

TextRPGTemplate 사용 메뉴얼

1. Scene 추가

1. Scene 추가

1. 해당 Scene을 위한 ID 생성

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using TextRPG.Context;
7 using TextRPG.View;
8
9 namespace TextRPG.Scene
10 {
11     참조 16개
12     public static class SceneID
13     {
14         public const string Main = "Main";
15         public const string Status = "Status";
16         public const string Inventory = "Inventory";
17         public const string Wear = "Wear";
18         public const string Shop = "Shop";
19         public const string Nothing = "None";
20         public const string Buy = "Buy";
21         public const string Rest = "Rest";
22         public const string Sell = "Sell";
23         public const string DungeonSelect = "DungeonSelect";
24         public const string DungeonClear = "DungeonClear";
25         public const string DungeonFail = "DungeonFail";
26     }
27     참조 34개
28     internal abstract class AScene
29     {
30         참조 81개
31         protected GameContext gameContext { get; set; }
32         참조 44개
```

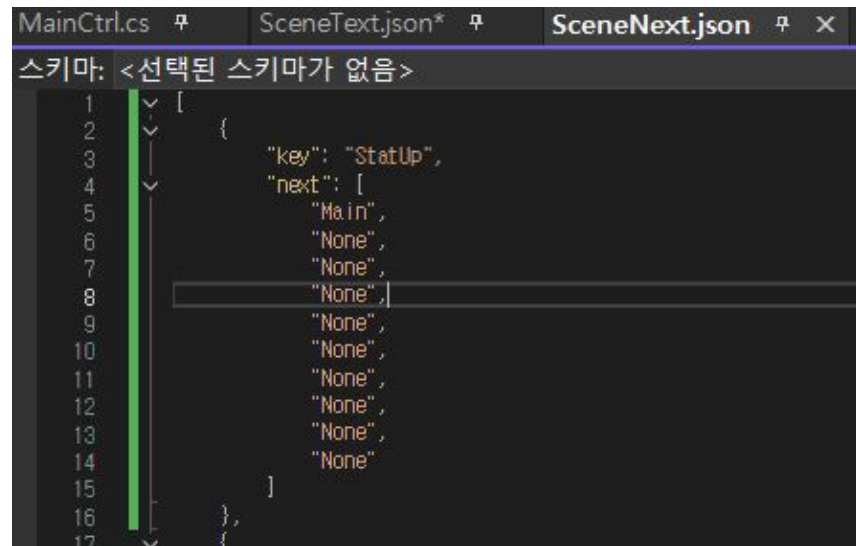
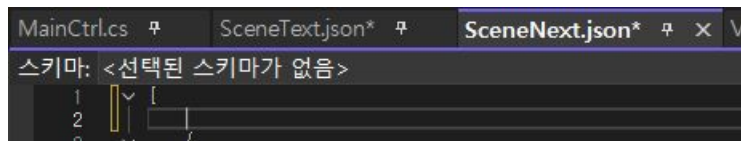


```
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using TextRPG.Context;
7 using TextRPG.View;
8
9 namespace TextRPG.Scene
10 {
11     참조 16개
12     public static class SceneID
13     {
14         public const string Main = "Main";
15         public const string Status = "Status";
16         public const string Inventory = "Inventory";
17         public const string Wear = "Wear";
18         public const string Shop = "Shop";
19         public const string Nothing = "None";
20         public const string Buy = "Buy";
21         public const string Rest = "Rest";
22         public const string Sell = "Sell";
23         public const string DungeonSelect = "DungeonSelect";
24         public const string DungeonClear = "DungeonClear";
25         public const string DungeonFail = "DungeonFail";
26         public const string StatUp = "StatUp";
27     }
28     참조 34개
29     internal abstract class AScene
30     {
31         참조 81개
32         protected GameContext gameContext { get; set; }
33         참조 44개
34         protected Dictionary<string, AView> viewMap { get; set; }
```

Ascene.cs

1. Scene 추가

2. SceneNext에 해당 Scene 항목 추가(Scene 간 연결)



SceneNext.json

1. Scene 추가

3. SceneNext에서 다른 Scene과 해당 Scene 연결

