해주세요!

나만의 학교 위키 만들기

6차시 - 학교위키를 완성하고 공유해요👍

설리번 프로젝트 Web1팀 — 발표 관련 정보 (일시, 자료 등)

목차

1. CRUD

a. CRUD란?

2. GET & POST

a. GET

b. POST

3. CRUD 실습

a. 추가 기능 (Create)

b. 삭제 기능 (Delete)

c. 수정 기능 (Update)

4. 학교 위키 만들기 마무리

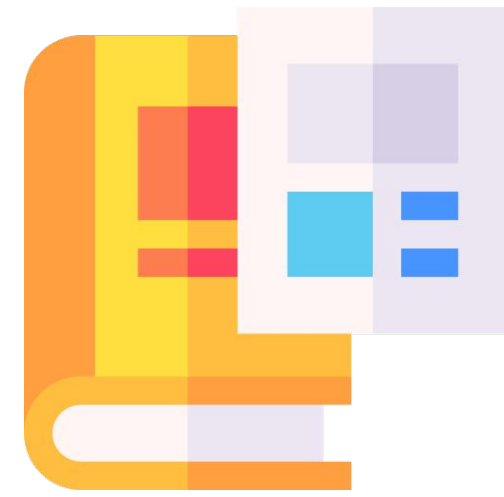
a. Project & App

6차시 - 완성한 위키를 배포해요

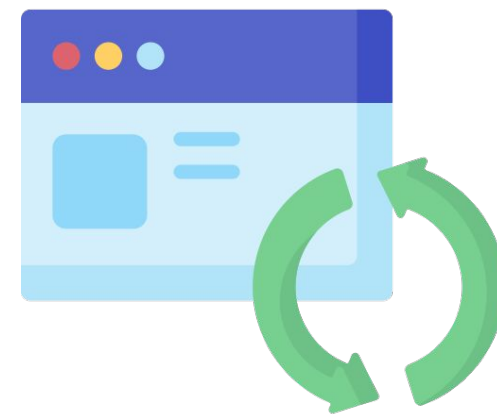
CRUD



Create



Read



Update



Delete

사용자 인터페이스가 **갖추어야 할 기능**(정보의 참조/검색/갱신)
을 가리키는 용어로서 사용

GET & POST

< GET >

서버로부터 정보를 조회하기 위해 설계된 메소드

요청을 전송할 때 필요한 데이터를 **Body**에 담지 않고 *쿼리스트링을 통해 전송.

* URL의 끝에 ?와 함께 이름과 값으로 쌍을 이루는 요청 파라미터를 쿼리스트링이라고 부름

< POST >

리소스를 생성/변경하기 위해 설계된 메소드

요청을 전송할 때 필요한 데이터를 **Body**에 담아 전송. 또한 데이터 내용이 눈에 보이지 않아 GET보다 보안적인 면에서 안전.

서버에게 동일한 요청을 여러 번 전송하더라도

→ **GET**은 동일한 응답이 돌아온다.

→ **POST**는 응답이 항상 다를 수 있다.

추가 기능 (Create)

게시글을 추가하는 기능을
만들어봅시다!

→ 스스로 해보다 막히면 언제든지
편하게 선생님들께 질문해주세요 ✨

```
def create(request):  
    if request.method == 'POST':  
        post = Post()  
        if 'image' in request.FILES:  
            post.image = request.FILES['image']  
        post.title = request.POST['title']  
        post.text = request.POST['text']  
  
        post.save()  
  
        return redirect('detail', post.id)
```


삭제 기능 (Delete)

게시글을 삭제하는 기능을
만들어봅시다!

→ 스스로 해보다 막히면 언제든지
편하게 선생님들께 질문해주세요 ✨

```
def delete(request, post_id):  
    post = get_object_or_404(Post, pk = post_id)  
    post.delete()  
    return redirect('home')
```

CRUD 실습

수정 기능 (Update)

게시글을 수정하는 기능을
만들어봅시다!

→ 스스로 해보다 막히면 언제든지
편하게 선생님들께 질문해주세요 ✨

```
def update(request, post_id):  
    post = get_object_or_404(Post, pk = post_id)  
  
    if request.method == 'POST':  
        if 'image' in request.FILES:  
            post.image = request.FILES['image']  
        post.title = request.POST['title']  
        post.text = request.POST['text']  
  
        post.save()  
        return redirect('detail', post.id)  
    else:  
        return render(request, 'update.html', {'post': post})
```

우리학교 위키 페이지 완성하기

- 저번주에 만들었던 위키에서 더 발전시켜보아요!
- 학교 이미지나 정보를 넣어서 완성도 있는 위키를 만들어도 좋습니다.

→ 설리번 프로젝트는 스스로 구현하는 것을 목표로 하고 있지만, 충분한 시도에도 구현이 어렵다면 언제든지 편하게 선생님들께 질문해주세요 😊

내용이 여기를 넘지
않도록
주의해주세요!

내가 만든 우리학교 위키 발표하기

내가 만든 위키를 발표하고
다른 친구들 작품도 구경해보아요👀

나만의 학교 위키 만들기를 마무리하며

6주 동안 수고하셨습니다~!! 😄

감사합니다 🍷

개인 관련 정보 (이메일, 페이스북 등)

contact@sullivanproject.io

SULLIVAN PROJECT