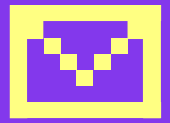
2022 시스템분석설계
포트폴리오

컴퓨터정보공학과
20185029 나승환

머릿말



꽤 긴 시간 휴학하며 윈래도 수업 진도를 잘 따라가지 못 했었는데 전에 배웠던 내용을 알고 있다는 가정하에 수업을 진행하는 과목들이 많아지면서 예상보다 더 어렵게 공부했던 거 같다. 정확한 개념도 잡히지 않은 상태로 수업은 계속 진행되고 학년이 올라가며 해야 할 일도 많아지고 이래저래 힘든 학기였다. 처음 이 시스템분석설계 과목을 수강할 때 졸업 작품과 연계되는 조별로 진행되는 과목이라는 말을 듣고 걱정이 많았다. 연계 과목이라 거의 필수적으로 들어야 하는 과목인데 복학생이라 반에 아는 사람도 없고 그렇다고 수업적으로 아는 것이 많은 것도 아니고 외향적인 성격도 아니라 주변 사람들과 잘 어울리는 것도 아니고 이래저래 힘든 수업이 될거 같다고 생각했다.

팀이 정해지고 졸업 작품을 정하는데 우리 조는 게임을 만들게 되었다. 평소부터 게임을 좋아했던 나로서는 좋은 기회라고 생각했지만 수업시간에 배우지 않는 새로운 것을 따로 공부해야 하니 마냥 좋지만은 않았다. 처음 유니티를 공부하며 생각보다 더욱 어려운 난이도때문에 결국 유니티는 기초적인 것만 하고 나는 데이터베이스쪽을 담당하게 되었다.

이 포트폴리오에서는 실습보다는 개념같은 기본기를 위주로 다룰 것이다. 처음 시작한다는 생각으로 이 시스템분석설계 과목에서 사용하는 깃허브를 수업 시간에 배운 것보다 좀 더 알아보고, 졸업 작품을 만들 때 사용되는 유니티와 데이터베이스도 알아볼 예정이다.

목차

Chapter 01

깃과 깃허브

Level 1 깃

깃이란?

깃(Git) - 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 **분산 버전 관리 시스템**이다.

※깃을 사용하는 이유는?

깃은 파일을 수정할 때마다 그 데이터를 기록하여 필요할 때 이전 버전으로 파일을 복구할 수 있게 해주며, 이를 분산시켜 여러 개발자가 협업하여 프로젝트를 효율적으로 관리할 수 있도록 도와준다. 소스코드 관리 도구 중 Git을 사용하는 이유는 로컬저장소를 이용한 빠른 퍼포먼스와 브랜치를 통한 효율적인 협업에 있습니다.

Level 1 깃

기본 용어 정리

Repository(저장소) - 파일이나 디렉토리를 저장하는 장소
local repo(sitory) - 개인PC에 존재하는 저장소로 working directory, index, head로 이루어져있다.
remote repo(sitory) - 원격 저장소이다. 파일이 원격 저장소 전용 서버에서 관리하고 공유가 가능하다.

Commit - 파일 및 폴더의 추가/변경 사항들을 저장할 때(기록할 때) 사용
사용자는 언제든지 커밋한 시점으로 되돌아 갈 수 있다.

Branch - 독립된 directory로 보통 새로운 기능을 개발할 때 따로 만들어서 개발한다.

Merge - branch의 변경사항을 다른 branch의 반영할 때 사용한다

Fetch - 원격 저장소의 최신 이력을 확인 및 업데이트 할 수 있다.

Level 1 깃

기본 용어 정리

Push - commit한 내용을 remote repo에 올린다.

Pull - 원격 저장소의 내용을 local repo(로컬 저장소)에 반영한다.
fetch + merge의 기능이다.

Pull Request - branch에서 완료된 작업을 프로젝트를 같이하는 사람들끼리 리뷰하고
master로 합치도록 요청하기 위해 사용된다.

Rebase - 두 branch를 합칠 때 사용한다. merge랑 비슷하지만 다르다.

Reset - 이전 버전(또는 특정 commit)으로 돌아갈 때 사용한다.

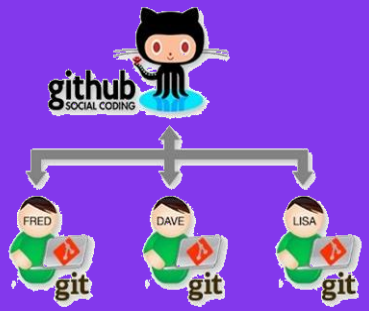
Level 2 깃허브

깃허브란?

깃허브(Github) - 분산 버전 관리 툴인 깃 저장소 호스팅을 지원하는 웹 서비스이다.

※깃과 깃허브의 차이점

Git은 버전 관리 '**프로그램**'이고 Github는 버전 관리, 소스 코드 공유, 분산 버전 제어 등이 가능한 원격 '**저장소**'이다. 그리고 Git은 텍스트 명령어 입력 방식인데 반해, Github는 그래픽 유저 인터페이스(GUI)를 제공합니다.



