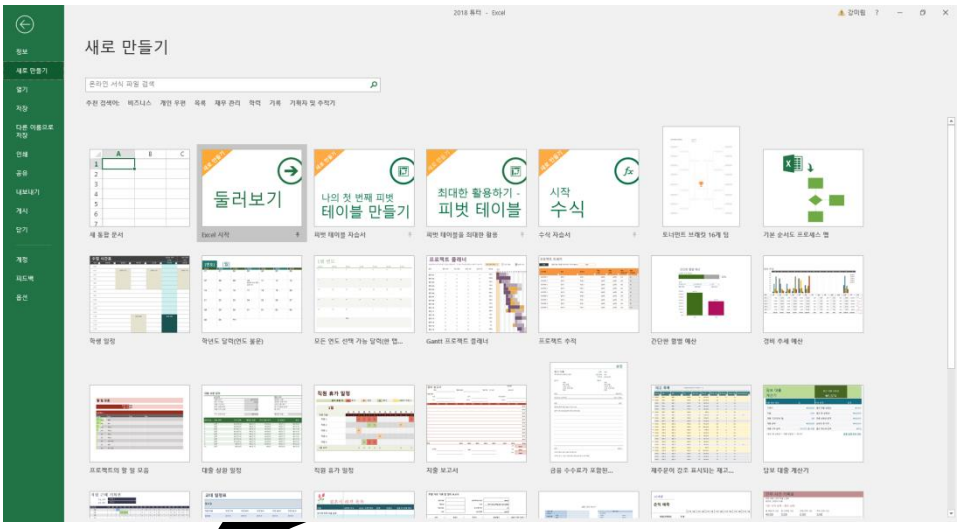
[정규과정] Unity3D를 이용한 4시간으로 게임 만들기

주제 : Unity에서 csv 파일 연동하는 법

유니티에서 엑셀 파일인 csv 파일을
읽어 들이는 방법에 대해 알아봅니다.



튜터 이주영

엑셀 파일을 csv 파일로 내보내고
이를 유니티로 읽어 들이는 과정에 대해서 알아보겠습니다.

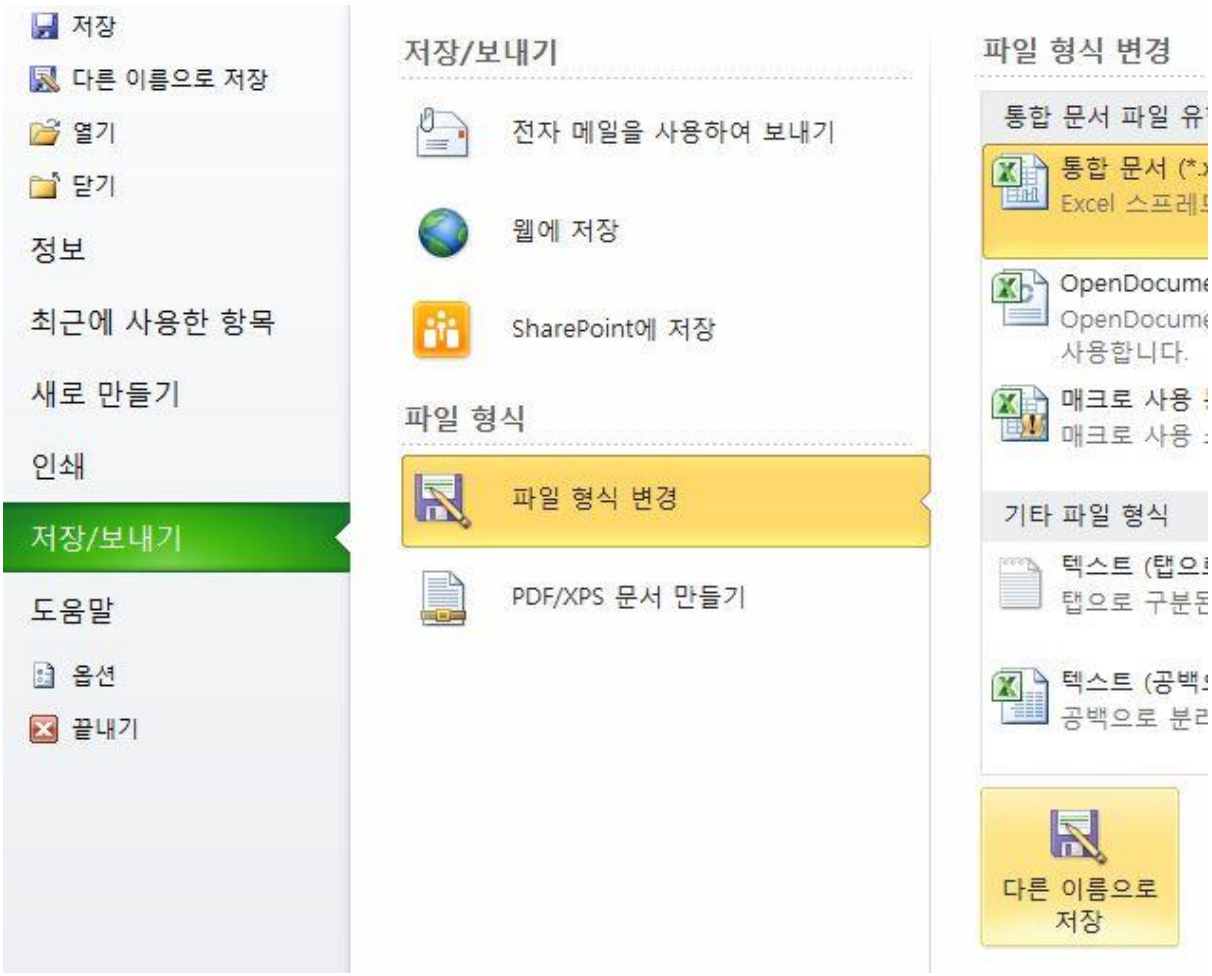
[액셀 데이터를 csv 파일로 내보내기]