```
77  {
78  |  |  "key": "Main",
79  |  |  "next": [
80  |  |  |  "None",
81  |  |  |  "Status",
82  |  |  |  "Inventory",
83  |  |  |  "Shop",
84  |  |  |  "DungeonSelect",
85  |  |  |  "Rest",
86  |  |  |  "None",
87  |  |  |  "None",
88  |  |  |  "None",
89  |  |  |  "None"
90  |  |  ]
91  |  },
92  |  {
93  |  |  "key": "Rest",
```

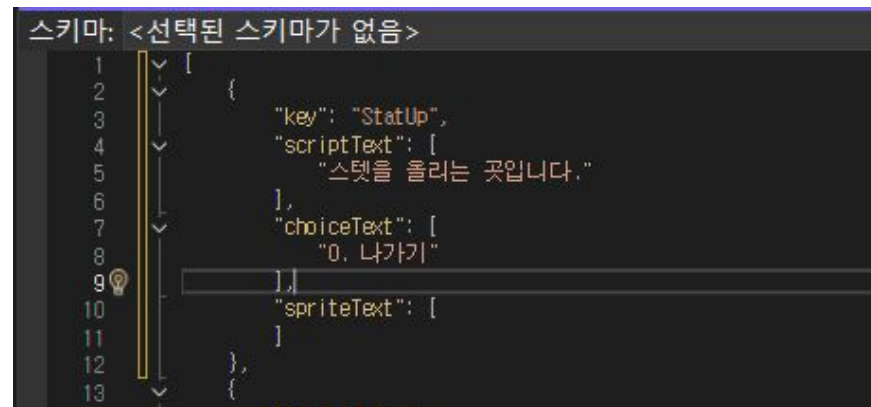
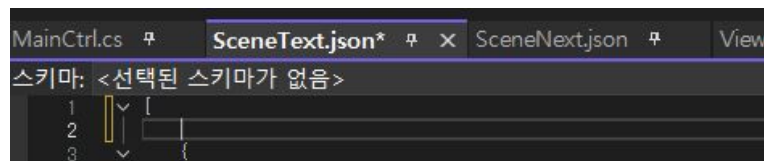


```
77  {
78  |  |  "key": "Main",
79  |  |  "next": [
80  |  |  |  "None",
81  |  |  |  "Status",
82  |  |  |  "Inventory",
83  |  |  |  "Shop",
84  |  |  |  "DungeonSelect",
85  |  |  |  "Rest",
86  |  |  |  "StatUp",
87  |  |  |  "None",
88  |  |  |  "None",
89  |  |  |  "None"
90  |  |  ]
91  |  },
92  |  {
93  |  |  "key": "Rest",
```

SceneNext.json

1. Scene 추가

4. SceneText에 해당 Scene 관련 항목 추가(Scene 내 정적 텍스트)



SceneText.json

1. Scene 추가

5. SceneText에 다른 Scene에도 해당 Scene 연결 관련 설명 텍스트 추가
(이 부분은 선택 사항)

```
13 {
14   "key": "Main",
15   "scriptText": [
16     "스파르타 마을에 오신 여러분 환영합니다.",
17     "이곳에서 던전으로 들어가기전 활동을 할 수 있습니다."
18   ],
19   "choiceText": [
20     "1. 상태 보기",
21     "2. 인벤토리",
22     "3. 상점",
23     "4. 던전입장",
24     "5. 휴식하기"
25   ],
26   "spriteText": [
```



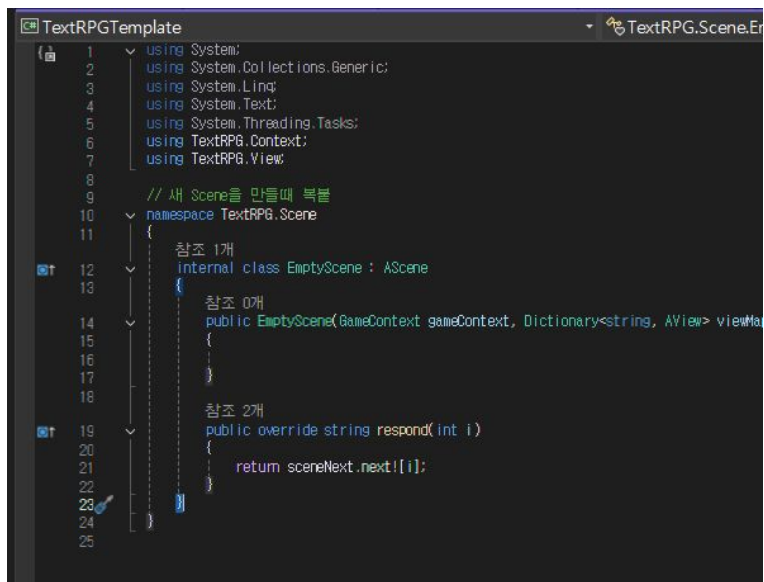
```
13 {
14   "key": "Main",
15   "scriptText": [
16     "스파르타 마을에 오신 여러분 환영합니다.",
17     "이곳에서 던전으로 들어가기전 활동을 할 수 있습니다."
18   ],
19   "choiceText": [
20     "1. 상태 보기",
21     "2. 인벤토리",
22     "3. 상점",
23     "4. 던전입장",
24     "5. 휴식하기",
25     "6. 스텟 올리기"
26   ],
27   "spriteText": [
```

SceneText.json

1. Scene 추가

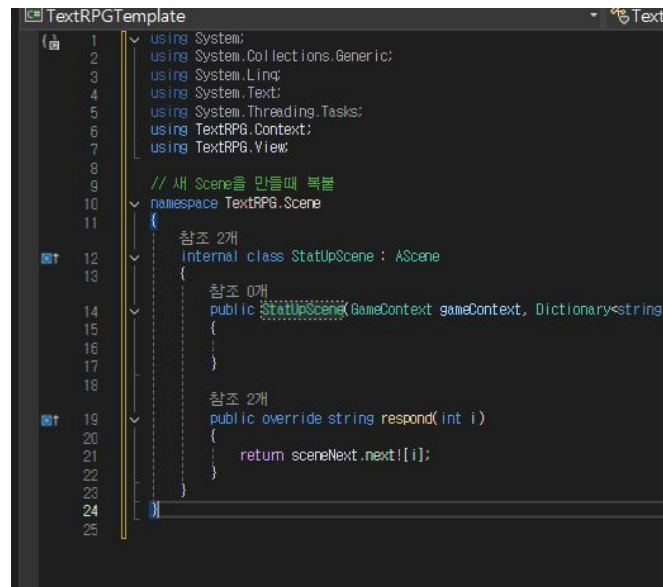
6. 해당 Scene 클래스 추가

EmptyScene.cs 내용을 복사해서 클래스와 생성자 이름만 바꾸면 됨



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using TextRPG.Context;
7 using TextRPG.View;
8
9 // 새 Scene을 만들때 복사
10 namespace TextRPG.Scene
11 {
12     참조 1개
13     internal class EmptyScene : AScene
14     {
15         참조 0개
16         public EmptyScene(GameContext gameContext, Dictionary<string, AView> viewMap)
17         {
18         }
19
20         참조 2개
21         public override string respond(int i)
22         {
23             return sceneNext.next![i];
24         }
25     }
26 }
```