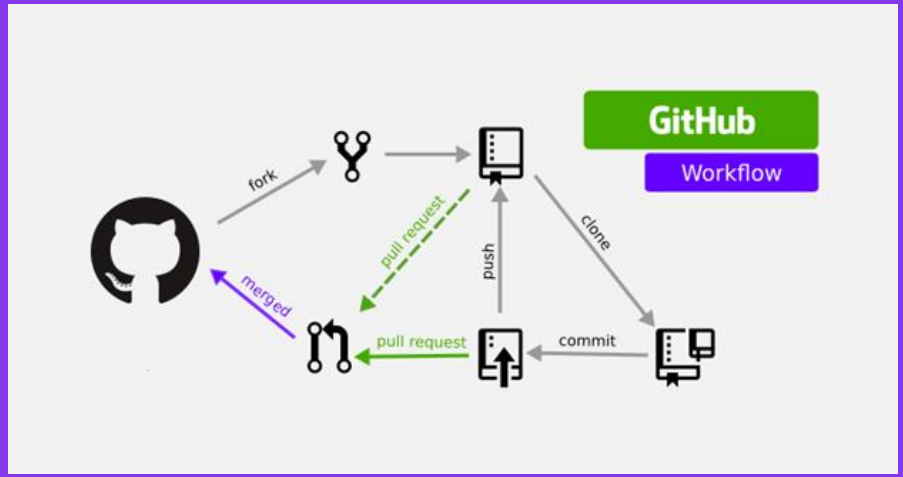
Level 2 깃허브

기본 용어 정리

fork - 다른 깃허브 저장소(오픈 소스 프로젝트)를 복사하는 작업

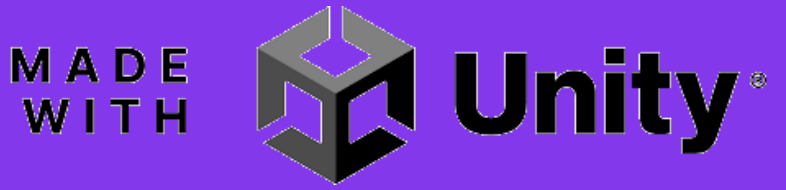
upstream - 오픈소스 프로젝트의 저장소

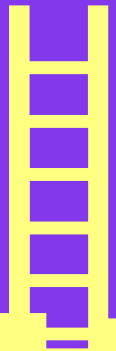
origin - 나의 저장소



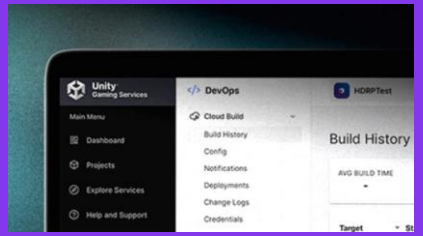
Chapter 02

Unity





Level 1 유니티



유니티란?

Unity - 3D 및 2D 비디오 게임의 개발 환경을 제공하는 게임 엔진이자, 3D 애니메이션과 건축 시각화, 가상현실(VR) 등 인터랙티브 콘텐츠 제작을 위한 통합 제작 도구이다. 게임 개발에 사용되는 언어는 C#과 자바스크립트(UnityScript)를 지원한다.

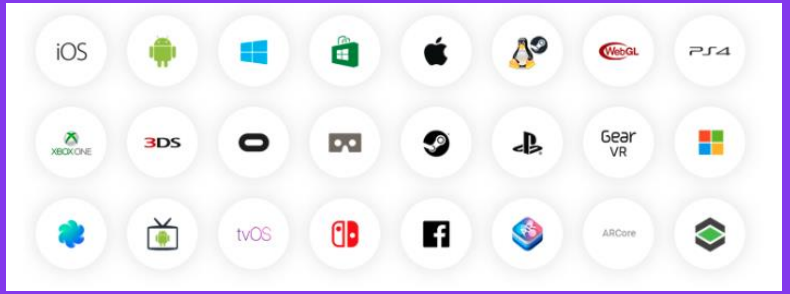
유니티의 특징

1. 제작 워크플로에 맞게 확장되는 올인원 에디터

작업물의 미리 보기를 실시간으로 간편하게 볼 수 있는 플레이 모드 등, Unity 에디터에는 개발 주기에 빨리 편집하고 반복하기 위해 사용할 수 있는 여러 툴이 있다.

Ex) 올인원(all-in-one) 에디터, 2D & 3D, AI 길찾기 도구, Efficient workflows 등

Level 1 유니티



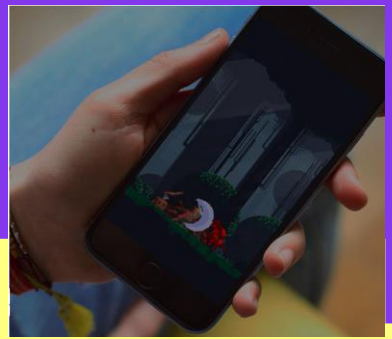
유니티의 특징

2. 멀티 플랫폼 지원

25개 이상의 플랫폼 모바일, 데스크탑, 콘솔, TV, VR, AR, 웹에 이르기까지 기존 제작 엔진 대비 더 강화된 플랫폼을 지원한다.

3. 모듈형 런타임

Unity의 새로운 모듈형 런타임을 사용하여 용량이 작고 빠르게 구동되는 소규모 인스턴트 게임을 제작할 수 있다. 자유롭게 파일 크기 조정이 가능하며 이미 익히 사용하고 있는 에디터로 작업이 가능하다.



