class ObjectPoolingManager 설계

By 남정웅

Index

0. 설계 목표

1) 프로그래머가 아니더라도 유니티상에서 쉽게 사용 할 수 있게 설계
2) 하나의 class에서 풀들을 일괄 관리(인스펙터뷰에서 편집 가능)
3) 하나의 풀을 여러 클래스에서 사용 할 수 있게 설계
4) 게임시작전 유니티 에디터에서 미리 객체들을 생성가능
5) Instantiate와 같이 인스펙터뷰에서 프리팹을 할당하는 형식으로 사용
6) 풀에서 활성화 시킬 객체가 없는 경우 처리방법 선택가능

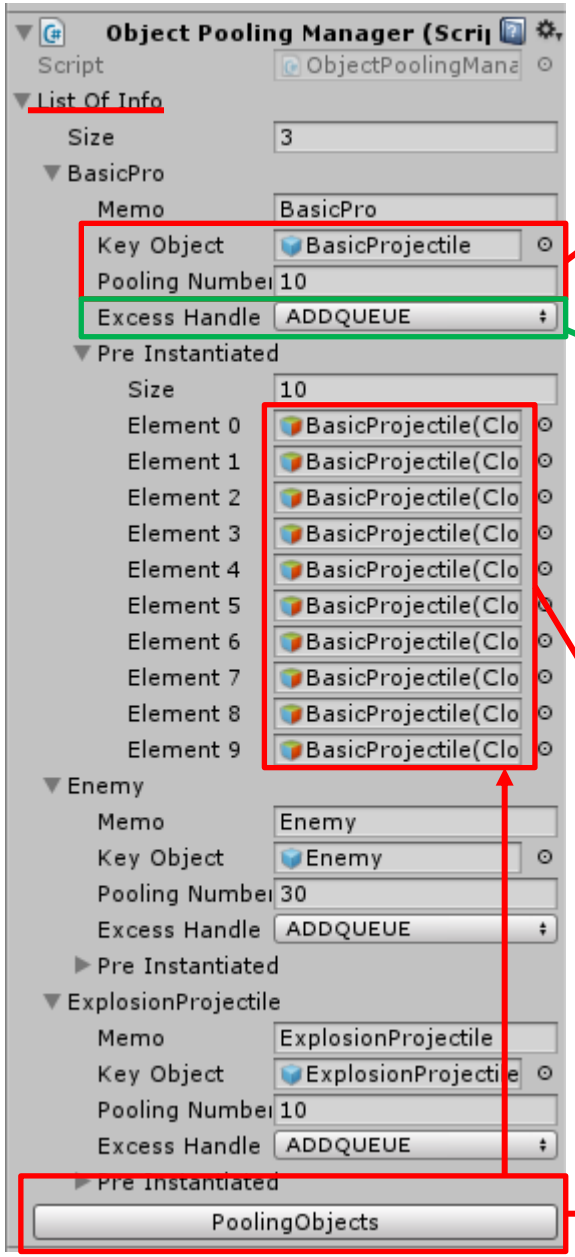
1. class ObjectPoolingManager

1) 풀링 할 객체들의 풀을 모두 이 클래스에서 관리(싱글톤)
2) 풀의 자료구조는 Queue이용
3) Dictionary를 활용하여 하나의 풀을 여러 클래스에서 사용 할 수 있게 설계
4) enum분기를 통해 풀에 있는 객체가 모두 활성화 중일 때 어떻게 처리할지 구현
5) 일반적인 Instantiate처럼 인스펙터뷰에서 프리팹을 할당해서 사용
6) 오브젝트들을 풀링 할 때 class IsReusing 컴포넌트를 추가하며 IsReusing 컴포넌트의 유무에 따라 객체가 사라 질 때 ObjectPoolingManager의 DisableOrDestroy() 함수를 통해 비활성화 시킬지 파괴 할지 결정

*풀을 Queue로 만드는 이유

1) 객체를 활성화 시킬 때 Dequeue()후 Enqueue()로 큐의 가장 뒤쪽에 배치
2) 일반적으로 먼저 활성화된 객체가 먼저 비활성화 되는 경우가 많음
3) 다시 비활성화된 객체를 찾을때 Queue의 첫번째 요소가 비활성화 되어 있을 가능성이 높음
4) 이 로써 Queue의 순회를 최소한으로 할 수 있는 기대를 할 수 있음

