Django Relationship Fields

2021130058 한예진

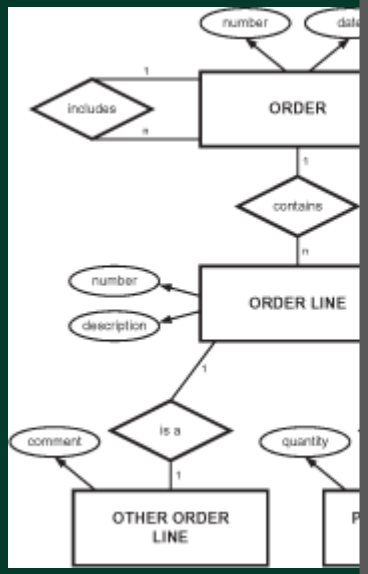
Field?

✓ 데이터베이스의 구조(model)에 속하는 데이터베이스 테이블의 열(column)

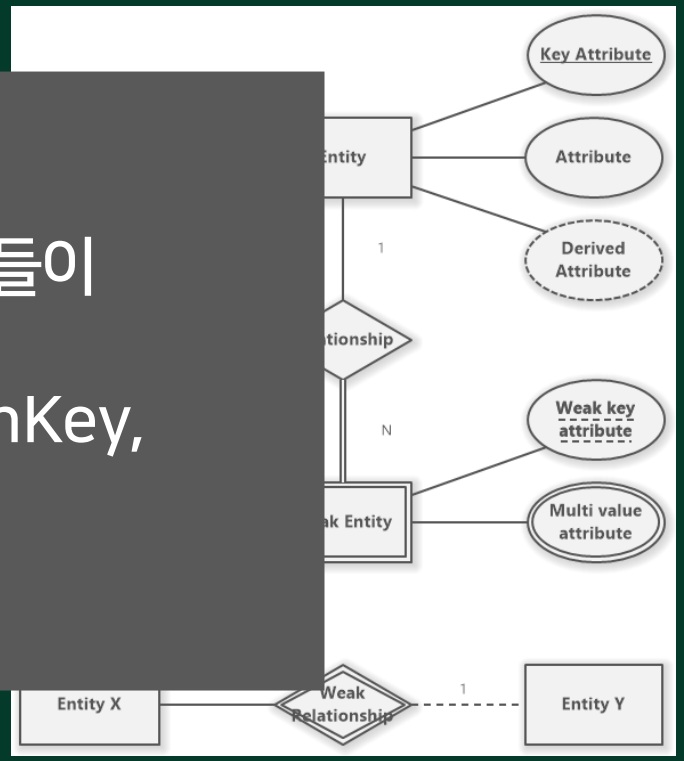
✓ 필드를 통해 저장되는 것:

- ✓ 테이블의 열 저장할 때 데이터의 유형 (문자열, 정수 등)
- ✓ 폼 필드를 렌더링할 때 사용할 기본 HTML 위젯
- ✓ 관리자 페이지와 자동 생성되는 폼들에 적용될 최소 요구 사항

Field?



이렇게 관계를 나타내는 Fields들이
Relationship Fields!
→ManyToManyField, ForeignKey,
OneToOneField



세션 복습

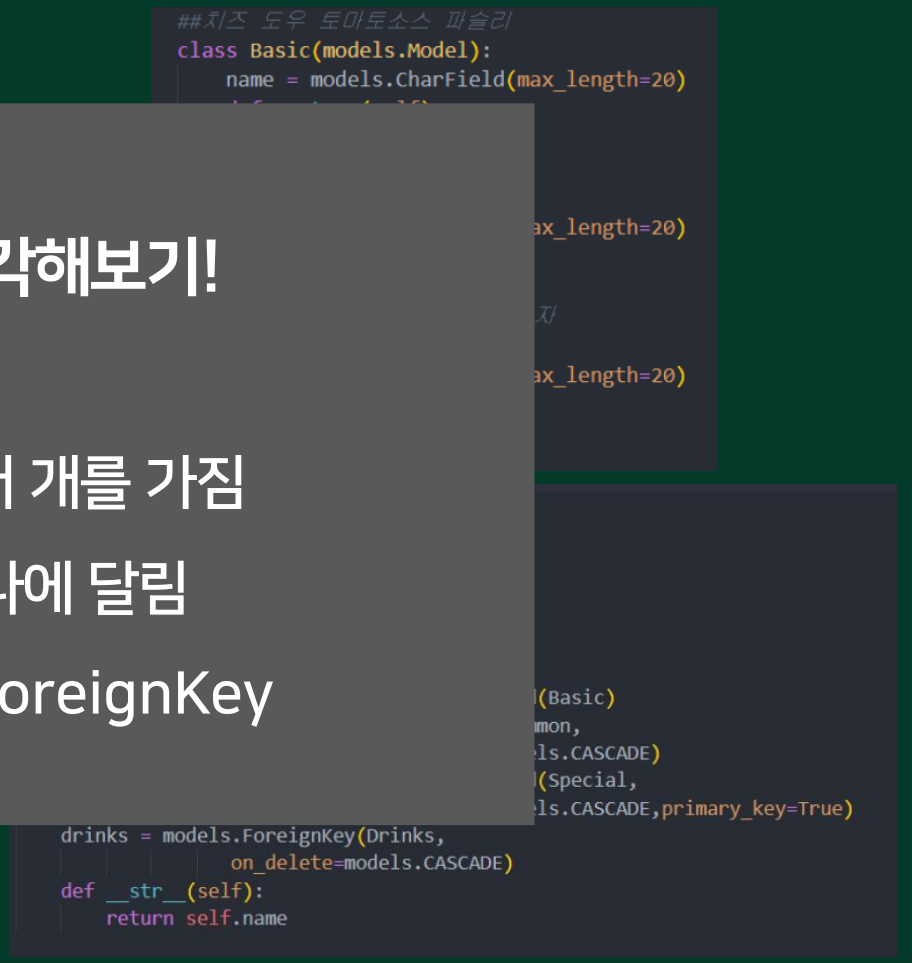
- ✓ ManyToMan
  - ex. 피자-필수토
- ✓ ForeignKey
  - ex. 피자-선택토
- ✓ OneToOneF
  - ex. 피자-메인토

서로의 입장에서 생각해보기!

Ex) 게시글은 댓글 여러 개를 가짐

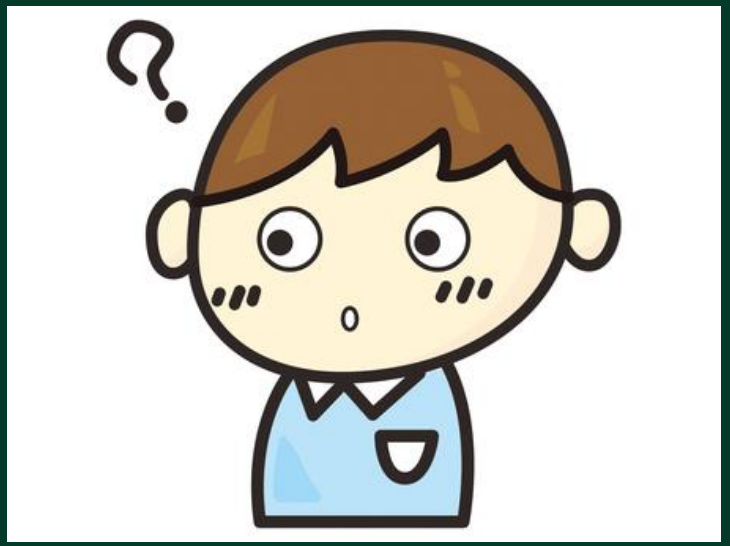
\+ 댓글은 게시글 하나에 달림

-> 일대다 관계이므로 ForeignKey



세션 복습

```python
class StudentId(models.Model):
    student_num = models.CharField(max_length=20)

    def __str__(self):
        return self.student_num

class Professor(models.Model):
    prof_name = models.CharField(max_length=50)

    def __str__(self):
        return self.prof_name

class Lecture(models.Model):
    Lec_name = models.CharField(max_length=50)
    Lec_prof = models.ForeignKey(Professor, on_delete=models.CASCADE)

    def __str__(self):
        return self.Lec_name

class Student(models.Model):
    name = models.CharField(max_length=20)
    student_id = models.OneToOneField(StudentId, on_delete=models.CASCADE)
    is_lecture = models.ManyToManyField(Lecture)

    def __str__(self):
        return self.name
```



ManyToManyField

특징 (1) 데이터를 가져올 때 _set을 붙이기

✓ 참조되는 테이블에서 참조하는 테이블의 데이터를 가져올 때,
참조하는 테이블 이름 뒤에 _set을 붙여줘야 함
ex. 한국어문법의이해.student_set.all()

✓ 또는 related_name을 추가해주기
ex. ls_lecture = models.ManyToManyField(Lecture, related_name=students)
→ 한국어문법의이해.students.all()

ManyToManyField

특징 (2) Through Model

✓ Through Model이란?