Level 1 유니티

유니티의 특징

4. 리얼타임 개발에 CAD 데이터 활용

3D 데이터 최적화 솔루션인 PiXYZ 소프트웨어와 협업하여 대규모 CAD 어셈블리를 Unity에 신속하게 임포트, 관리 및 최적화하여 실시간 시각화 프로젝트를 지원하는 데 필요한 모든 기능을 제공한다. PiXYZ를 사용하면 CAD 데이터 준비 작업이 쉽고 간편해지므로 소스에 관계없이 모든 CAD 파일을 Unity에 최적화할 수 있다.

5. 빠른 협업 작업

Unity Teams를 사용해 협업과 간단한 워크 플로우 기능을 통해 크리에이티브 팀이 서로 효율적으로 작업할 수 있게 한다. **저장, 공유, 동기화** 간단한 버전 제어 및 클라우드 스토리지를 사용하여 Unity와 완벽하게 통합할 수 있다.

Level 1 유니티

유니티 기본 용어 정리

프로젝트(Project) - 씬의 상위 개념, 우리가 흔히 사용하는 프로젝트의 의미와 동일하다.

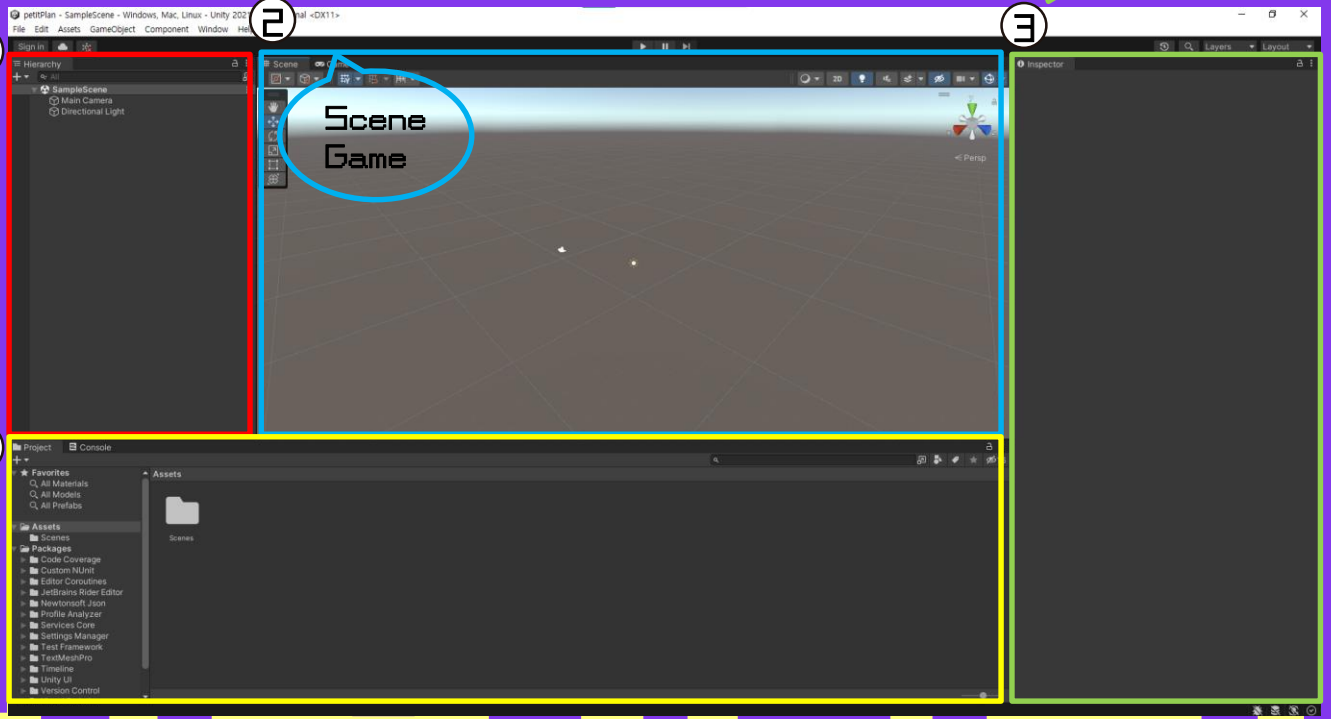
씬(Scene) - 프로젝트를 구성하는 장면 하나하나를 지칭한다.

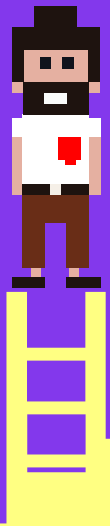
에셋(Asset) - 씬 내부에 있는 오브젝트

Ex)연극으로 예를 들면 연극 제목이 '프로젝트', 연극의 각 장면에 해당하는 것이 '씬', 등장인물, 소도구를 나타내는 것이 '에셋'이다.

Level 2 유니티 구성 요소

유니티 에디터의 화면 구성



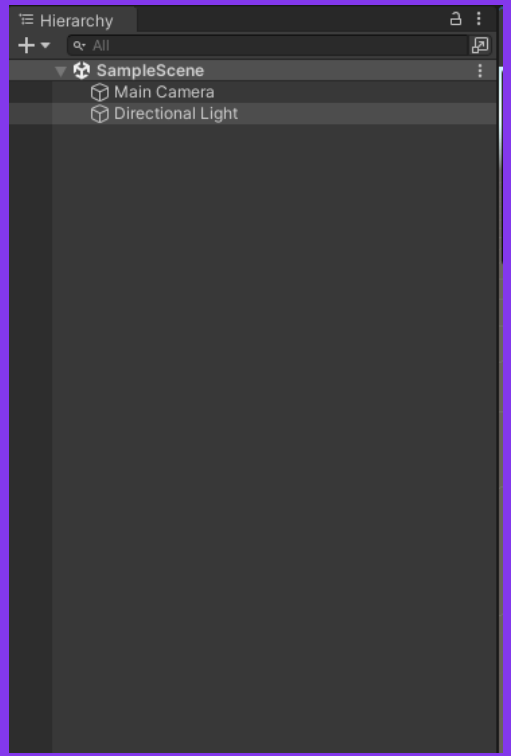


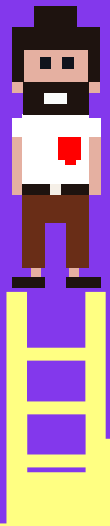
Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