2. 인스펙터뷰에서 풀 편집



미리 만들어 놓고 싶은
오브젝트와 그 숫자

풀에 활성화 시킬 객체가 없을 경우 처리방법 선택

List<InfoOfPool> listOfInfo

InfoOfPool

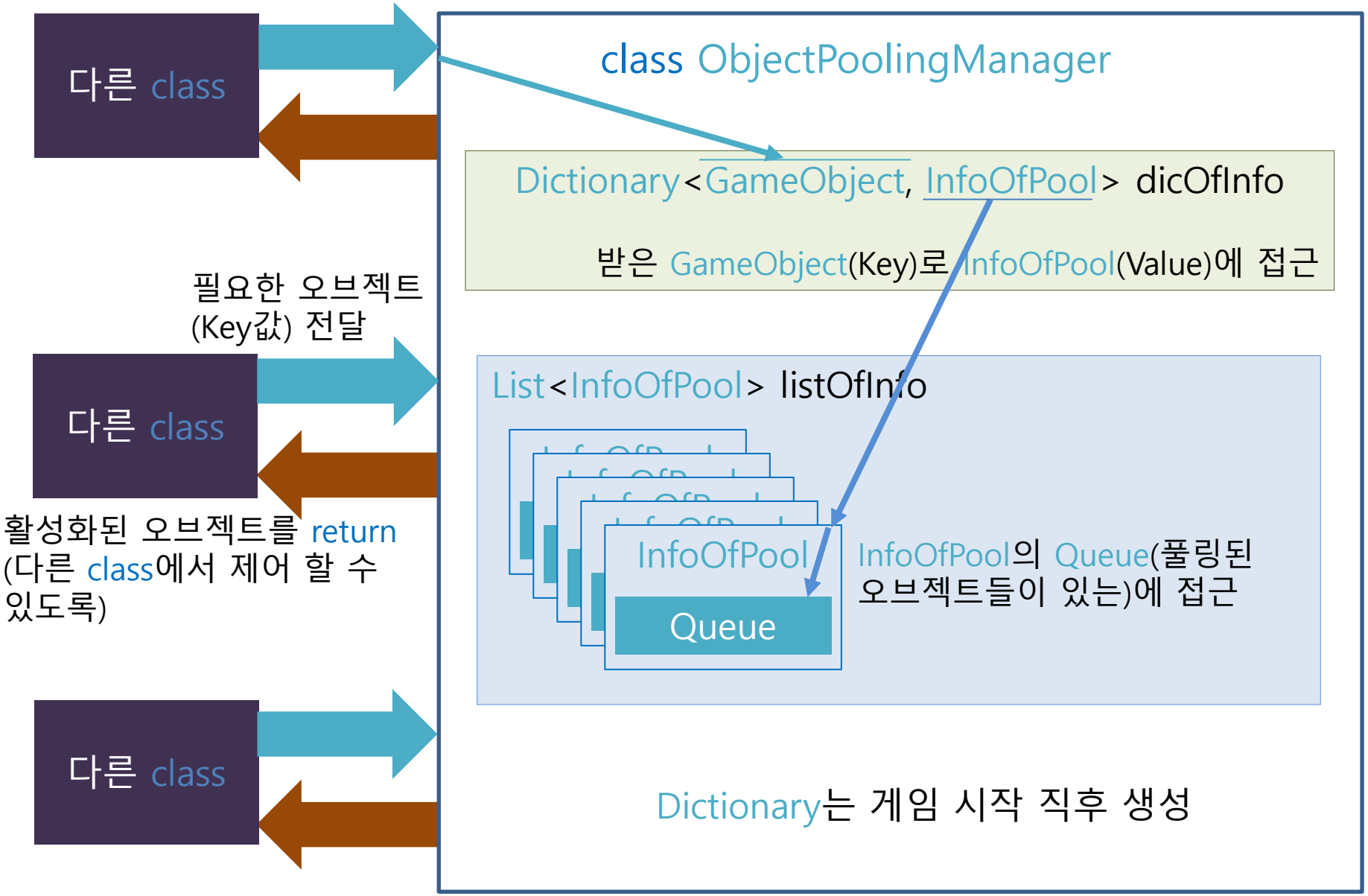
Queue

(인스펙터뷰에는 표시되지 않지만
InfoOfPool에는 Queue도 선언되어 있음)

생성된 인스턴스들은 게임 시작 후
Queue로 된 풀에 Enqueue

버튼을 누르면 KeyObject와 PoolingNumber의 값을
토대로 미리 풀에 넣을 인스턴스들를 생성함

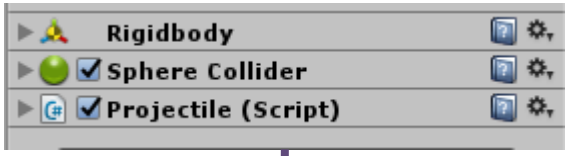
3. 게임 시작 후 다른 클래스에서 사용 할 때



4. 객체가 사라질 때

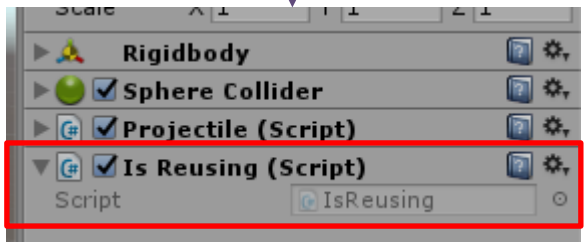
그 자체로는 아무런 기능이 없는, class ObjectPoolingManager에서 풀링 된 객체에 추가 시키며 재사용 가능 여부를 판단하기 위한 컴포넌트용 class