→ django에서 ManyToManyField로 데이터를 정의하면 자동으로 두 테이블의 관계를 관리해주는 테이블

```python
class ToppingAmount(models.Model):

    REGULAR = 1
    DOUBLE = 2
    TRIPLE = 3
    AMOUNT_CHOICES = (
        (REGULAR, 'Regular'),
        (DOUBLE, 'Double'),
        (TRIPLE, 'Triple'),
    )

    pizza = models.ForeignKey('Pizza', related_name='topping_amounts', on_delete=models.SET_NULL, null=True)
    topping = models.ForeignKey('Topping', related_name='topping_amounts', on_delete=models.SET_NULL, null=True, blank=True)
    amount = models.IntegerField(choices=AMOUNT_CHOICES, default=REGULAR)
```

ForeignKey

- ✓ ForeignKey는 모델 클래스, on_delete 인수 필요
- ✓ ForeignKey의 다양한 option들
  - ① Related_name
  - ② On_delete
  - ③ Limited_choices_to
  - ④ Related_query_name
  - ⑤ To_field
  - ⑥ db_constraint
  - ⑦ Swappable

ForeignKey

1) On_delete

CASCADE	ForeignKey를 포함하는 요소도 삭제
PROTECT	삭제하는 대상과 관계있는 요소가 함께 삭제되지 X
SET_NULL	ForeignKey 값을 NULL로 변경 (null=True일 때)
SET(function)	ForeignKey를 function으로 수행
SET_DEFALUT	ForeignKey를 기본 값으로 수행 (default 값이 필요)
DO_NOTHING	아무 것도 안함

Foreign Key

2) Limited_choices_to

✓ 필드가 렌더링될 때, 사용가능한 선택지 제한을 설정

✓ ManyToManyFieId에서도 사용 가능

```
staff_member = models.ForeignKey(
    User,
    on_delete=models.CASCADE,
    limit_choices_to={'is_staff': True},
)
```

OneToOneField

- ✓ Foreign Key와의 차이점: 역관계 (reverse relationship)
  - ✓ OneToOneField 모델의 역참조는 하나의 객체 반환
  - ✓ ForeignKey의 역참조는 QuerySet 반환

클라이언트와 서버 통신

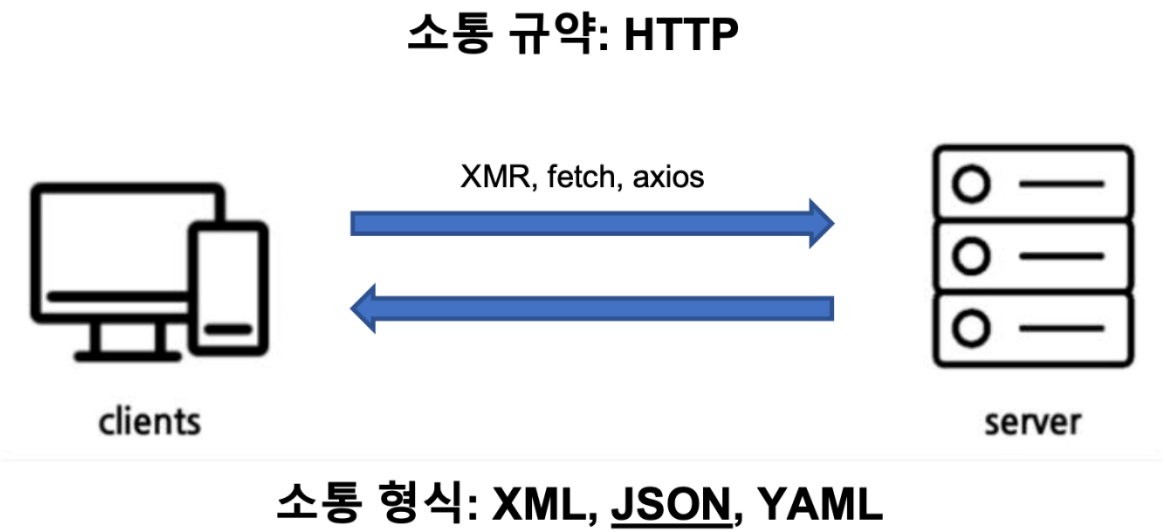
BE Mini Session

NEXT X LIKELION 정지우

세션 복습

1. 클라이언트 – 서버 통신

네트워크
: 컴퓨터 간 연결되어 있는 상태



세션 복습

1. 클라이언트 – 서버 통신

클라이언트와 서버는 웹에서 **HTTP 프로토콜**을 사용해서 통신한다.

클라이언트는 서버로 요청을 보내고,
서버는 요청에 따라 적절한 응답을 클라이언트로 회신한다.

필요에 따라 서버는 데이터베이스에 요청을 보내고,
회신 받은 응답을 활용한다.

HTTP란?

1. HTTP란?

HyperText Transfer Protocol

HTML과 같은 문서를 전송하기 위한
Application Layer Protocol

클라이언트가 HTTP messages 양식에 맞춰 요청을 보내면,
서버도 HTTP messages 양식에 맞춰 응답!

HTTP란?

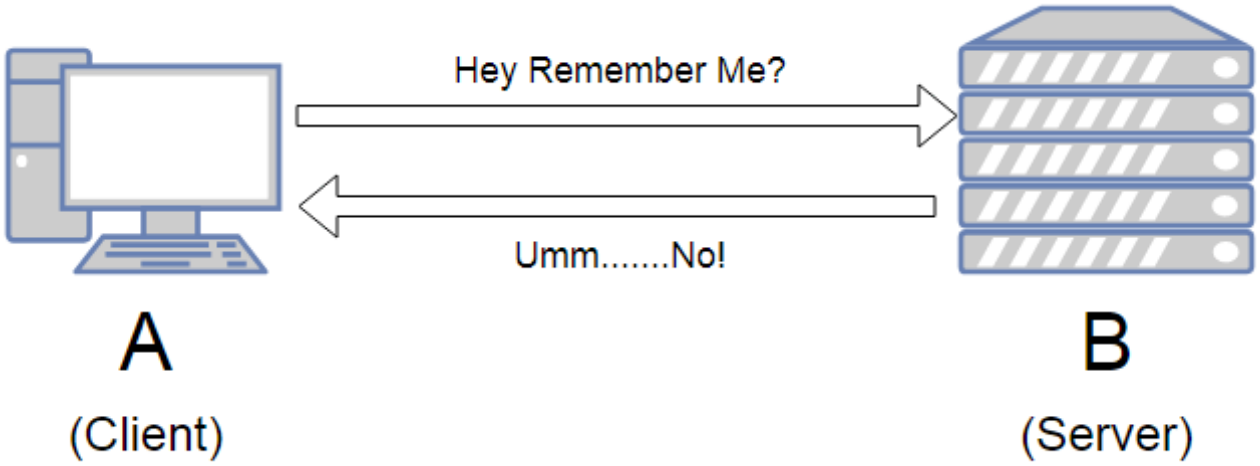
2. HTTP의 특성

Stateless & Connectionless
무상태성 & 비연결성

HTTP란?

2. HTTP의 특성 - 무상태성

Stateless : 무상태



HTTP란?

2. HTTP의 특성 - 무상태성

Stateful vs Stateless

(Stateful)

😀 : 아이스 아메리카노 얼마인가요?
🦁 : 5천원이요.
😀 : 두 잔 주세요.
🦁 : 만 원입니다. 결제 어떤 걸로 하세요?
😀 : 카드로 할게요.
🦁 : 만 원 결제되었습니다.

-> 아.아 상태 유지

-> 아아, 2잔 상태 유지

-> 아아, 2잔, 신용카드 상태 유지

HTTP란?

2. HTTP의 특성 - 무상태성

Stateful vs Stateless

(Stateful – 중간에 직원이 바뀌면?)

😀 : 아이스 아메리카노 얼마인가요?
🦁 : 5천원이요.
😀 : 두 잔 주세요.
🦁 : ?? 어떤 음료 두 잔이요??
😀 : 카드로 할게요.
🦁 : ?? 어떤 음료를 신용카드로 몇 잔 결제하세요??

-> 🦁 만 기억하고 있기 때문에 상태정보를 🦁 에게 알려줘야 함!

HTTP란?

2. HTTP의 특성 - 무상태성

Stateful vs Stateless

(Statefuless)

😁 : 아이스 아메리카노 얼마인가요?
🦁 : 5천원이요.
😁 : 두 잔 주세요.
🦁 만원입니다. 어떤 결로 결제하세요?
😁 : 카드로 할게요.
🐱 : 만원 결제 완료되었습니다.

-> 😁 이 자신의 주문을 기억하고 있다면 누가 와도 가능함!

HTTP란?

2. HTTP의 특성 - 무상태성

Stateful vs Stateless

(Stateful)
항상 같은 서버가 유지되어야 한다. 만약 서버에 장애가 생긴다면?
-> 서버1에 장애가 생기면 유지되던 상태정보가 다 날아가므로 다시 요청..