1. Hierarchy

'하이어라키'라고도 부르며, scene에 배치된 모든 오브젝트가 표시되는 화면이다. 간단하게 오브젝트를 생성할 수도 있으며 삭제, 계층 구조도 수정할 수 있다. 아무 설정도 안한 기본으로 플레이어의 시점이 되는 'Main Camera'와 햇빛의 방향이 되는 'Directional Light'가 있다.



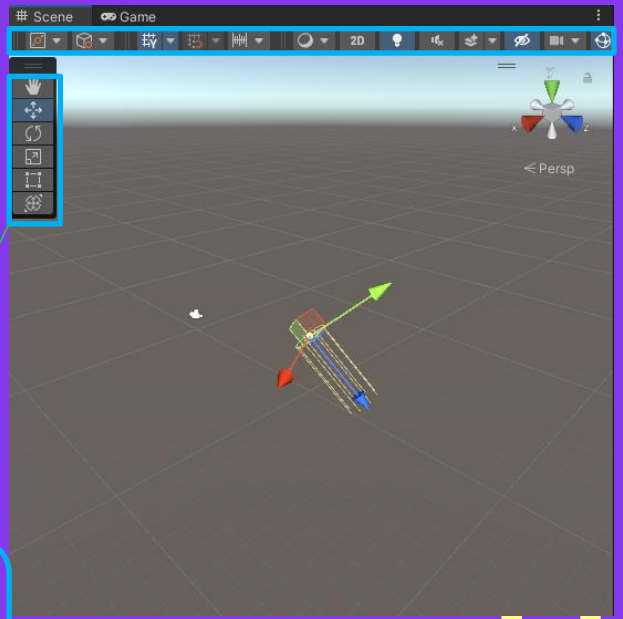


Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

2. Scene & Game

Scene
제작하고 있는 게임의 오브젝트가 표시되어 배치, 크기 조절 등등 확인할 수 있다. 유니티로 게임을 제작한다면 가장 많이 보게 되는 뷰이다.
표시된 툴바에서 다양한 기능을 이용할 수 있다.



변환 도구(Transform Tools)
선택 중인 게임 오브젝트를 이동, 회전, 확대, 축소 할 수 있다.

Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

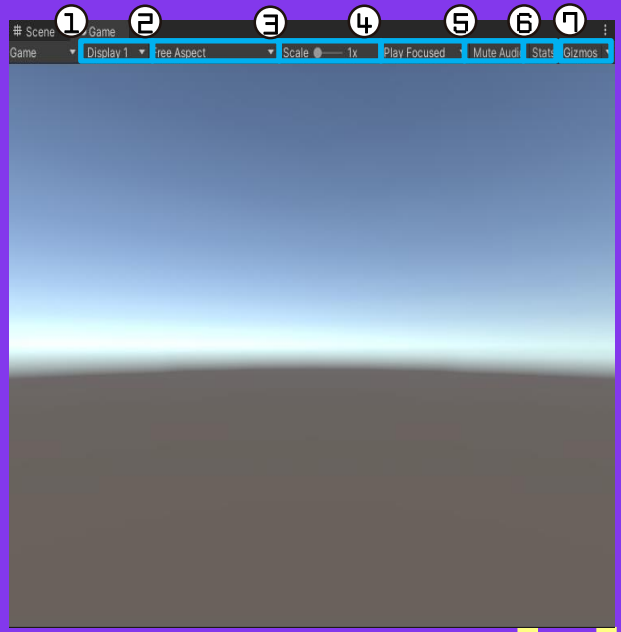
2. Scene & Game

Game

실제로 게임을 시뮬레이션 할 수 있다. 설정한 카메라로 인해 뷰가 보이며, 자신이 원하는 내용으로 흘러가는지, 버그가 있는지 확인할 수 있다.

1) Display(디스플레이)

Display는, 만약 여러 대의 카메라가 있고, 각각의 카메라에게 Display를 다르게 설정해줬다면, 클릭하는 곳이다. 기본적으로 Display1에 설정이 되어 있다.