프리팹



오브젝트가 풀링 될 때
(ObjectPooling(…)함수 실행 시)
자동으로 컴포넌트가 추가됨

class ObjectPoolingManager
를 통해 풀링 되었을 때



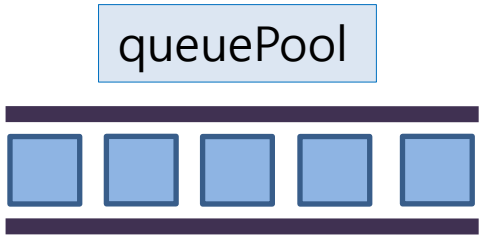
이 객체가 사라 질 때 class ObjectPoolingManager의 DisableOrDestroy 함수를 실행
DisableOrDestroy에서 이 컴포넌트의 유무로 비활성화 시킬지 파괴시킬지 결정

이 방법을 통해 풀을 이용한 방법이 아니라도
일반적인 instantiate을 통한 방법도 사용가능

5. 활성화 시킬 객체가 없을 때

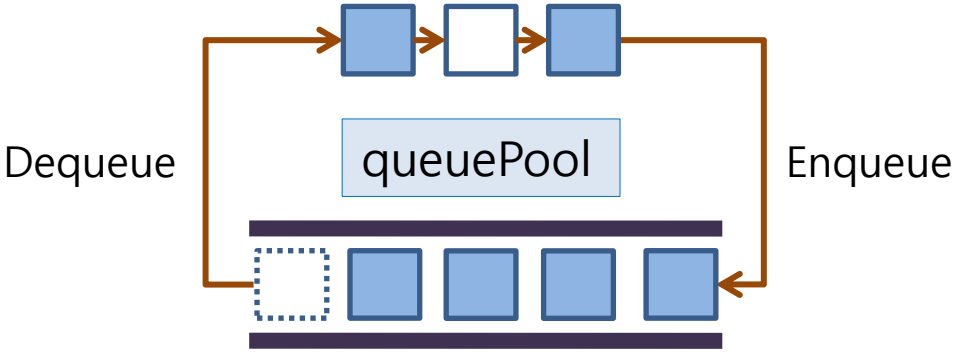
풀에 활성화시킬 객체가 없을 때 처리방법

1) DONOTMAKE



아무런 행동을 하지 않는다

2) DEACTIVEFIRST

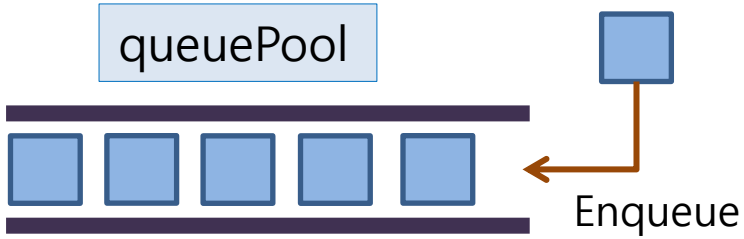


가장 먼저 활성화 되었던 객체(Queue의 가장 앞)를 비활성화 한 후 다시 활성화 (Transform은 다시 초기화)

인스펙터뷰에서 ExcessHandle 선택가능

```
//큐에있는 모든 객체가 활성화되어
//더 이상 활성화시킬 객체가 없을때 어떻게 처리할지 결정
public enum ExcessHandle
{
    //아무런 행동을 하지 않는다
    DONOTMAKE,
    //가장 먼저 활성화됬었던(Queue의 가장 앞) 객체를
    //비활성화 한 후 다시 활성화
    DEACTIVEFIRST,
    //새로운 객체를 생성하여 풀에 추가
    ADDQUEUE,
    //새로운 객체를 생성 하되 풀에 추가 하지 않음
    DONOTREUSE
}
```

3) ADDQUEUE



객체를 새로 Instantiate한 후 큐에 추가 (IsReusing컴포넌트가 있는 상태)

4) DONOTREUSE

Instantiate만 하고 큐에 추가하지 않음 ((IsReusing컴포넌트가 없는 상태)