(Stateless)
아무 서버나 호출해도 된다. 만약 서버에 장애가 생긴다면?
-> 다른 서버에서 응답을 전달하면 되니 다시 요청할 필요 없음!

HTTP란?

2. HTTP의 특성 - 비연결성

Connection Oriented vs Connectionless

(Connection Oriented)
클라이언트가 요청을 보내지 않더라도 계속 연결을 유지
-> 연결을 유지하는 서버의 자원이 계속 소모됨..

(Connectionless)
실제 요청을 주고 받을 때만 연결을 유지하고 응답하면 연결 종료
-> 최소한의 자원으로 서버 유지 가능

HTTP란?

2. HTTP의 특성 – 한계와 극복

Stateless vs Connectionless

(Stateless)
모든 것을 무상태로 설계할 수 없는 경우도 있음!
-> 상태 유지 최소한으로 활용
Ex) 로그인
-> 로그인 상태를 서버에 유지

(Connectionless)
자원들을 각각 보낼 때마다 연결과 종료를 반복하는 것은 비효율적..
-> HTTP 지속 연결 : 모든 자원에 대한 응답이 돌아온 후 연결 종료

BE Mini Session

JSON

NEXT X LIKELION 전성운

목차

1. JSON

2. JSON XML YAML

3. 끝

JSON이란?

1. JSON?

JSON
=JavaScript Object Notation

자바스크립트에서 물건을 나타내는 표기

Object = 데이터
Notation = 저장방식

=데이터를 저장하는 방식!

JSON이란?

2. JSON의 저장방식

엑셀(.xlsx)

전체 (19,466) | 정상 (1,384) | 종료 (18,082)

약정코드	구성원코...	구성원명	생애주기단...	약정상...	약정일	모금상품	납입주기	정기납입액
222	2021030033	도너스	등록	정상	2021-03-25	겨울캠페인	정기납(무기한)	10,000/월
C21000221	2021030031	개인정보삭제	등록	정상	2021-03-24	플라스틱제로	일시납	
C21000220	2021030031	개인정보삭제	등록	정상	2021-03-23	장학사업	일시납	
C21000219	2021030021	홍길동	등록	정상	2021-03-23	겨울캠페인	일시납	
C21000218	2021030030	삼성2	등록	정상	2021-03-22	장학사업	정기납(무기한)	100,000/월
C21000217	2021030030	삼성2	등록	정상	2021-03-22	장학사업	정기납(무기한)	100,000/월
C21000216	2021030030	삼성2	등록	정상	2021-03-22	정기후원 (1년, 12회 납...	정기납(무기한)	5,000/월
C21000215	2021030027	유형1테스트	활성	정상	2021-03-19	장학사업	일시납	
C21000214	2021030028	유형1약정테...	활성	정상	2020-08-06	물품후원	일시납	
C21000213	2021030028	유형1약정테...	활성	정상	2021-08-05	물품후원	일시납	

JSON(.json)

```
{
    hey: "guy",
    anumber: 243,
  - anobject: {
        whoa: "nuts",
      - anarray: [
            1,
            2,
            "thr<h1>ee"
        ],
        more: "stuff"
    },
    awesome: true,
    bogus: false,
    meaning: null,
    japanese: "明日がある。",
    link: http://jsonview.com,
    notLink: "http://jsonview.com is great"
}
```

JSON이란?

3. JSON의 강점

1. Key 값을 이용하여 Value 값을 검색할 수 있다.
2. 다양한 방식으로 변환하거나 응용할 수 있다.
3. 저장할 때 용량이 적어서 효율적이다.

JSON이란?

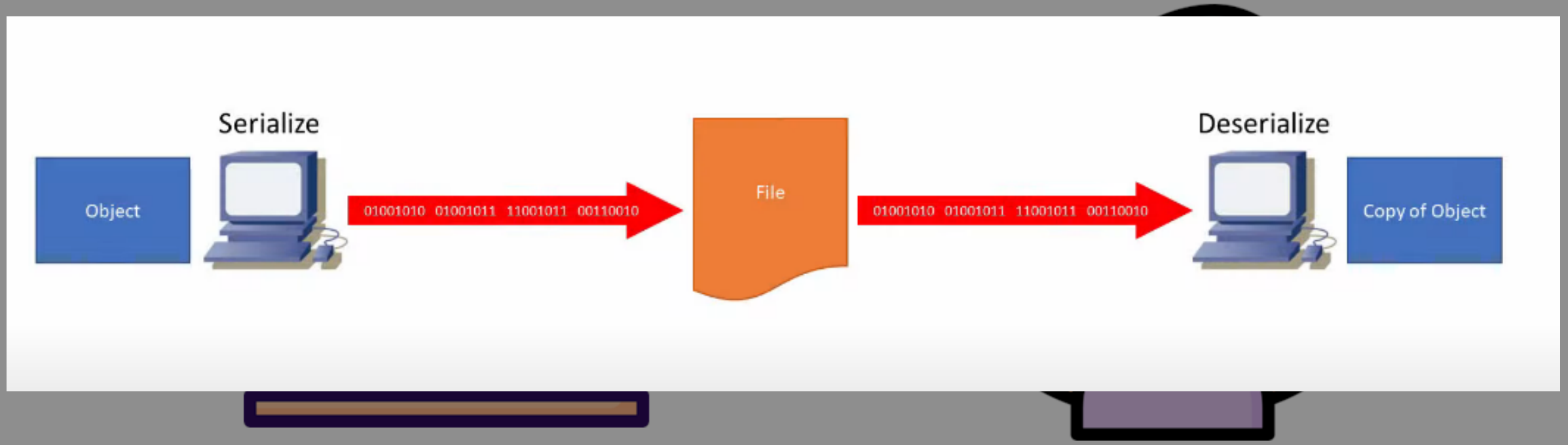
3. 대체 왜 JSON을 좋아하는 걸까??

Json...

다른 형식..

JSON이란?

3. 대체 왜 JSON을 좋아하는 걸까??



JSON이란?

4. JSON의 저장방식

엑셀(.xlsx)

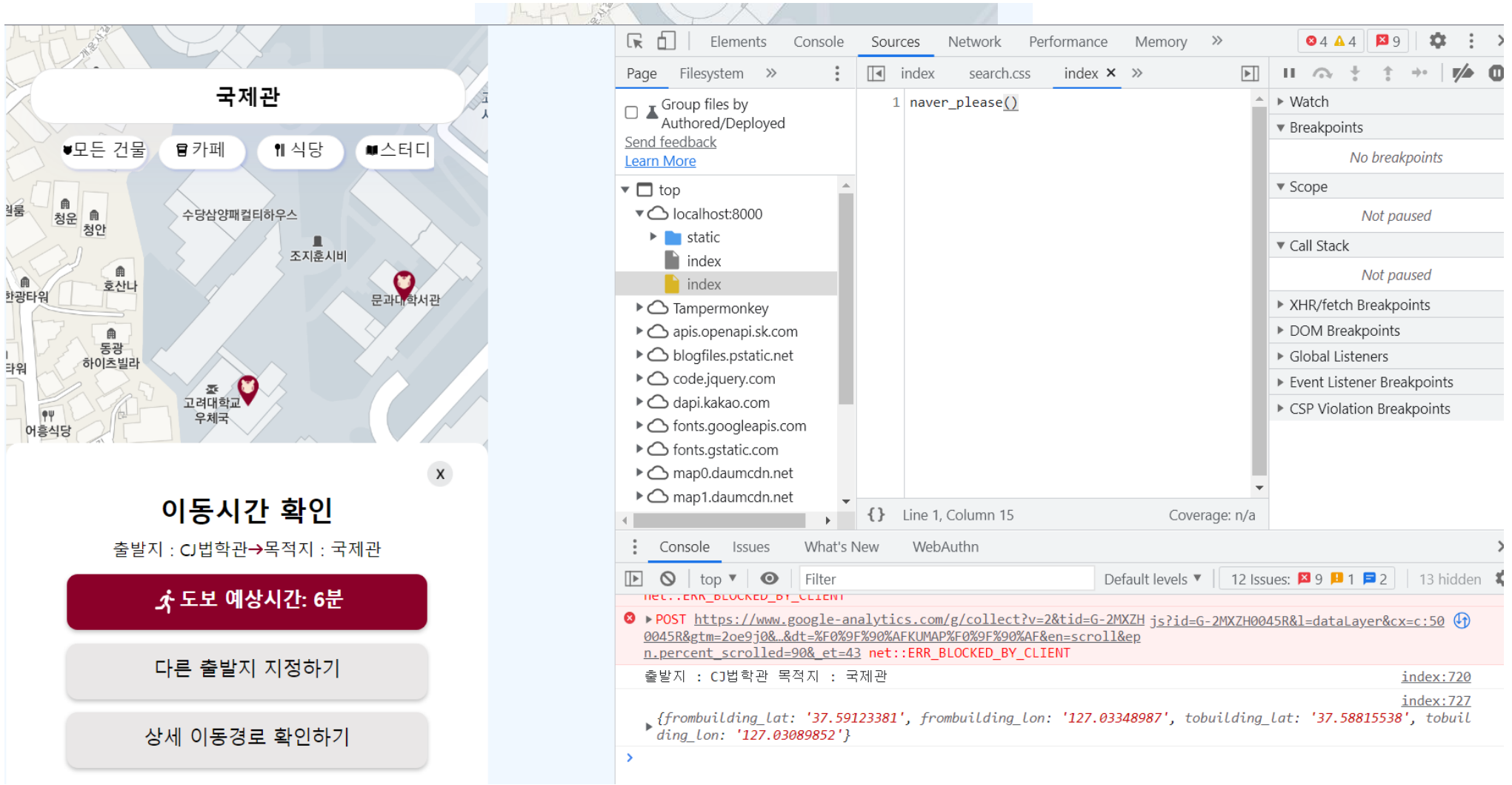
전체 (19,466) | 정상 (1,384) | 종료 (18,082)