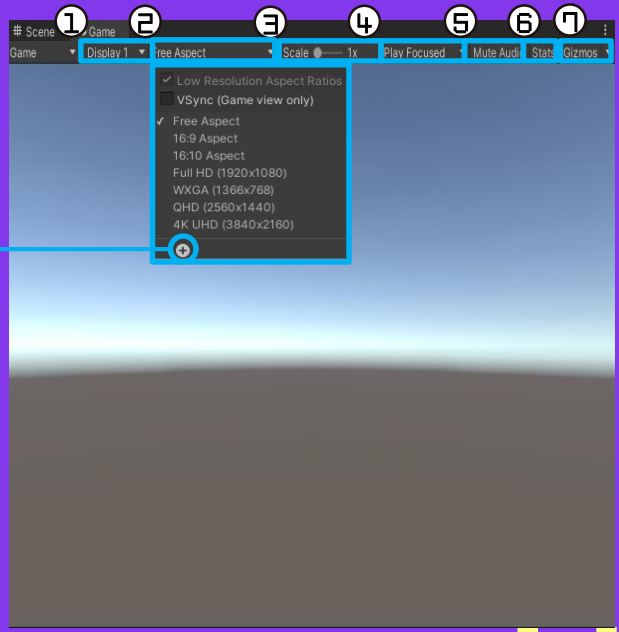
Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

2. Scene & Game

2) Aspect(해상도 지정 창)

Game View를 어떤 해상도 비율로 볼 것인 지를 설정할 수 있는 창입니다. PC 버전에선, 기본적으로, 5:4, 4:3, 3:2, 16:10, 16:9 비율이 있고 1024*768 해상도로 되어 있고, 클릭 후 밑에 + 버튼을 클릭하시면, 사용자 지정 해상도를 설정할 수가 있습니다.



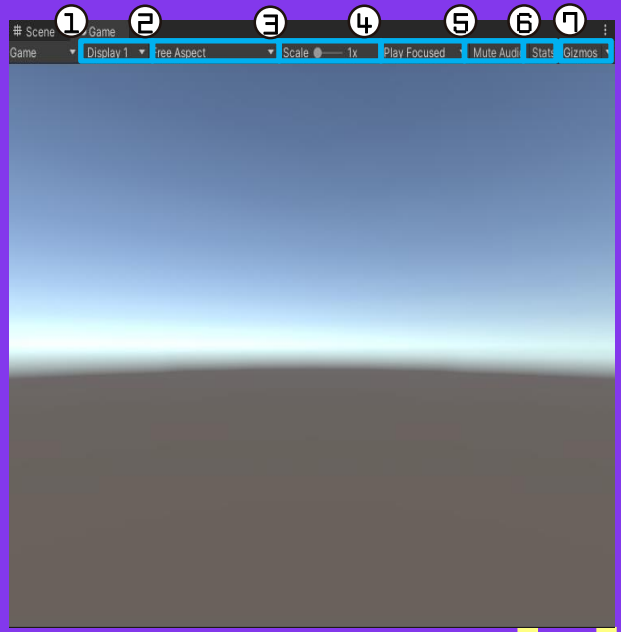
3) Scale Slider(크기 슬라이더)

게임 뷰를 몇 배율로 볼 것인지를 슬라이더 형식으로 만들어 놓은 것이다. 왼쪽으로 갈 수록 0.1 배율 씩 낮아지고, 오른쪽으로 갈 수록 0.1배율 씩 높아진다. 최대 5배까지 올라간다.

Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

2. Scene & Game



4) Maximize on Play(최대화 시작)

Maximize On Play가 활성화가 되어 있다면, 에디터 창을 100%로 최대화 시켜서, 보여지게 된다.

5) Mute Audio(오디오 끄기)

버튼이 활성화가 될 경우, 오디오를 듣지 않는다.

6) Stats(성능 버튼)

버튼을 클릭 할 경우, 게임의 오디오 및 그래픽에 대한 렌더링 정보가 출력됩니다. 즉, 게임의 성능을 간단하게 측정할 수 있게 됩니다.

7) Gizmo(기즈모)

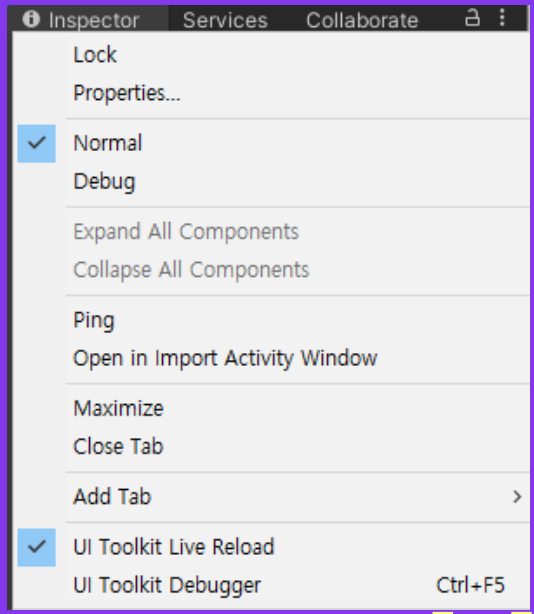
버튼을 클릭하면, 스크립트나 컴포넌트 등을 끄고 키면서 렌더링을 무효화하거나 렌더링을 하거나 할 수가 있다.

Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

3. Inspector

오브젝트의 구성이나 설정 및 소리, 동작, 스크립트를 확인 및 편집을 할 수 있다. 또한 Scene View에서 간단하게 설정한 각도, 크기, 위치를 수치로 확인할 수 있으며 변형도 가능하다.