	A	B	C	D
1	EXP	Bonus	Text	
2	10	0.1	Beginner	
3	20	0.2	Beginner	
4	30	0.3	Beginner	
5	40	0.4	Beginner	
6	50	0.5	Beginner	
7	60	0.6	Beginner	
8	70	0.7	Beginner	
9	80	0.8	Beginner	
10	90	0.9	Beginner	
11	100	1	Beginner	
12	110	1.1	Beginner	
13	120	1.2	Beginner	
14	130	1.3	Beginner	
15	140	1.4	Beginner	
16	150	1.5	Beginner	
17	160	1.6	Beginner	

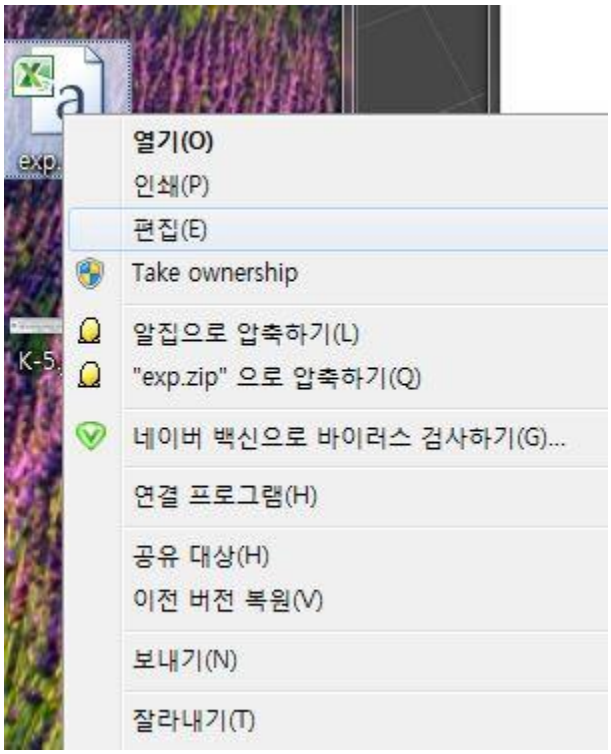
위와 같은 데이터 테이블을 작성해 보았습니다.

여기서 중요한 점은
가장 위의 한 줄은 header이며,
엑셀의 가장 윗줄처럼
하위 항목들이 무엇인지를 알려주는 부분이라는 점입니다.