약정코드	구성원코...	구성원명	생애주기단...	약정상...	약정일	모금상품	납입주기	정기납입액
222	2021030033	도너스	등록	정상	2021-03-25	겨울캠페인	정기납(무기한)	10,000/월
C21000221	2021030031	개인정보삭제	등록	정상	2021-03-24	플라스틱제로	일시납	
C21000220	2021030031	개인정보삭제	등록	정상	2021-03-23	장학사업	일시납	
C21000219	2021030021	홍길동	등록	정상	2021-03-23	겨울캠페인	일시납	
C21000218	2021030030	삼성2	등록	정상	2021-03-22	장학사업	정기납(무기한)	100,000/월
C21000217	2021030030	삼성2	등록	정상	2021-03-22	장학사업	정기납(무기한)	100,000/월
C21000216	2021030030	삼성2	등록	정상	2021-03-22	정기후원 (1년, 12회 납...	정기납(무기한)	5,000/월
C21000215	2021030027	유형1테스트	활성	정상	2021-03-19	장학사업	일시납	
C21000214	2021030028	유형1약정테...	활성	정상	2020-08-06	물품후원	일시납	
C21000213	2021030028	유형1약정테...	활성	정상	2021-08-05	물품후원	일시납	

JSON(.json)

```
{
    hey: "guy",
    anumber: 243,
  - anobject: {
        whoa: "nuts",
      - anarray: [
            1,
            2,
            "thr<h1>ee"
        ],
        more: "stuff"
    },
    awesome: true,
    bogus: false,
    meaning: null,
    japanese: "明日がある。",
    link: http://jsonview.com,
    notLink: "http://jsonview.com is great"
}
```

JSON의 활용(쿠맵...)


국제관
모든 건물
카페
식당
스터디
청운
청안
수당삼양패컬티하우스
조지훈시비
호산나
문과대학서관
동광
하이츠빌라
고려대학교
우체국
어흥식당
이동시간 확인
출발지 : CJ법학관→목적지 : 국제관
도보 예상시간: 6분
다른 출발지 지정하기
상세 이동경로 확인하기
Elements
Console
Sources
Network
Performance
Memory
Page
Filesystem
index
search.css
Group files by Authored/Deployed
Send feedback
Learn More
top
localhost:8000
static
Tampermonkey
apis.openapi.sk.com
blogfiles.pstatic.net
code.jquery.com
dapi.kakao.com
fonts.googleapis.com
fonts.gstatic.com
map0.daumcdn.net
map1.daumcdn.net
naver_please()
Line 1, Column 15
Coverage: n/a
Watch
Breakpoints
No breakpoints
Scope
Not paused
Call Stack
Not paused
XHR/fetch Breakpoints
DOM Breakpoints
Global Listeners
Event Listener Breakpoints
CSP Violation Breakpoints
Console
Issues
What's New
WebAuthn
Filter
Default levels
12 Issues: 9 1 2
13 hidden
POST https://www.google-analytics.com/g/collect?v=2&tid=G-2MXZH0045R&gtm=2oe9j0&…&dt=%F0%9F%90%AFKUMAP%F0%9F%90%AF&en=scroll&epn.percent_scrolled=90&_et=43 net::ERR_BLOCKED_BY_CLIENT
js?id=G-2MXZH0045R&l=dataLayer&cx=c:50
출발지 : CJ법학관 목적지 : 국제관
index:720
index:727
{frombuilding_lat: '37.59123381', frombuilding_lon: '127.03348987', tobuilding_lat: '37.58815538', tobuilding_lon: '127.03089852'}

JSON의 활용(쿠맵...)

```
$.ajax({
  url: '/time/' + fromBuildingName +'/'+toBuildingName,
  async: false,
  type: "GET",
  dataType: "json",
  success: (latlon) => {      JHLvJz, last month • 밤샘결과모달완성및아이콘업데이트
        console.log(latlon)
        $.ajax({
          method : "POST",
          url : "https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1&format=json&callback=result",
          async : false,
          data : {
            "appKey" : "{{tmapKey}}",
            "startX" : latlon.frombuilding_lon,
            "startY" : latlon.frombuilding_lat,
            "endX" : latlon.tobuilding_lon,
            "endY" : latlon.tobuilding_lat,
            "reqCoordType" : "WGS84GEO",
            "resCoordType" : "EPSG3857",
            "startName" : "출발지",
            "endName" : "도착지"
          },
          success: function(response){
            var resultData = response.features;
            times = ((resultData[0].properties.totalTime) / 60).toFixed(0)
            showThirdModal(times,fromBuildingName,toBuildingName)
          },
                  })
```

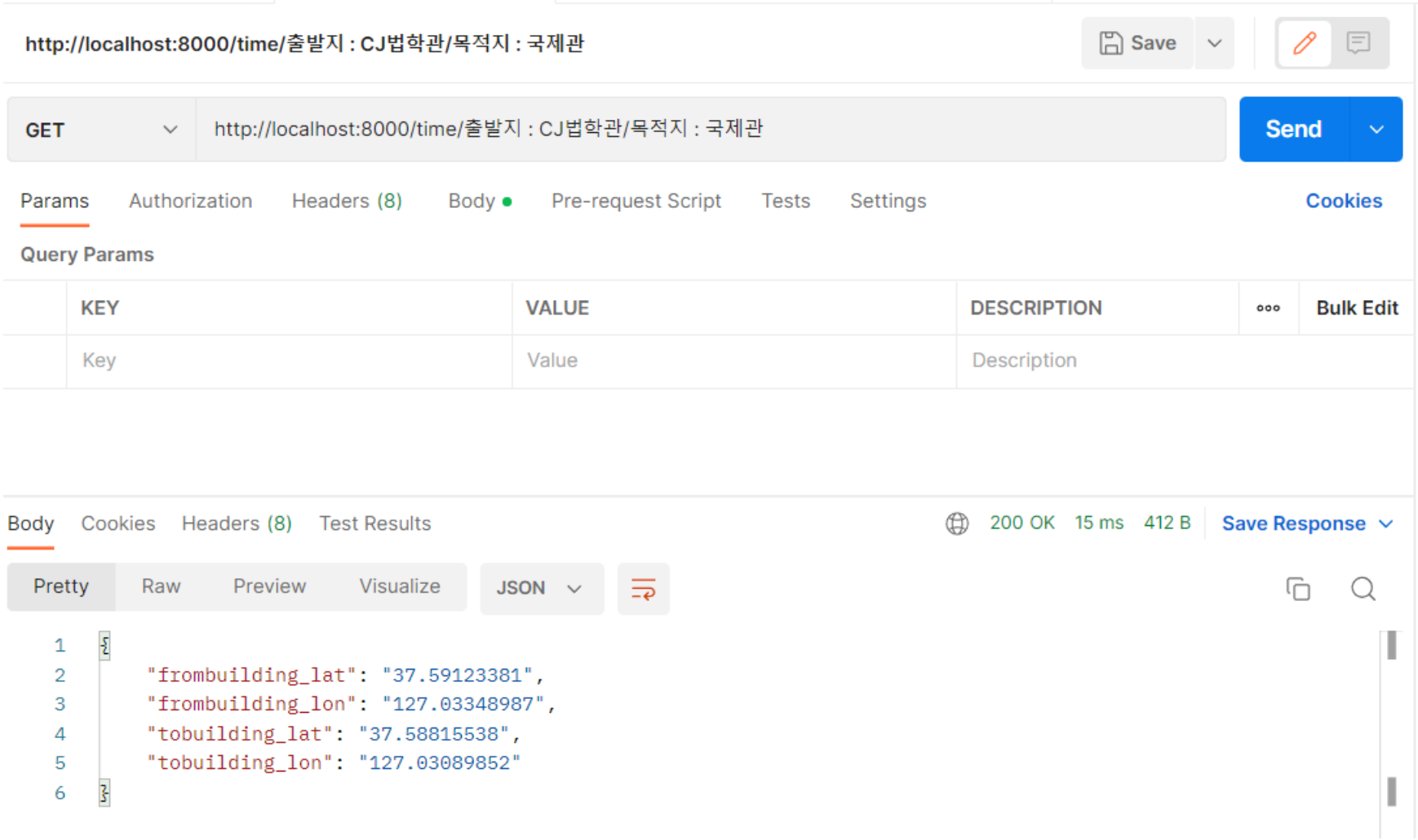
JSON의 활용(쿠맵...)

```
$.ajax({
  url: '/time/' + fromBuildingName +'/'+toBuildingName,
  async: false,
  type: "GET",
  dataType: "json",
  success: (latlon) => {
        console.log(latlon)
        $.ajax({
          method : "POST",
          url : "https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1&format=json&callback=result",
          async : false,
          data : {
            "appKey" : "{{tmapKey}}",
            "startX" : latlon.frombuilding_lon,
            "startY" : latlon.frombuilding_lat,
            "endX" : latlon.tobuilding_lon,
            "endY" : latlon.tobuilding_lat,
            "reqCoordType" : "WGS84GEO",
            "resCoordType" : "EPSG3857",
            "startName" : "출발지",
            "endName" : "도착지"
          },
          success: function(response){
            var resultData = response.features;
            times = ((resultData[0].properties.totalTime) / 60).toFixed(0)
            showThirdModal(times,fromBuildingName,toBuildingName)
          },
                  })
```

