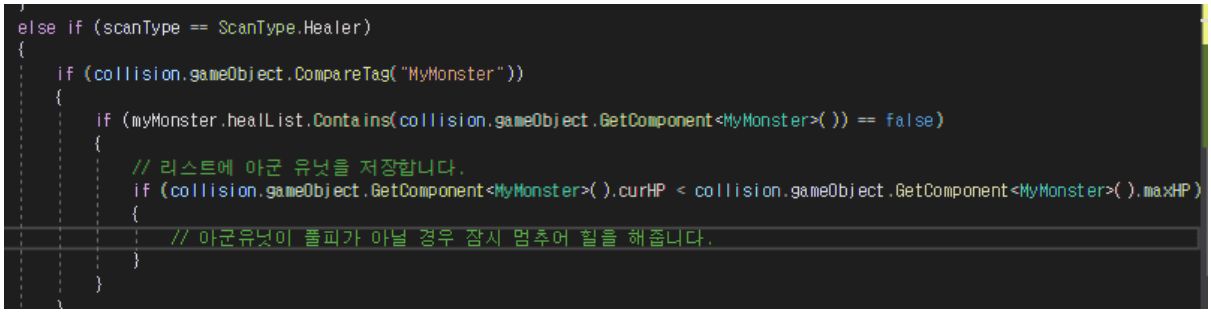
# GetComponent 최적화

- 최근에 디커플링이랑 코드 깔끔하게 정리하는 거 공부하다가 옛날에 즐겨 찾기 추가해 놨던 코드 리뷰 글이 괜찮아서 정리함. (https://gall.dcinside.com/mgallery/board/view?id=game_dev&no=98575)



```
else if (scanType == ScanType.Healer)
{
    if (collision.gameObject.CompareTag("MyMonster"))
    {
        if (myMonster.healList.Contains(collision.gameObject.GetComponent<MyMonster>()) == false)
        {
            // 리스트에 아군 유닛을 저장합니다.
            if (collision.gameObject.GetComponent<MyMonster>().curHP < collision.gameObject.GetComponent<MyMonster>().maxHP)
            {
                // 아군유닛이 풀피가 아닐 경우 잠시 멈추어 힐을 해줍니다.
            }
        }
    }
```

아군 힐러 유닛을 만들려고 함. 범위내에 아군의 체력을 확인해서 최대 체력이 아닐 때 힐을 하려고 함. 전투가 실시간으로 이루어지니 OnTriggerStay로 계속 체크.

질문자의 궁금점 - GetComponent를 너무 많이 쓰면 최적화에 좋지 않다는 글을 봄.

1. GetComponent 많이 사용하여도 상관없는건가?

2. 실시간으로 체력을 관리할 수 있는 다른 방법이 있나?

- 답변자 내용

**1. GetComponent 많이써도 상관없음?**

덜 쓸 수록 좋긴하겠다만 필요할 때 잘 쓰라고 있는거니 부담은 안 가져도 됨. 일단 만들고 / 문제되면 최적화를 고민.

막연하게 GetComponent 많이해서 아.. 최적화걱정되는데~ 로 생각하지말고

목표 프레임 안나오는데(혹은 주기적인 체크로) 프로파일링 돌림

- > update에서 잦은 GetComponent로 인해 비용이 많이 발생하는 걸 확인
- > 개선

의 과정으로 생각하는게 좋을거 같음. 너무 이른 최적화는 독임.

**2. 실시간 체력관리 다른방법?**

- > 방법이야 많은데 게임 전체 구조나 요구사항등을 다 알 수 없으니

섣불리 이렇게 저렇게 구조를 바꿔라 얘기하긴 좀 글고..

이에대한 답변은 질문글에 다른사람들이 달아논 댓글이나 다른 겜 코드들 참고해보면 될 듯

- ----------

**3. 아님말고 코드훈수**

글내용으로 미뤄봐서 OnTriggerStay에서 이걸 돌리고 있는것으로 보임

대충 비슷하게 만들었음.

myMonster는 어디서 뭐하던앤지 알수없어서 대충 아무렇게나 뺐음

```
public MyMonster myMonster;  Unchanged
Event function
private void OnTriggerStay(Collider collision)
{
    if (collision.gameObject.CompareTag("MyMonster"))
    {
        if (myMonster.healList.Contains( item: collision.gameObject.GetComponent<MyMonster>()) == false)
        {
            // 리스트에 아군 유닛을 저장합니다.
            if (collision.gameObject.GetComponent<MyMonster>().curHP <
                collision.gameObject.GetComponent<MyMonster>().maxHP)
            {
                // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
            }
        }
    }
}
```

1. 라이더 말 듣기

invert if 해주라고 하시니 고쳐주자

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.CompareTag("MyMonster")) return;

    if (myMonster.healList.Contains( item: collision.gameObject.GetComponent<MyMonster>()) != false) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (collision.gameObject.GetComponent<MyMonster>().curHP <
        collision.gameObject.GetComponent<MyMonster>().maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

2. 필요없는거 지워주고

!= false : == true 어차피 똑같음

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.CompareTag("MyMonster")) return;

    if (myMonster.healList.Contains( item: collision.gameObject.GetComponent<MyMonster>()) != false) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (collision.gameObject.GetComponent<MyMonster>().curHP <
        collision.gameObject.GetComponent<MyMonster>().maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

