1. 유니티 구조

* **라이프 사이클 (Lifecycle)**

(<https://docs.unity3d.com/kr/2023.2/Manual/ExecutionOrder.html>)

* + Awake (첫 씬 시작 시)

- 오브젝트의 새 인스턴스가 생성될 때 호출되는 첫 번째 라이프사이클 함수

- 씬에서 오브젝트마다 한 번만 호출됨

- Start 함수보다 먼저 호출

- 오브젝트가 비활성화일 경우엔 호출되지 않지만 스크립트가 비활성화일 경우엔 호출됨

- 에디터 실행이전 초기화 단계에서 주로 사용

* + Start (첫 프레임 업데이트 전)

- 스크립트 인스턴스가 활성화된 경우, 첫 프레임 업데이트 이전에 호출됨

- 모든 Awake가 수행된 이후 호출됨

- 에디터 실행이후 초기화 단계에서 주로 사용

* + Update

- 프레임당 한 번 호출

- 프레임 업데이트를 위한 주요 작업 함수 (게임 로직)

- 디바이스 성능이나 최적화 상황에 따라 framerate가 변하므로 함수 호출 시간이 매번 달라질 수 있음

- 최대한 가볍게 구현하는 것이 이상적

* + OnDisable (끝낼 때)

- 게임 오브젝트나 스크립트가 disabled 되거나 inactive 될 때 호출됨

* + OnDestroy (오브젝트 Destroy할 때)

- 오브젝트 생존 기간의 마지막 프레임이 업데이트된 후 호출

* **State Machine (상태 머신, Finite State Machine FSM)**

상태 머신은 상태(state)를 기반으로 하여, 동작을 제어하는 방식을 구현하기 위한 디자인 패턴.

* + State : 객체의 동작을 상태로 분류함. Ex) Idle, Move, Attack
  + Transition : 상태와 상태 사이의 전이를 정의함. Ex) Idle에서 Move로 상태가 전이될 때의 조건을 정의.
  + Action : 상태에서 실행될 동작을 정의함. Ex) Attack상태에서 수행할 동작을 정의.

텍스트, 스크린샷, 원, 도표이(가) 표시된 사진

자동 생성된 설명

1. 방치형 게임 만들기 (배경 꾸미기)

* Unity Hub에서 프로젝트 우클릭 > Reveal in finder(Mac기준) / Show in Explorer(Windows기준) > Asset 폴더 > Resources 폴더 생성 > Backgrounds 폴더 생성 > 이미지 넣기
* UI > Image > Backgrounds 폴더에서 찾은 이미지 넣기
* Game > Free Aspect(화면비율조정) > 맞는 비율 선택
* 이미지 Size Width\*Height 조절, (Pos X, Pos Y, Pos Z) 위치설정
* Background Scroll 오브젝트 생성 > Script 생성 후 연결 > Element에 오브젝트(배경이미지) 추가

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* **델타 타임([Time](https://docs.unity3d.com/2023.2/Documentation/ScriptReference/Time.html).deltaTime)**

델타 타임은 “**마지막 프레임으로부터 현재 프레임까지의 간격**”을 의미함. 즉, 이전 프레임이 완료되기까지 걸린 시간임.

프레임 단위로 결과가 도출되는 총 게임에서, 기기 성능 등에 따라 A는 10FPS(Frames Per Second), B는 60FPS로 작업을 수행한다면 A가 쏜 총알이 1초에 10m를 날아가는 동안 B가 쏜 총알은 1초에 60m를 날아가게 되어 공정하지 못한 결과가 나오게 됨. 이를 위해 프레임 간 시간 간격인 델타 타임을 적용하게 됨. 즉, 델타 타임을 곱하여 이동 거리 등을 연산함.