JSON의 활용(쿠맵...)

```python
@csrf_exempt
def time(request, from_building, to_building):
    print(from_building, to_building)       You, 1 second ago • Uncommit
    from_building = from_building[6:]
    to_building = to_building[6:]
    print(1234, from_building,to_building)
    fromBuilding = Building.objects.get(building_name = from_building)
    toBuilding = Building.objects.get(building_name = to_building)
    data = {
        'frombuilding_lat':str(fromBuilding.building_lat),
        'frombuilding_lon':str(fromBuilding.building_lon),
        'tobuilding_lat':str(toBuilding.building_lat),
        'tobuilding_lon':str(toBuilding.building_lon),
    }
    print(data)
    return JsonResponse(data)
```

JSON의 활용(쿠맵...)


http://localhost:8000/time/출발지 : CJ법학관/목적지 : 국제관
Save
GET
http://localhost:8000/time/출발지 : CJ법학관/목적지 : 국제관
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Key
Value
Description
Body
Cookies
Headers (8)
Test Results
200 OK
15 ms
412 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"frombuilding_lat": "37.59123381",
"frombuilding_lon": "127.03348987",
"tobuilding_lat": "37.58815538",
"tobuilding_lon": "127.03089852"
}

JSON의 활용(쿠맵...)

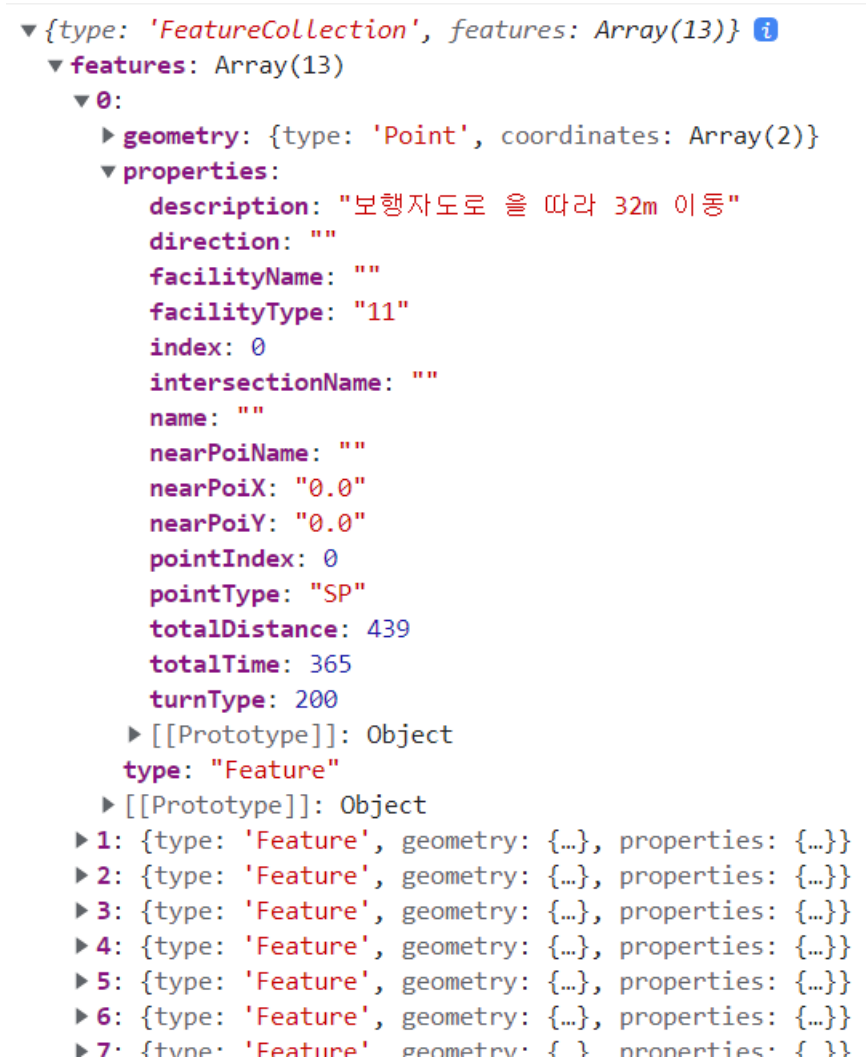
```
$.ajax({
  url: '/time/' + fromBuildingName +'/'+toBuildingName,
  async: false,
  type: "GET",
  dataType: "json",
  success: (latlon) => {       JHLvJz, last month • 밤샘결과모달완성및아이콘업데이트
      console.log(latlon)
      $.ajax({
        method : "POST",
        url : "https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1&format=json&callback=result",
        async : false,
        data : {
          "appKey" : "{{tmapKey}}",
          "startX" : latlon.frombuilding_lon,
          "startY" : latlon.frombuilding_lat,
          "endX" : latlon.tobuilding_lon,
          "endY" : latlon.tobuilding_lat,
          "reqCoordType" : "WGS84GEO",
          "resCoordType" : "EPSG3857",
          "startName" : "출발지",
          "endName" : "도착지"
        },
        success: function(response){
          var resultData = response.features;
          times = ((resultData[0].properties.totalTime) / 60).toFixed(0)
          showThirdModal(times,fromBuildingName,toBuildingName)
        },
                  })
```

JSON의 활용(쿠맵...)

```
success: (latlon) => {
        console.log(latlon)
        $.ajax({
          method : "POST",
          url : "https://apis.openapi.sk.com/tmap/routes/pedestrian?version=1&format=json&callback=result",
          async : false,
          data : {
            "appKey" : 'l7xx8023ed37ddc6432ca276a6b0046e843b',
            "startX" : "127,03348987",
            "startY" : "37.59123381",
            "endX" : "127.03089852",
            "endY" : "37.58815538",
            "reqCoordType" : "WGS84GEO",
            "resCoordType" : "EPSG3857",
            "startName" : "출발지",
            "endName" : "도착지"
          },
          success: function(response){
            var resultData = response.features;
            times = ((resultData[0].properties.totalTime) / 60).toFixed(0)
            showThirdModal(times,fromBuildingName,toBuildingName)
          },
                  })
}
```

JHLvJz, last month • 밤샘결과모달완성및아이콘업데이트

JSON의 활용(쿠맵...)


{type: 'FeatureCollection', features: Array(13)}
features: Array(13)
0:
geometry: {type: 'Point', coordinates: Array(2)}
properties:
description: "보행자도로 을 따라 32m 이동"
direction: ""
facilityName: ""
facilityType: "11"
index: 0
intersectionName: ""
name: ""
nearPoiName: ""
nearPoiX: "0.0"
nearPoiY: "0.0"
pointIndex: 0
pointType: "SP"
totalDistance: 439
totalTime: 365
turnType: 200
[[Prototype]]: Object
type: "Feature"
[[Prototype]]: Object
1: {type: 'Feature', geometry: {…}, properties: {…}}
2: {type: 'Feature', geometry: {…}, properties: {…}}
3: {type: 'Feature', geometry: {…}, properties: {…}}
4: {type: 'Feature', geometry: {…}, properties: {…}}
5: {type: 'Feature', geometry: {…}, properties: {…}}
6: {type: 'Feature', geometry: {…}, properties: {…}}

JSON의 활용(쿠맵...)

```
/* DB 정보 저장 */
const buildings = {{ buildings|safe }};
console.log(buildings)
const temp1 = JSON.stringify(buildings);
console.log(temp1)
const temp2 = JSON.parse(temp1)
console.log(temp2)
```