3. 중복되는 거 합쳐주자

collision.gameObject.GetComponent<MyMonster>를 왜 매번 계속 새로 하는거야

걍 보기좋게 줄여

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.CompareTag("MyMonster")) return;

    var colMonster = collision.gameObject.GetComponent<MyMonster>();

    if (myMonster.healList.Contains(colMonster)) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (colMonster.curHP < colMonster.maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

**4. GetComponent대신에 TryGetComponent를 사용**

GetComponent는 할당에 성공 실패여부와 상관없이 GC Allocation이 발생함

(유니티 버젼 올라가면서 GetComponent 개선후 에디터에선 할당 / 런타임엔 비할당 하루정도 생각해봤는데.. 둘다 GC 문제도 없고 런타임에 GetComponent가 더빠르니 TryGet을 꼭 고집할필요는 없어보임.
솔직히 둘이 차이 난다고해봐야 그게 퍼포먼스적으로 얼마나 영향줄만큼 큰 문제겠어 그냥 경우에따라 이런땐 TryGet하고 / 컴포넌트 null체크 안해도되는 확실한상황에선 걍 GetComponent써도 아무 문제없다고생각

- https://docs.unity3d.com/Manual/performance-garbage-collector.html
- https://tistory.jeon.sh/11
- https://forum.unity.com/threads/does-getcomponent-t-no-longer-allocates-garbage.978090/#post-6354051

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.CompareTag("MyMonster")) return;

    if (!collision.gameObject.TryGetComponent<MyMonster>(out var colMonster)) return;

    if (myMonster.healList.Contains(colMonster)) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (colMonster.curHP < colMonster.maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

**5. CompareTag 체크 이후의 GetComponent의 논리적 불필요성**

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.CompareTag("MyMonster")) return;

    if (!collision.gameObject.TryGetComponent<MyMonster>(out var colMonster)) return;

    if (myMonster.healList.Contains(colMonster)) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (colMonster.curHP < colMonster.maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

논리적으로 생각해보셈 CompareTag 목적은 해당 콜라이더의 GameObject가

MyMonster라는 컴포넌트를 포함하고 있는 "MyMonster"라는 태그를 달고있는 GameObject인가?

를 확인하기 위함임

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.TryGetComponent<MyMonster>(out var colMonster)) return;

    if (myMonster.healList.Contains(colMonster)) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (colMonster.curHP < colMonster.maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

태그 확인이 필요할까? 걍 TryGetComponent로 목적을 달성할 수 있다

q) MyMonster태그가 달려있지 않은 다른 무언가가 MyMonster Component를 달고있을수도 있잖아요!!

a) 그렇게 만들었을리 없겠지

6. component.TryGetComponent

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.gameObject.TryGetComponent<MyMonster>(out var colMonster)) return;

    if (myMonster.healList.Contains(colMonster)) return;

    // 리스트에 아군 유닛을 저장합니다.
    if (colMonster.curHP < colMonster.maxHP)
    {
        // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
    }
}
```

gameObject로 한단계 더 거치는거 없어도 될듯?

7. healList 추가 기능은 OnTriggerStay가 아니라 MyMonster 가 수행하도록

```
public class MyMonster : MonoBehaviour
{
    private List<MyMonster> healList = new List<MyMonster>();
    public int              curHP;  Serializable
    public int              maxHP;  Serializable

    1 usage
    public void Add2HealList(MyMonster m)
    {
        if (healList.Contains(m)) return;
        if (m.curHP < m.maxHP) return;

        // Add 2 HealList
        healList.Add(m);
    }
}
```

하면서 healList는 private으로 변경

외부에서 내부의 healList를 사용하는 경우 기능단위를 호출하도록

public void Heal()

{

foreach( m in healList)

// heal

}

함수를 클래스안에 만들어주면 될듯

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.TryGetComponent<MyMonster>(out var colMonster)) return;
    myMonster.Add2HealList(colMonster);
}
```

짠 이렇게 보기좋게 만들 수 있다.

list에 집어넣는 입장에서 어떠한 조건이나 추가처리를 다른곳에서 신경쓰지 않아도됨

- ----------

**Before**

```
public MyMonster myMonster;  Unchanged
Event function
private void OnTriggerStay(Collider collision)
{
    if (collision.gameObject.CompareTag("MyMonster"))
    {
        if (myMonster.healList.Contains( item: collision.gameObject.GetComponent<MyMonster>()) == false)
        {
            // 리스트에 아군 유닛을 저장합니다.
            if (collision.gameObject.GetComponent<MyMonster>().curHP <
                collision.gameObject.GetComponent<MyMonster>().maxHP)
            {
                // 아군유닛이 돌파가 아닐 경우 잠시 멈추어 힐을 해줍니다.
            }
        }
    }
}
```

**After**

```
private void OnTriggerStay(Collider collision)
{
    if (!collision.TryGetComponent<MyMonster>(out var colMonster)) return;
    myMonster.Add2HealList(colMonster);
}
```

더 잘 읽히고 간단해졌지?

한눈에 봐도 컴포넌트 있는지 확인해서 있으면 힐리스트에 추가해라~ 하고 목적도 확인하기 쉬워짐

q) 그래서 이거하면 겜 빨라짐?

뭐 꼴랑이거로 유의미한 차이가 있겠음?

궁금하면 프로파일링해보셈

것보다 주석없으면 1년뒤에 주르르륵 달린 긴 코드 보면서 스트레스 받을꺼

한눈에 잘읽히고 간단하고 쉽게 명확하게 줄엿다는데 의의를 가질래