1) Lock

현재 component로 계속 보이게 지정

2) Normal - Debug

Normal과 Debug에 해당하는 창 모드로 변경

Level 2 유니티 구성 요소

유니티 에디터의 화면 구성

3. Inspector

3) Expand All Components 세부 정보 펴기

4) Collapse All Components 세부 정보 접기

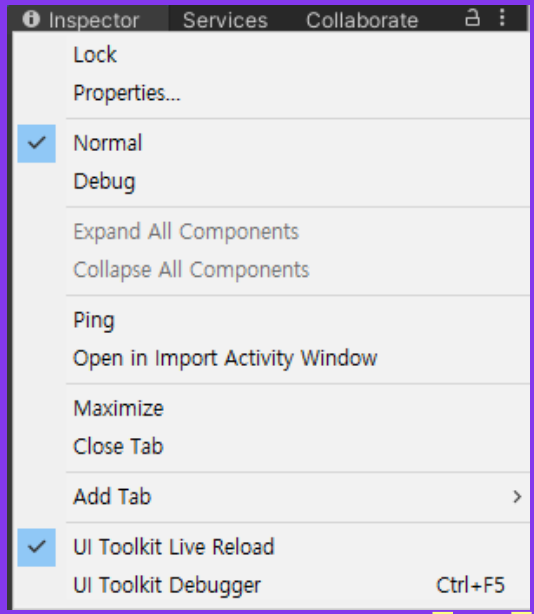
5) Ping Hierarchy에 현재 사용 중인 Scene을 표시해준다.

6) Open in Import Activity Window
Import Activity를 새로운 창으로 띄운다.

7) Maximize 전체 화면

8) Close Tab 현재 탭(Inspector) 닫기

9) UIElements Debugger & UIToolkit Debugger
유니티 화면 구성이 나오는 창



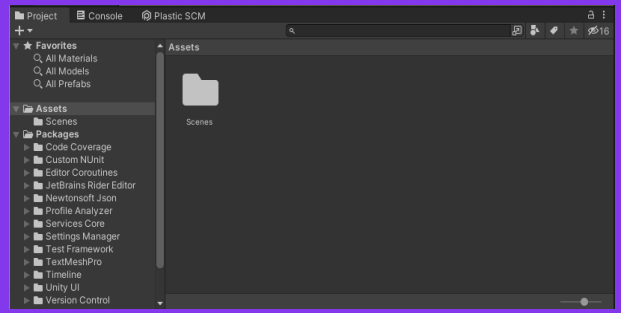
Level 2 유니티 구성 요소

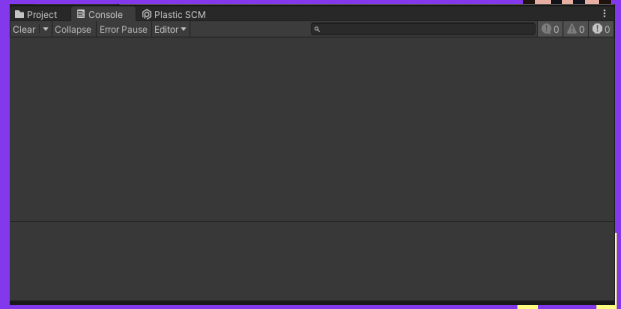
유니티 에디터의 화면 구성

4. Project & Console

Project
사용될 모든 오브젝트와 텍스트, 음원 등 게임에 사용될 자료가 표시된다. 캐릭터나 사물의 모양이나 기준점, 이름 등을 수정할 수 있습니다.

Console
프로그램이 보내는 메시지나, 오류, 경고 등 프로그램 컴파일에 대한 경고가 표시됩니다.





Chapter 03

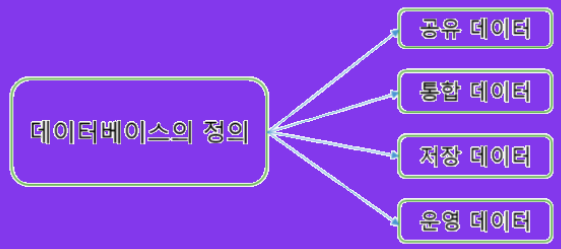
데이터베이스

MySQL

Level 1 데이터베이스

데이터베이스란?

Database - 여러 사람이 공유하여 사용할 목적으로 체계화해 통합, 관리하는 데이터의 집합이다. 논리적으로 연관된 하나 이상의 자료의 모음으로 그 내용을 고도로 구조화함으로써 검색과 갱신의 효율화를 꾀한 것이다. 즉, 몇 개의 자료 파일을 조직적으로 통합하여 자료 항목의 중복을 없애고 자료를 구조화하여 기억시켜 놓은 자료의 집합체라고 할 수 있다.



데이터베이스의 특징

1. 실시간 접근성
2. 지속적인 변화
3. 동시 공유
4. 내용에 대한 참조
5. 데이터 논리적 독립성

Level 1 데이터베이스

DBMS란?

DataBase Management System(DBMS)
데이터베이스를 만들고, 저장 및 관리할 수 있는 기능들을 제공하는 응용 프로그램이다.
종류에 따라 DataBase Server까지 지원하기도 한다. 대표적인 DBMS로는
Oracle, MySQL, MariaDB, Microsoft SQL Server, SQLite 등이 있다.

데이터베이스 모델

1. 계층형 모델(Hierarchical model)
2. 네트워크형 모델(Network model)
3. 관계형 모델(Relational model)
4. 객체-관계형 모델(Object-relational model)
5. 객체형 모델(Object model)

※우리가 자주 사용하는 관계형 데이터베이스는 3번에 해당한다.

Level 1 데이터베이스

관계형 데이터베이스

관계형 데이터베이스는 서로 관련된 데이터 포인트에 대한 액세스를 저장 및 제공하는 데이터베이스 유형이다. 데이터베이스가 표현하고자 하는 개념이 개체로 표현되고 이 개체를 테이블로 표현한다.