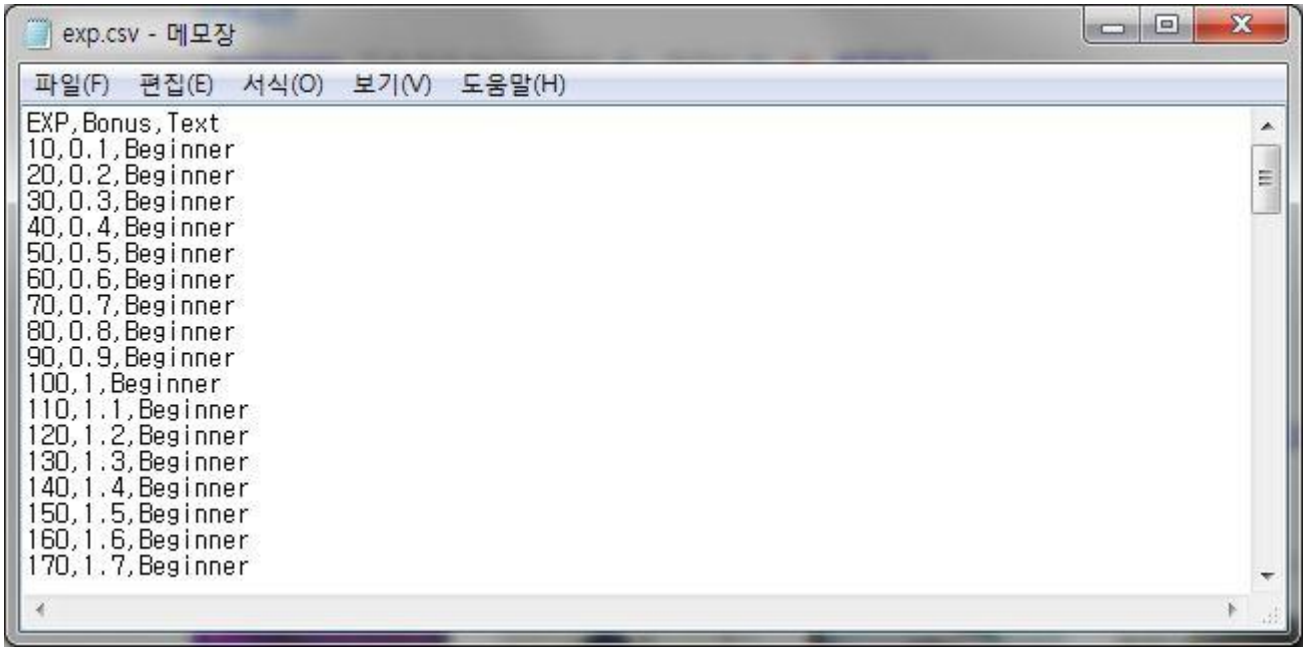
저장/보내기 - 파일 형식 변경 - 다른 이름으로 저장



파일 형식을 CSV 로 선택을 하고 저장해 줍니다.

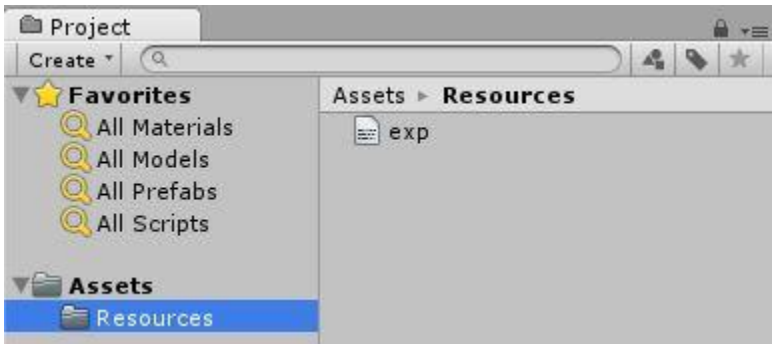


이렇게 생성된 파일을 메모장으로 열어보거나 편집을 눌러보면



위와 같이 값이 들어가 있음을 확인할 수 있습니다.
가장 윗줄은 헤더이고 콤마(,)를 통해 각 값들을 **구분해** 주는 형식입니다.

[유니티로 CSV 파일을 읽어보자]



Assets 폴더 아래에 Resources 라는 폴더를 만들어 주고
위에서 만들었던 CSV 파일을 넣어 줍니다.

```
using UnityEngine;
using System;
using System.Collections;
using System.Collections.Generic;
using System.Text.RegularExpressions;

public class CSVReader
{
    static string SPLIT_RE = @",(?=(?:[^""]*""[^""]*"")*(?![^""]*""))";
    static string LINE_SPLIT_RE = @"\r\n|\n\r|\n|\r";
    static char[] TRIM_CHARS = { '\"' };

    public static List<Dictionary<string, object>> Read(string file)
    {
        var list = new List<Dictionary<string, object>>();
        TextAsset data = Resources.Load (file) as TextAsset;

        var lines = Regex.Split (data.text, LINE_SPLIT_RE);

        if(lines.Length <= 1) return list;

        var header = Regex.Split(lines[0], SPLIT_RE);
        for(var i=1; i < lines.Length; i++) {

            var values = Regex.Split(lines[i], SPLIT_RE);
            if(values.Length == 0 ||values[0] == "") continue;

            var entry = new Dictionary<string, object>();
            for(var j=0; j < header.Length && j < values.Length; j++ ) {
                string value = values[j];
                value = value.TrimStart(TRIM_CHARS).TrimEnd(TRIM_CHARS).Replace("\\", "");
                object finalvalue = value;
                int n;
                float f;
                if(int.TryParse(value, out n)) {
                    finalvalue = n;
                } else if (float.TryParse(value, out f)) {
                    finalvalue = f;
                }
                entry[header[j]] = finalvalue;
            }
            list.Add (entry);
        }
        return list;
    }
}
```

그리고 이 CSVReader.cs 스크립트를 프로젝트에 임포트시켜줍니다.
앞으로 csv 파일을 읽어 들일 때 위의 CSVReader 클래스를 사용하게 됩니다.