EmptyScene.json



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using TextRPG.Context;
7 using TextRPG.View;
8
9 // 새 Scene을 만들때 복사
10 namespace TextRPG.Scene
11 {
12     참조 2개
13     internal class StatUpScene : AScene
14     {
15         참조 0개
16         public StatUpScene(GameContext gameContext, Dictionary<string, AView> viewMap)
17         {
18         }
19
20         참조 2개
21         public override string respond(int i)
22         {
23             return sceneNext.next![i];
24         }
25     }
26 }
```

StatupScene.json(예시)

1. Scene 추가

7. 해당 Scene을 게임에 연결

```
225 // 새 Scene을 만들면 이 부분에 추가  
226 참조 1개  
227 static void InitSceneFactoryMap(Dictionary<string, SceneMaker> sceneFactoryMap)  
228 {  
229     RegisterScene<MainScene>(sceneFactoryMap, SceneID.Main);  
230     RegisterScene<WearScene>(sceneFactoryMap, SceneID.Wear);  
231     RegisterScene<InventoryScene>(sceneFactoryMap, SceneID.Inventory);  
232     RegisterScene<StatusScene>(sceneFactoryMap, SceneID.Status);  
233     RegisterScene<ShopScene>(sceneFactoryMap, SceneID.Shop);  
234     RegisterScene<BuyScene>(sceneFactoryMap, SceneID.Buy);  
235     RegisterScene<SellScene>(sceneFactoryMap, SceneID.Sell);  
236     RegisterScene<RestScene>(sceneFactoryMap, SceneID.Rest);  
237     RegisterScene<DungeonSelectScene>(sceneFactoryMap, SceneID.DungeonSelect);  
238     RegisterScene<DungeonClearScene>(sceneFactoryMap, SceneID.DungeonClear);  
239     RegisterScene<DungeonFailScene>(sceneFactoryMap, SceneID.DungeonFail);  
240 }  
241  
242  
243  
244
```



```
226 static void InitSceneFactoryMap(Dictionary<string, SceneMaker> sceneFactoryMap)  
227 {  
228     RegisterScene<MainScene>(sceneFactoryMap, SceneID.Main);  
229     RegisterScene<WearScene>(sceneFactoryMap, SceneID.Wear);  
230     RegisterScene<InventoryScene>(sceneFactoryMap, SceneID.Inventory);  
231     RegisterScene<StatusScene>(sceneFactoryMap, SceneID.Status);  
232     RegisterScene<ShopScene>(sceneFactoryMap, SceneID.Shop);  
233     RegisterScene<BuyScene>(sceneFactoryMap, SceneID.Buy);  
234     RegisterScene<SellScene>(sceneFactoryMap, SceneID.Sell);  
235     RegisterScene<RestScene>(sceneFactoryMap, SceneID.Rest);  
236     RegisterScene<DungeonSelectScene>(sceneFactoryMap, SceneID.DungeonSelect);  
237     RegisterScene<DungeonClearScene>(sceneFactoryMap, SceneID.DungeonClear);  
238     RegisterScene<DungeonFailScene>(sceneFactoryMap, SceneID.DungeonFail);  
239     RegisterScene<StatusScene>(sceneFactoryMap, SceneID.StattUp);  
240 }  
241  
242  
243  
244
```

2. Scene 내용 수정

2. Scene 내용 수정

1. 해당 Scene 클래스에 DrawScene 추가

EmptyScene에서 복사해 오기