Ex) backgroundOffset[i] -= Time.deltaTime \* scrollSpeed;

* 유니티 단축키 : 화면 움직이기 Q / 오브젝트 이동시키기 W

휴대 전화, 스크린샷, 정보기기, 휴대용 통신 장치이(가) 표시된 사진

자동 생성된 설명

1. **Enum** **(열거형)**

(<https://learn.microsoft.com/ko-kr/dotnet/csharp/language-reference/builtin-types/enum>)

* 열거형은 열거형 상수를 가독성 있게 표현하기 위한 방법
* 기본적으로 열거형의 상수 값은 int 형식이며, 0으로 시작하여 순서에 따라 1씩 증가함

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

* 연결된 상수 값을 명시적으로 지정할 수 있음

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* 명시적 변환을 통해 열거형 형식과 기본 정수 형식을 변환하여 사용 가능

(이 경우 case 구문 2가지는 동일한 의미임)

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

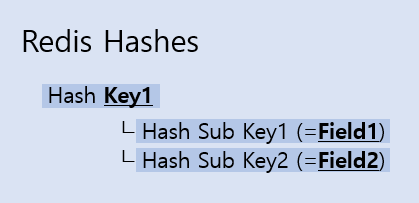
1. 디버깅

* **디버깅(Debugging)** : 프로그래밍 과정 중에 발생하는 오류나 비정상적인 연산, 즉 버그를 찾고 수정하는 것
* 디버깅 방법
  + 프로그램을 몇 가지 단계로 나누어 단계별로 체크
  + 오류가 있는 범위를 좁혀 나가기
  + 입출력이 정상적으로 되었는지, 반복문의 시작과 끝부분, 함수의 시작과 끝부분 등 주요 코드 위주로 체크해 나가면 좋음
  + 중단점(Breakpoint) 설정/해제(F9) : 생성된 중단점에서 [우클릭 – 조건] 또는, 중단점을 생성할 곳에 [우클릭 – 조건부 중단점 삽입]을 실행하면 해당하는 조건에 부합할 때만 중단점에 걸리게 할 수 있음
  + F5 : 디버그 모드로 실행. 중단점을 발견하면 해당 라인 실행 직전에 중단. 한번 더 누르면 다음 중단점이 나타나거나 종료 되기 전까지 실행
  + F10 : 프로시저 단위 실행
  + F11 : 한 단계씩 실행, 함수를 만나면 함수 안으로 들어감
  + Shift+F5 : 디버깅 종료

1. **데이터베이스** **(DB)**

* **RDB (Relational Database, 관계형 데이터베이스)**
  + 데이터를 로우(Row: 행, 줄)와 컬럼(Column: 열, 칸)이라는 일종의 **표 형태**로 저장. 데이터의 종속성은 관계(Relation)로 표현됨. 한 테이블에 있는 모든 로우는 같은 길이의 컬럼을 가지고 있으며 이 컬럼의 구조와 데이터의 관계가 테이블 **스키마(Schema)**로 사전 정의됨.
  + 데이터 무결성을 보장함.
  + SQL (Structured Query Language) 사용
  + MySQL, PostgreSQL, MariaDB, MSSQL(Microsoft SQL Server), Oracle Database, Amazon Aurora 등
* **NoSQL (Not only SQL)**
  + 기존 RDBMS 형태의 관계형 데이터베이스가 아닌 다른 형태의 데이터 저장 기술을 의미하며, 관계형 데이터베이스의 한계를 극복하기 위한 데이터 저장소의 새로운 형태. 예외는 존재하나, 보통 비관계형 데이터베이스라고도 부른다.
  + 명시적인 스키마 없이 동작(유동적), 분산 저장, 수평적 확장 용이 등의 특징을 가지고 있음.
  + Redis, MongoDB, HBase 등
  + Redis 기초
    - Key-Value 형식으로 값을 저장함
    - 메모리 기반의 데이터베이스로, 빠른 속도로 데이터를 저장하고 조회할 수 있으나 데이터 휘발의 단점이 있어 이를 백업 기능으로 보완함
    - 공식사이트 : <https://redis.io/>
    - Redis 기초 명령어
      * SET : SET key value > key에 value를 저장
      * GET : GET key > key값에 해당하는 value를 조회
      * HSET : HSET key field value > 지정한 hash key 하위의 field에 value를 저장. 필드 값이 있을 경우 덮어쓰며, 키가 존재하지 않으면 새 해시키를 생성.

(field는 hash key의 sub key 개념)



* + - * DEL : DEL key > 해당 키를 제거하며, 존재하지 않는 키는 무시됨.
      * KEYS : KEYS \* > 모든 키를 확인.

(KEYS pattern 꼴로 \*, ?, [a-c] 등 glob-style pattern을 모두 사용할 수 있음)

* + - * HKEYS : HKEYS key > hash의 키에 저장된 모든 필드명 확인.

1. 과제

* 방치형 게임 UI 꾸미기