관계형 데이터베이스의 특징

1. 데이터의 분류, 정렬, 탐색 속도가 빠르다.
2. 오랫동안 사용된 만큼 신뢰성이 높고, 어떤 상황에서도 데이터의 무결성을 보장해 준다.
3. 기존에 작성된 스키마를 수정하기가 어렵다.
4. 데이터베이스의 부하를 분석하는 것이 어렵다.

Level 1 데이터베이스

관계형 데이터베이스 용어

1. 릴레이션(Relation)
데이터들을 표(Table)의 형태로 표현한 것으로, 구조를 나타내는 릴레이션 스키마와 실제 값들인 릴레이션 인스턴스로 구성된다

2. 열(column)
각각의 열은 유일한 이름을 가지고 있으며, 자신만의 타입을 가지고 있습니다.
이러한 열은 **필드(field)** 또는 속성(attribute)이라고도 불립니다.

3. 행(row)
행은 관계된 데이터의 묶음을 의미한다. 한 테이블의 모든 행은 같은 수의 열을 가지고 있다.
이러한 행을 튜플(tuple) 또는 **레코드(record)**라고도 부른다.

4. 도메인(Domain)
도메인은 하나의 속성이 취할 수 있는 같은 타입의 원자값들의 집합이다. 도메인은 실제 속성 값이 나타날 때 그 값의 합법 여부를 시스템이 검사하는데 에도 이용된다.

Level 1 데이터베이스

관계형 데이터베이스 용어

4. 키(key)

테이블에서 행의 식별자로 이용되는 열을 키(key) 라고 한다.

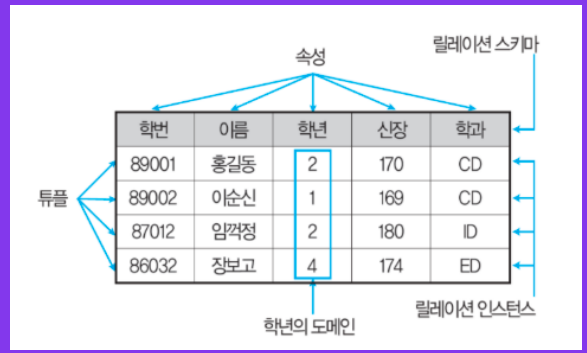
※키의 종류 1) 기본 키(primary key) 2) 외래 키(foreign key) 3) 후보 키 4) 대체 키

5. 관계(relationship)

테이블 간의 관계는 관계를 맺는 테이블의 수에 따라 나눌 수 있다.

1) 일대일(one-to-one) 관계
2) 일대다(one-to-many) 관계
3) 다대다(many-to-many) 관계

관계형 데이터베이스에서는 이러한 관계를 나타내기 위해 외래 키를 사용한다.



6. 스키마(schema)

자료의 구조, 표현 방법, 자료 간의 관계를 형식 언어로 정의한 구조이다.

스키마는 테이블의 각 열에 대한 항목과 타입뿐 아니라 기본 키와 외래 키도 나타내야 합니다.

Level 2 SQL

SQL이란?

SQL(Structured Query Language)
관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어이다.

SQL 구문 종류

SQL 구문의 종류는 기능에 따라 다음 세 가지로 구분한다.

1. 데이터 정의 언어(DDL : Data Definition Language)

2. 데이터 조작 언어(DML : Data Manipulation Language)

3. 데이터 제어 언어(DCL : Data Control Language)

Level 2 SQL

1. 데이터 정의 언어(DDL : Data Definition Language)

데이터베이스, 테이블, 뷰, 인덱스, 도메인, 제약 조건 등 각종 개체를 생성, 수정, 삭제 등을 관리하기 위한 명령어이다.

종류	기능
Create	데이터베이스, 테이블 뷰 인덱스, 도메인, 제약조건 등 각종 개체를 생성
Drop	데이터베이스, 테이블 뷰 인덱스, 도메인, 제약조건 등 각종 개체를 삭제
Alter	이미 생성된 테이블과 도메인을 수정

2. 데이터 조작 언어(DML : Data Manipulation Language)

데이터 정의어로 정의된 데이터베이스 내의 데이터를 조작하는 명령어로 레코드를 조회, 삽입, 수정, 삭제하는데 사용한다.

종류	기능
Select	테이블의 데이터 검색
Insert	테이블의 새로운 레코드 삽입
Update	테이블의 레코드 수정
Delete	테이블의 레코드 삭제

Level 2 SQL

3. 데이터 제어 언어(DCL : Data Control Language)

데이터베이스에 대한 정확성과 안정성을 위해 개체, 사용자, 작업 수행 등을 관리하는 명령어이다. 데이터 제어 언어는 데이터베이스를 공용으로 사용하기 위한 데이터 제어를 정의하고 기술하는 언어로서 관리를 위한 도구이다. 데이터 제어 언어는 주로 관리자가 사용하는 언어로 데이터 보안, 무결성, 데이터 회복 등을 수행하기 위한 목적으로 사용된다.

종류	기능
Grant	사용자 계정이나 역할에 대하여 개체의 접근 권한이나 SQL 사용 권한을 부여
Deny	Grant와 반대되는 의미로 개체에 접근하지 못하게 하거나 SQL을 수행하지 못하도록 거부
Revoke	이미 Grant 또는 Deny로 설정된 권한을 제거

Chapter 04

실습

NAME	TYPE	NULLABLE
ANIMAL_ID	VARCHAR(N)	FALSE
ANIMAL_TYPE	VARCHAR(N)	FALSE
DATETIME	DATETIME	FALSE
INTAKE_CONDITION	VARCHAR(N)	FALSE
NAME	VARCHAR(N)	TRUE
SEX_UPON_INTAKE	VARCHAR(N)	FALSE

* 실습 문제는 https://programmers.co.kr/ 에서 사용했습니다.