JSON의 활용(쿠맵...)

```
/* DB 정보 저장 */
const buildings = {{ buildings|safe }};
console.log(buildings)
```

```
index:511
▼ (53) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
  ▼ 0:
    ▶ fields: {building_name: '공학관', history: "- 1996년 준공되어 주로 공과대학 학생들이 사용하는 건물\r\n- 건물 구조상 'ㄱ'모양으…층은 비상계단이나 엘레베이터를 이용해야 함\r\n- 공학관 후문은 창의관 2층 후문으
      model: "KUMAPapp.building"
      pk: 2
    ▶ [[Prototype]]: Object
  ▶ 1: {model: 'KUMAPapp.building', pk: 3, fields: {…}}
  ▶ 2: {model: 'KUMAPapp.building', pk: 4, fields: {…}}
  ▶ 3: {model: 'KUMAPapp.building', pk: 5, fields: {…}}
  ▶ 4: {model: 'KUMAPapp.building', pk: 6, fields: {…}}
  ▶ 5: {model: 'KUMAPapp.building', pk: 7, fields: {…}}
  ▶ 6: {model: 'KUMAPapp.building', pk: 8, fields: {…}}
  ▶ 7: {model: 'KUMAPapp.building', pk: 9, fields: {…}}
  ▶ 8: {model: 'KUMAPapp.building', pk: 10, fields: {…}}
  ▶ 9: {model: 'KUMAPapp.building', pk: 11, fields: {…}}
  ▶ 10: {model: 'KUMAPapp.building', pk: 12, fields: {…}}
  ▶ 11: {model: 'KUMAPapp.building', pk: 13, fields: {…}}
  ▶ 12: {model: 'KUMAPapp.building', pk: 14, fields: {…}}
  ▶ 13: {model: 'KUMAPapp.building', pk: 15, fields: {…}}
  ▶ 14: {model: 'KUMAPapp.building', pk: 16, fields: {…}}
  ▶ 15: {model: 'KUMAPapp.building', pk: 17, fields: {…}}
  ▶ 16: {model: 'KUMAPapp.building', pk: 18, fields: {…}}
  ▶ 17: {model: 'KUMAPapp.building', pk: 19, fields: {…}}
```

JSON의 활용(쿠맵...)

```
const temp1 = JSON.stringify(buildings);
console.log(temp1)
```

[{"model":"KUMAPapp.building","pk":2,"fields":{"building_name":"공학관","history":"- 1996년 준공되어 주로 공과대학 학생들이 사용하는 건물\r\n- 건물 구조상 'ㄱ'모양으로 꺾여 있어 아산이학관과 마주보 index:513
는 쪽은 7층, 하나스퀘어를 바라보는 면은 5층으로 설계됨\r\n- 정문의 중앙계단은 5층까지만 있어 6, 7층은 비상계단이나 엘레베이터를 이용해야 함\r\n- 공학관 후문은 창의관 2층 후문으로 통
함","building_lat":37.58362696,"building_lon":127.0253717}},{"model":"KUMAPapp.building","pk":3,"fields":{"building_name":"기초과학관","history":"- 1993년 준공된, 정부 설립의 기초과학 지원 연구소가 있는 건
물\r\n- 산학연의 '연'을 대표함\r\n- 대학의 기초 과학 연구에 필요한 첨단 연구 기기를 설치하여 측정 및 분석 지원을 수행하고 있음","building_lat":37.58487054,"building_lon":127.0243645}},
{"model":"KUMAPapp.building","pk":4,"fields":{"building_name":"로봇융합관","history":"- 로봇이 철골 구조 공사를 담당한 국내 최초의 건물로, 2011년 건립됨\r\n- 한국건설교통기술평가원의 관련 연구과제 수행을 위해
지어짐","building_lat":37.58184275,"building_lon":127.02645794860015}},{"model":"KUMAPapp.building","pk":5,"fields":{"building_name":"신공학관","history":"- 2016년 준공된 공과대학 학생들이 주로 이용하는 건
물\r\n- 교수연구실, 연구실, 전산실습실, KUICEE 등이 위치함\r\n- 자연계 지역만의 깔끔하고 모던한 이미지를 대표함\r\n- 1층 로비의 무인 자동차는 1992년 한민홍 교수가 국내 최초로 개발한 무인자동차
임","building_lat":37.58250949,"building_lon":127.0264242}},{"model":"KUMAPapp.building","pk":6,"fields":{"building_name":"애기능학생회관","history":"- 1982년 건립되어 공과대학 별관으로 쓰였는데, 2001년 리
모델링하여 애기능학생회관으로 바뀜\r\n- 각 단과대학 학생회 및 애기능 동아리 연합회 등 학생자치 전용 공간이 위치함\r\n- 매 학기 초 공과대학 학생회가 1층에서 '헌책방'을 운영
함","building_lat":37.58244616986338,"building_lon":127.0275337}},{"model":"KUMAPapp.building","pk":7,"fields":{"building_name":"우정정보관(정보통신관)","history":"- 100억여 원을 기부한 (주)부영주택 이중근
회장을 호를 따 2011년 준공된 건물\r\n- 교수연구실, 강의실 등이 위치해있고 지하1층은 미래융합기술관과 연결됨\r\n- 건물 전면이 커튼 월 시스템으로 설계되어 충분한 환기가 가능
함","building_lat":37.58523904,"building_lon":127.0284348}},{"model":"KUMAPapp.building","pk":8,"fields":{"building_name":"이학관 별관","history":"- 이과대학의 지구환경과학과 실험실을 비롯하여 다양한 연구소
들이 입주해 있음\r\n- 5층짜리 건물이지만 4층이 존재하지 않음\r\n- 이과 학생들의 필수 이수과목인 실험 수업이 이루어지기도 함","building_lat":37.5839417,"building_lon":127.0280211}},
{"model":"KUMAPapp.building","pk":9,"fields":{"building_name":"제2실험관","history":"- 금속관련 공정 연구실, 화공생명공정실험실, 강의실 등이 위치한 건물\r\n- 공과대학 학생들이 실험 및 실습을 하는 건물\r\n- 1
층의 금속공정연구실의 문을 열 때는 음악소리가 남","building_lat":37.58324853,"building_lon":127.0253999}},{"model":"KUMAPapp.building","pk":10,"fields":{"building_name":"창의관","history":"- 2003년 준공되
어, 계단 강의실, 라운지, 컴퓨터실 등이 위치한 공과대학 건물\r\n- 공과대학의 학생증을 이용해야만 출입할 수 있음\r\n- 2층 후문으로 나가면 공학관 후문과 통하며, 벚나무가 있는 산책로가 조성되어 있어 숨은 데이트 명소라
불림\r\n- 2층 여자화장실이 한쪽 복도 끝에 숨겨져 있음","building_lat":37.58316279745723,"building_lon":127.0260452}},{"model":"KUMAPapp.building","pk":11,"fields":{"building_name":"환경실험
관","history":".","building_lat":37.584216559287064,"building_lon":127.02780048904}},{"model":"KUMAPapp.building","pk":12,"fields":{"building_name":"하나스퀘어","history":"- 2006년 준공된, 고려대 과학도서
관 앞을 동서로 가로지르는 학생복지 및 휴게시설 건물\r\n- 물에 배가 떠 있는 모양의 선큰(Sunken) 구조 건물\r\n- 하나스퀘어의 미니스톱은 시중의 편의점보다 약간 저렴한 가격으로 판매
함","building_lat":37.5846168,"building_lon":127.0253533}},{"model":"KUMAPapp.building","pk":13,"fields":{"building_name":"하나과학관","history":"- 하나은행이 본교발전기금 120억원 기부해 2014년 건립된 건물
\r\n- 대형 실험연구공간으로서 실험실을 중심으로 강의실, 연구실, 세미나실 등으로 구성됨\r\n- A/B동으로 구성되며 지하 2층-지하 1층, 지상 5층-지상 1층이 서로 연결됨\r\n- 지하1층에 큰 휴게실이 있어 보건과학대학 학생
들이 주로 여기서 휴식을 취함","building_lat":37.5858469,"building_lon":127.0251516}},{"model":"KUMAPapp.building","pk":14,"fields":{"building_name":"CJ식품안전관","history":"- 2008년 CJ로부터 70억원의 연구
자금을 유치하여 건립한 식품안전 교육 관련 건물\r\n- 연구실, 세미나실, 3개의 산업체 연구소 등이 들어서 식품 관련 연구와 교육에 큰 도움이 됨\r\n- 연구실이 많아 학부생들은 거의 이용하지 않
음","building_lat":37.5860654,"building_lon":127.0265924}},{"model":"KUMAPapp.building","pk":15,"fields":{"building_name":"과학도서관","history":"- 본교 건축공학과 박윤성 교수가 설계하여 1983년 준공된 도서
관\r\n- 중앙도서관의 1.3배 크기로 안암 캠퍼스 내 가장 큰 단일 도서관\r\n- 1층 서고에는 한국서적, 2층 서고에는 외국서적이 배치되어 있음","building_lat":37.5845688,"building_lon":127.0265505}},
{"model":"KUMAPapp.building","pk":16,"fields":{"building_name":"메디힐지구환경관","history":"- 엘앤피코스메틱 권오섭 대표의 120억 기부에 의해 2020년 건립된 건물\r\n- 지구환경 변화와 미래 지구환경을 개선하기
위한 시설","building_lat":37.5855144849069,"building_lon":127.02561010664337}},{"model":"KUMAPapp.building","pk":17,"fields":{"building_name":"미래융합기술관","history":"- 첨단 과학기술 연구와 산학협력 활
성화를 위해 2008년 준공된 건물\r\n- 사이버국방학과 강의가 대부분 이 건물에서 열림\r\n- 외부에서 봤을 때 꼭대기의 모양이 학사모의 형태를 띠고 있
음","building_lat":37.58483588654065,"building_lon":127.0282450192107}},{"model":"KUMAPapp.building","pk":18,"fields":{"building_name":"산학관","history":"- 1996년 건립된, 국내 최초로 설립된 한국 산학연 종
합연구원이 자리한 건물\r\n- 멀티미디어 화상회의실, 세미나실, 휴게실 등이 있음\r\n- 대학원생과 기업이 연계하여 연구를 하는 용도로 쓰이므로 학부생이 찾게 되는 경우는 드
뭄","building_lat":37.5841446,"building_lon":127.0244645}},{"model":"KUMAPapp.building","pk":19,"fields":{"building_name":"생명과학관 동관","history":"- 2003년 건립되어 일반 강의실, 휴게실, 실험실 등이 있
는 생명과학대학 건물\r\n- 4층의 오정강당에서 대형 강의와 세미나실용으로 쓰임\r\n- 월드컵 응원전 때 오정강당에 모여 치킨을 시켜먹으며 응원하기도 함","building_lat":37.5855889,"building_lon":127.028103}},
{"model":"KUMAPapp.building","pk":20,"fields":{"building_name":"생명과학관 서관","history":"- 1977년 철제 콘크리트와 화강석으로 준공된 생명과학대학 건물\r\n- 2004년 리모델링을 실시하여 지하 1층과 각 층 사이에
실험실이 있음\r\n- 지하에는 생명대 학생들이 쓰는 사물함이, 옥상에는 예쁜 옥실이 있음\r\n- 명명 공간에서 흡대폭이 거의 더 Show more (26.0 kB) Copy