출처:http://bravenewmethod.com/2014/09/13/lightweight-csv-reader-for-unity/#comment-7111)

여기까지 했으면 사전 작업은 다 한 것입니다.
CSVReader 클래스도 있고 테스트로 읽어볼 csv 파일도 준비된 것이죠.

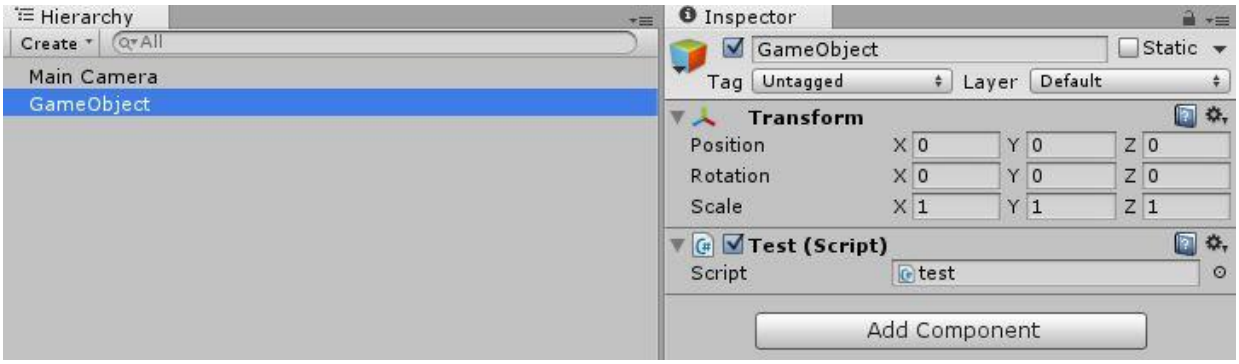
이제는 CSV 파일을 진짜로 읽어보는 일만 남았습니다!

```
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class test : MonoBehaviour {

    public int _exp = 0;

    void Start () {
        List<Dictionary<string,object>> data = CSVReader.Read("exp");

        for(var i=0; i< data.Count; i++){
            Debug.Log("index " + (i).ToString() + " : " + data[i]["EXP"] + " " + data[i]["Bonus"] + " " + data[i]["Text"]);
        }

        _exp = (int)data[0]["EXP"];
        Debug.Log(_exp);
    }
}
```

CSV 파일을 연습삼아 읽어보는 코드를 작성해 보았습니다.
여기서 중요한 것은 CSV 파일의 각 값들에 액세스 하는 방법인데요.

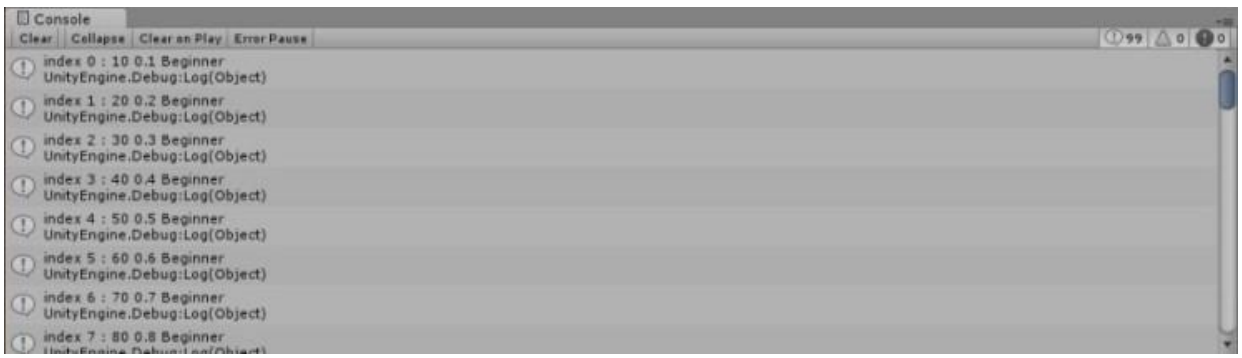
data [i] ["EXP"]

이렇게 하면 i 번째 라인의 EXP 헤더가 가리키는 값을 반환합니다.
data [i] [j] 이런 이차원 배열에서 j 대신 헤더를 사용하고 있다고 보면 되죠.



빈 GameObject 를 생성하여 위에서 작성한 테스트 스크립트를
Inspector에 삽입해 줍니다.

그리고 실행을 해보면



위와 같이 CSV 파일을 불러들여서 0~99라인까지 출력했음을 알 수 있습니다.
CSV로 읽어 들인 값을 변수에 넣어주는 작업을 할 때는
_exp = (int)data[0]["EXP"];
이런 식으로 형변환 하여 넣어주면 됩니다.