Level 1 SQL 프로그래밍

예제 1 중복 제거

동물 보호소에 들어온 동물의 이름은 몇 개인지 조회하는 SQL 문을 작성해주세요.
(이름이 NULL인 경우는 집계하지 않으며 중복되는 이름은 하나로 칩니다.)

solution.sql

```sql
SELECT COUNT(NAME) COUNT
FROM (SELECT NAME FROM ANIMAL_INS GROUP BY NAME) AS A
WHERE NAME IS NOT NULL
```

실행 결과

COUNT
96

Level 1 SQL 프로그래밍

예제 2 고양이와 개는 몇 마리 있을까

동물 보호소에 들어온 동물 중 고양이와 개가 각각 몇 마리인지 조회하는 SQL문을 작성해주세요.
(이때 고양이가 개보다 먼저 조회해주세요.)

solution.sql

```sql
SELECT ANIMAL_TYPE, COUNT(*) COUNT
FROM ANIMAL_INS
GROUP BY ANIMAL_TYPE
ORDER BY ANIMAL_TYPE
```

실행 결과

ANIMAL_TYPE	COUNT
Cat	15
Dog	85

Level 1 SQL 프로그래밍

예제 3 동명 동물 수 찾기

동물 보호소에 들어온 동물 이름 중 두 번 이상 쓰인 이름과 해당 이름이 쓰인 횟수를 조회하는 SQL문을 작성해주세요. (결과는 이름이 없는 동물은 집계에서 제외하며, 결과는 이름 순으로 조회해주세요.)

sql

```sql
SELECT NAME,COUNT(*) COUNT
FROM ANIMAL_INS
WHERE NAME IS NOT NULL
GROUP BY NAME
HAVING COUNT > 1
ORDER BY NAME
```

실행 결과

NAME	count
Lucy	3
Raven	2

Level 1 SQL 프로그래밍

예제 4 없어진 기록 찾기

입양을 간 기록은 있는데, 보호소에 들어온 기록이 없는 동물의 ID와 이름을 ID 순으로 조회하는 SQL문을 작성해주세요.

solution.sql

```sql
SELECT A.ANIMAL_ID, A.NAME
FROM ANIMAL_INS B
RIGHT JOIN ANIMAL_OUTS A ON A.ANIMAL_ID = B.ANIMAL_ID
WHERE B.ANIMAL_ID IS NULL
```

실행 결과

ANIMAL_ID	NAME
A349480	Daisy
A349733	Allie
A349990	Spice
A362137	*Darcy

Level 1 SQL 프로그래밍

예제 5 보호소에서 중성화 한 동물

보호소에서 중성화 수술을 거친 동물 정보를 알아보려 합니다. 보호소에 들어올 당시에는 중성화 되지 않았지만, 보호소를 나갈 당시에는 중성화된 동물의 아이디와 생물 종, 이름을 조회하는 아이디 순으로 조회하는 SQL 문을 작성해주세요.

solution.sql

```sql
SELECT A.ANIMAL_ID, A.ANIMAL_TYPE, A.NAME
FROM ANIMAL_INS A
LEFT JOIN ANIMAL_OUTS B ON A.ANIMAL_ID = B.ANIMAL_ID
WHERE A.SEX_UPON_INTAKE != B.SEX_UPON_OUTCOME
ORDER BY A.ANIMAL_ID
```

실행 결과

ANIMAL_ID	ANIMAL_TYPE	NAME
A382192	Dog	Maxwell 2
A410330	Dog	Chewy

맺음말

이번 포트폴리오를 작성하면서 1학년 때 사용했던 책을 다시 뒤져보고 자세한 내용을 다시 인터넷에 검색함으로서 알았던 개념은 보다 정확하고 자세하게 알 수 있게 되었고, 잘못 알고 있던 내용을 바로 잡을 수 있었다. 알고 있던 것이 별로 없어서 작성하는데 많은 시간을 사용했지만 아깝지 않은 시간이었던 것 같다. 물론 아직 학기가 끝난 것은 아니고 다음 학기 나아가 졸업 작품도 남았지만 오랫동안 만들어서 그런지 큰 고비를 하나 넘은 느낌이다.

개념에 관련된 내용은 그냥 정보를 적는 거라 찾는데 좀 시간이 걸리고 크게 어려움이 없었지만 마지막 실습 부분에서는 본인이 직접 문제를 해결하는 거라 더욱 시간이 많이 걸렸다. 하지만 문제를 실제로 풀어보면서 앞에 정리한 개념들을 다시 한 번 보게 되고 직접 사용하여 이것들이 어떻게 활용되는지 눈으로 확인 할 수 있었다. 그리고 졸업 과제를 하게 될 때도 도움이 많이 되었다.

마지막으로 분명 나와 같이 복학하여 수업에 잘 따라가지 못하는 사람들이 있을 텐데 실제로 예제를 풀어보는 것 보다 개념 위주의 공부를 하면 좋을 것이라고 말해주고 싶다.
(물론 이것만 해서는 부족할 것이다. 내가 그렇듯이)


Thank
You