JSON의 활용(쿠맵...)

```
const temp2 = JSON.parse(temp1)
console.log(temp2)
```

```
index:515
(53) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
  0:
    fields:
      building_lat: 37.58362696
      building_lon: 127.0253717
      building_name: "공학관"
      history: "- 1996년 준공되어 주로 공과대학 학생들이 사용하는 건물\r\n- 건물 구조상 'ㄱ'모양으로 꺾여 있어 아산이학관과 마주보는 쪽은 7층, 하나스퀘어를 바라보는 면은 5층으로 설계됨\r\n- 정문의 중앙계단은 5층
    ▸ [[Prototype]]: Object
    model: "KUMAPapp.building"
    pk: 2
  ▸ [[Prototype]]: Object
▸ 1: {model: 'KUMAPapp.building', pk: 3, fields: {…}}
▸ 2: {model: 'KUMAPapp.building', pk: 4, fields: {…}}
▸ 3: {model: 'KUMAPapp.building', pk: 5, fields: {…}}
▸ 4: {model: 'KUMAPapp.building', pk: 6, fields: {…}}
▸ 5: {model: 'KUMAPapp.building', pk: 7, fields: {…}}
▸ 6: {model: 'KUMAPapp.building', pk: 8, fields: {…}}
▸ 7: {model: 'KUMAPapp.building', pk: 9, fields: {…}}
▸ 8: {model: 'KUMAPapp.building', pk: 10, fields: {…}}
▸ 9: {model: 'KUMAPapp.building', pk: 11, fields: {…}}
▸ 10: {model: 'KUMAPapp.building', pk: 12, fields: {…}}
▸ 11: {model: 'KUMAPapp.building', pk: 13, fields: {…}}
▸ 12: {model: 'KUMAPapp.building', pk: 14, fields: {…}}
▸ 13: {model: 'KUMAPapp.building', pk: 15, fields: {…}}
▸ 14: {model: 'KUMAPapp.building', pk: 16, fields: {…}}
```

JSON XML YAML

컴퓨터 끼리 데이터를 주고 받아야야하는데,
엔터키가 없다면??

-> JSON
-> XML

JSON XML YAML

2. JSON과 XML

JSON

```
{
    hey: "guy",
    anumber: 243,
  - anobject: {
        whoa: "nuts",
      - anarray: [
            1,
            2,
            "thr<h1>ee"
        ],
        more: "stuff"
    },
    awesome: true,
    bogus: false,
    meaning: null,
    japanese: "明日がある。",
    link: http://jsonview.com,
    notLink: "http://jsonview.com is great"
}
```

XML

JSON XML YAML

3. XML의 강점

1. JSON에 비하여 문법 오류에 더 강함
2. 주석을 달 수 있다.

∴ 안전성 = XML , 가벼움 = JSON

JSON XML YAML

3. YAML

JSON

```json
{
  "apiVersion": "v1",
  "kind": "Pod",
  "metadata": {
    "name": "spring-pod"
  },
  "spec": {
    "containers": [
      {
        "image": "amro/springapp:example",
        "name": "spring-app",
        "ports": [
          {
            "containerPort": 80,
            "protocol": "TCP"
          }
        ]
      }
    ]
  }
}
```

YAML ✔

```yaml
apiVersion: v1
kind: Pod
metadata:
    name: spring-pod
spec:
    containers:
    - image: amro/springapp:example
      name: spring-app
      ports:
      - containerPort: 80
        protocol: TCP
```

감사합니다

XML / YAML

Han Seong Su

XML이란?

Extended Markup Language

(확장된 마크업 언어)

XML이란?

XML

문서 토론

위키백과, 우리 모두의 백과사전.

XML(eXtensible Markup Language)은 W3C에서 개발된, 다른 특수한 목적을 갖는 마크업 언어를 만드는데 사용하도록 권장하는 다목적 마크업 언어이다. XML은 SGML의 단순화된 부분집합으로, 다른 많은 종류의 데이터를 기술하는 데 사용할 수 있다. XML은 주로 다른 종류의 시스템, 특히 인터넷에 연결된 시스템끼리 데이터를 쉽게 주고 받을 수 있게 하여 HTML의 한계를 극복할 목적으로 만들어졌다.

기계는 인간의 언어를 읽거나 이해할 수 없는 계산기에 불과하므로 XML과 같은 구조화된 마크업 언어들은 인간의 읽고 분석하여 이해하는 능력과 컴퓨터의 단순한 계산적인 판독 능력 사이에 타협점을 만들어 줄 수 있다. W3C가 만든 XML 1.0 Specification[1]과 몇몇 다른 관련 명세들[2]과 모든 자유 개방형 표준[3]에서 정의되었다.

W3C는 XML 설계 목표에서 단순성과 일반성, 그리고 인터넷을 통한 사용 가능성을 강조했다.[4] XML은 텍스트 데이터 형식으로 유니코드를 사용해 전 세계 언어를 지원한다. XML을 설계할 때는 주로 문서를 표현하는데 집중했지만, 지금은 임의의 자료구조를 나타내는 데 널리 쓰인다. 대표적인 예가 웹 서비스이다.

많은 API가 개발되어 XML 데이터를 처리하고자 하는 소프트웨어 개발자들이 활용하고 있다. 또한, 여러 가지 스키마 시스템이 있어서 XML 기반 언어의 정의를 보다 쉽게 할 수 있도록 도와준다.

역사 [편집]

XML은 SGML의 애플리케이션 프로파일이다.[5]

동적 정보 표시를 위한 SGML의 다재다능함은 인터넷의 성장 이전인 1980년대 말에 초기 디지털 미디어 출판사들에 의해 인지되었다.[6][7] 1990년대 중순, 일부 SGML 실천자들은 당시 새로운 월드 와이드 웹을 경험하였고 웹이 성장할수록 마주칠 가능성이 있던 문제들 중 일부를 SGML이 해결해줄 것이라 믿었다. 댄 커널리는 1995년 당시 직원으로 있었을 때 SGML을 W3C의 활동 목록에 추가하였다. 작업은 1996년 중순 썬 마이크로시스템즈의 엔지니어 존 보삭이 선언문을 만들고 협업자들을 모집하였을 때 시작되었다. 보삭은 SGML과 웹에 모두 경험이 있는 사람들의 작은 공동체와 잘 어울렸다.[8]

주요 디자인 결정은 1996년 7월과 11월 사이 도달했으며,[9] 당시 XML 사양의 최초 워킹 드래프트가 출판되었다.[10] 추가 디자인 작업이 1997년에 계속되었으며 XML 1.0은 1998년 2월 10일 W3C 권고안이 되었다.

XML 예제

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE recipe PUBLIC "-//Happy-Monkey//DTD
RecipeBook//EN""http://www.happy-
monkey.net/recipebook/recipebook.dtd">
<recipe>
        <title>Peanutbutter On A Spoon</title>
        <ingredientlist>
                <ingredient>Peanutbutter</ingredient>
        </ingredientlist>
        <preparation>Stick a spoon in a jar of
peanutbutter, scoopand pull out a big glob of
peanutbutter.</preparation>
</recipe>
```

XML 예제

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE recipe PUBLIC "-//Happy-Monkey//DTD
RecipeBook//EN""http://www.happy-
monkey.net/recipebook/recipebook.dtd">
<recipe>
        <title>Peanutbutter On A Spoon</title>
        <ingredientlist>
                <ingredient>Peanutbutter</ingredient>
        </ingredientlist>
        <preparation>Stick a spoon in a jar of
peanutbutter, scoopand pull out a big glob of
peanutbutter.</preparation>
</recipe>
```

XML 선언

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

XML 문서임을 밝히는 행
XML의 버전과 인코딩, 스탠드얼론 여부를 선언한다.
항상 문서의 첫 번째 행에 와야 한다.
(필수는 아니다)

XML 선언

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

버전(version) : XML 버전이다. 권고안은 1.0이다.

인코딩(encoding) : 인코딩 방법이다. 미 기입 시 기본값은 UTF-8이다.

스탠드얼론(standalone) : XML 문서를 XML 파서가 해석할 때 외부 DTD(Document Type Definition) 혹은 외부 XSD(XML Scheme Document) 등의 외부 XML 스키마를 사용할지 지정하는 부분이다. 미 기입 시 기본값은 no다.

XML 예제

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE recipe PUBLIC "-//Happy-Monkey//DTD
RecipeBook//EN""http://www.happy-
monkey.net/recipebook/recipebook.dtd">
<recipe>
        <title>Peanutbutter On A Spoon</titl
        <ingredientlist>
                <ingredient>Peanutbutter</ingr
        </ingredientlist>
        <preparation>Stick a spoon in a jar
peanutbutter, scoopand pull out a big glob of
peanutbutter.</preparation>
</recipe>
```

XML 예제

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE recipe PUBLIC "-//Happy-Monkey//DTD
RecipeBook//EN""http://www.happy-
monkey.net/recipebook/recipebook.dtd">
<recipe>
        <title>Peanutbutter On A Spoon</title>
        <ingredientlist>
                <ingredient>Peanutbutter</ingredient>
        </ingredientlist>
        <preparation>Stick a spoon in a jar of
peanutbutter, scoopand pull out a big glob of
peanutbutter.</preparation>
</recipe>
```

XML 본문

```
<recipe>
	<title>Peanutbutter On A Spoon</title>
	<ingredientlist>
		<ingredient delicious="yes">Peanutbutter</ingredient>
	</ingredientlist>
	<preparation>Stick a spoon in a jar of peanutbutter, scoopand
pull out a big glob of peanutbutter.</preparation>
</recipe>
```

HTML 양식이랑 비슷하다.

XML 본문

```
<recipe>
	<title>Peanutbutter On A Spoon</title>
	<ingredientlist>
		<ingredient delicious="yes">Peanutbutter</ingredient>
	</ingredientlist>
	<preparation>Stick a spoon in a jar of peanutbutter, scoopand
pull out a big glob of peanutbutter.</preparation>
</recipe>
```

HTML 양식이랑 비슷하다.

XML 본문

```
<ingredient delicious="yes">Peanutbutter</ingredient>
```

<ingredient>, </ingredient>
태그(Tag), 빈 태그도 가능 : <line-break />

delicious=“yes”
어트리뷰트(Attribute), name과 value로 나뉨

Peanutbutter
내용(Content), escape 문자를 쓰면 안 됨

Escape문자 변환

& : &
< : <
> : >
' : '
" : "

출처: https://stex.tistory.com/33 [Jays's IT Collect:티스토리]

==========

일일이 다 써주기 귀찮다.....

CDATA

<! [CDATA[내용]]>

특수문자를 써주고 싶으면
이 마크업을 사용해주자!

Ex)

```
<ingredient delicious="yes"><![CDATA[<Peanutbutter>]]></ingredient>
```

➔ <Peanutbutter>

XML 스키마

다시 돌아와서...

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE recipe PUBLIC "-//Happy-Monkey//DTD
RecipeBook//EN""http://www.happy-
monkey.net/recipebook/recipebook.dtd">
```

XML 스키마

XML 스키마

XML 문서가 유효한 것인지 체크하는 문서.
XML은 DTD 혹은 XSD로 필요한 요소 혹은 속성을 파악할 수 있다.
(즉 XML에서 사용되는 Element, Attribute 등에 대한 정보를 기술한 문서)

DTD

DTD(Document Type Definition)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE food [
<!ELEMENT food (name,type,cost)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT cost (#PCDATA)>
]>
<food>
    <name>상추</name>
    <type>야채</type>
    <cost>2000</cost>
</food>
```

출처 : http://www.tcpschool.com/xml/xml_dtd_intro

data.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE food SYSTEM "food.dtd">
<food>
    <name>상추</name>
    <type>야채</type>
    <cost>2000</cost>
</food>
```

food.dtd

```
<!ELEMENT food (name,type,cost)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT type (#PCDATA)>
<!ELEMENT cost (#PCDATA)>
```

XSD

XSD(XML Schema Definition)

food.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<food
xmlns="http://codingsam.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://codingsam.com food.xsd">
    <name>상추</name>
    <type>야채</type>
    <cost>2000</cost>
</food>
```

http://www.tcpschool.com/xml/xml_xsd_intro

food.xsd

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    targetNamespace="http://codingsam.com"
    xmlns="http://codingsam.com"
    elementFormDefault="qualified">

<xs:element name="food">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="type" type="xs:string"/>
            <xs:element name="cost" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>

</xs:schema>
```

XSD

해당 사이트에서는 xml의 xsd를 생성해준다.
https://www.freeformatter.com/xsd-generator.html

해당 사이트에서는 xml이 xsd에 유효한지 확인해준다.
https://www.freeformatter.com/xml-validator-xsd.html

YAML이란?

YAML
YAML Ain't Markup Language
Yet Another Markup Language

YAML은 JSON의 상위호환으로 만들어진 마크업 언어다.
그런데 마크업 언어처럼 안생겼다!
사람이 보기 좋게 생겼다. 굳ㅋ

YAML이란?

```
#!syntax yaml
name: Rust
on:
  push:
    branches: [master]
  pull_request:
    branches: [master]
env:
  CARGO_TERM_COLOR: always
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v2
    - name: Build
      run: cargo build --verbose
    - name: Run tests
      run: cargo test --verbose
```

- 들여쓰기 원칙
- 보기 편함

➔ 여러 프레임워크, CI툴에서 설정 파일로 사용

- 데이터 직렬화(serialization) 불편

➔ 데이터 전송, API 등에서는 거의 사용하지 않음

YAML이란?

https://www.json2yaml.com/convert-yaml-to-json

해당 사이트에서 JSON과 YAML의 차이점을 알 수 있다.

YAML

```yaml
---
# <- yaml supports comments, json does not
# did you know you can embed json in yaml?
# try uncommenting the next line
# { foo: 'bar' }

json:
  - rigid
  - better for data interchange
yaml:
  - slim and flexible
  - better for configuration
object:
  key: value
  array:
    - null_value:
    - boolean: true
    - integer: 1
    - alias: &example aliases are like variables
    - alias: *example
paragraph: >
  Blank lines denote

  paragraph breaks
content: |-
  Or we
  can auto
  convert line breaks
  to save space
alias: &foo
  bar: baz
alias_reuse: *foo
```

JSON

```json
{
  "json": [
    "rigid",
    "better for data interchange"
  ],
  "yaml": [
    "slim and flexible",
    "better for configuration"
  ],
  "object": {
    "key": "value",
    "array": [
      {
        "null_value": null
      },
      {
        "boolean": true
      },
      {
        "integer": 1
      },
      {
        "alias": "aliases are like variables"
      },
      {
        "alias": "aliases are like variables"
      }
    ]
  },
  "paragraph": "Blank lines denote\nparagraph breaks\n",
  "content": "Or we\ncan auto\nconvert line breaks\nto save space",
  "alias": {
    "bar": "baz"
  },
  "alias_reuse": {
    "bar": "baz"
  }
}
```

참고자료

http://www.tcpschool.com/xml/xml_dtd_intro
https://www.youtube.com/watch?v=caILr5oH4p8
https://ko.wikipedia.org/wiki/XML_%EC%8A%A4%ED%82%A4%EB%A7%88_(W3C)
https://aws.amazon.com/ko/what-is/xml/
https://www.youtube.com/watch?v=55FrHTNjTCc&t=380s

https://www.redhat.com/ko/topics/automation/what-is-yaml
https://yaml.org/
https://luran.me/397
https://velog.io/@jnine/YAML%EC%9D%B4%EB%9E%80
https://namu.wiki/w/YAML

Thank You.