```
19  참조 5개  
20  public override void DrawScene()  
21  {  
22      ClearScene();  
23      List<string> dynamicText = new();  
24      ((DynamicView)viewMap[ViewID.Dynamic]  
25      ((SpriteView)viewMap[ViewID.Sprite])  
26  
27      Render();  
28  
29  }
```



```
참조 5개  
public override void DrawScene()  
{  
    ClearScene();  
  
    List<string> dynamicText = new();  
    ((DynamicView)viewMap[ViewID.Dynamic]  
    ((SpriteView)viewMap[ViewID.Sprite]  
  
    Render();  
}
```

EmptyScene.cs

StatupScene.cs(예시)

2. Scene 내용 수정

2. DrawScene 내용 수정

GameContext에서 현재 게임 정보들 참조

```
참조 5개  
public override void DrawScene()  
{  
    ClearScene();  
  
    List<string> dynamicText = new();  
    ((DynamicView)viewMap[ViewID.Dynamic]).SetText(dynamicText.ToArray());  
    ((SpriteView)viewMap[ViewID.Sprite]).SetText(sceneText.spriteText!);  
  
    Render();  
}
```



```
참조 5개  
public override void DrawScene()  
{  
    ClearScene();  
  
    List<string> dynamicText = new();  
    dynamicText.Add("어떤 스텟을 올릴까요?");  
    dynamicText.Add($"1. 공격력 : {gameContext.ch.defaultAttack}");  
    dynamicText.Add($"2. 방어력 : {gameContext.ch.defaultGuard}");  
    ((DynamicView)viewMap[ViewID.Dynamic]).SetText(dynamicText.ToArray());  
    ((SpriteView)viewMap[ViewID.Sprite]).SetText(sceneText.spriteText!);  
  
    Render();  
}
```

StatupScene.cs(예시)

StatupScene.cs(예시)

2. Scene 에서 데이터 처리

2. Scene에서 데이터 처리

1. respond 내용 수정

매개변수는 키보드로 받은 입력 char

참조 2개

```
public override string respond(int i)
{
    return sceneNext.next![i];
}
```



참조 2개

```
public override string respond(int i)
{
    switch (i)
    {
        case 1: gameContext.ch.defaultAttack++; break;
        case 2: gameContext.ch.defaultGuard++; break;
    }
    return sceneNext.next![i];
}
```

StatupScene.cs(예시)

StatupScene.cs(예시)

데이터 처리 결과를 보여주고 싶으면 상태 창에서 확인하거나, 처리 결과를 보여주는 **Scene**을 새로 만들기, **prev** ~ 상태, **cur** ~ 상태 정보를 만들어서 처리 결과를 보여주는 **Scene**에서 처리 전후를 보여줄 수 있게 하기

3. 캐릭터 관련 새로운 데이터 타입 추가 (치명타? 등등)

3. 캐릭터 관련 새로운 데이터 타입 추가

1. Character 클래스에 관련 매개변수 추가

```
10 internal class Character
11 {
12     참조 4개
13     public string? name { get; set; }
14     참조 4개
15     public string? job { get; set; }
16     참조 6개
17     public float defaultAttack { get; set; }
18     참조 6개
19     public float defaultGuard { get; set; }
20     참조 9개
21     public int hp { get; set; }
22     참조 16개
23     public int gold { get; set; }
24     참조 8개
25     public int clearCount { get; set; }
26     참조 34개
27     public Inventory inventory
28     { get; set; }
29     참조 12개
```

Character.cs

```
internal class Character
{
    참조 4개
    public string? name { get; set; }
    참조 4개
    public string? job { get; set; }
    참조 0개
    public float critical { get; set; }
    참조 6개
    public float defaultAttack { get; set; } // 현재 6개
    참조 6개
    public float defaultGuard { get; set; } // 현재 6개
    참조 9개
    public int hp { get; set; }
    참조 16개
    public int gold { get; set; }
    참조 8개
    public int clearCount { get; set; }
```

Character.cs

3. 캐릭터 관련 새로운 데이터 타입 추가

2. SaveData 클래스에 관련 매개변수 추가

```
13 public string? job { get; set; }  
14 참조 2개  
15 public float attack { get; set; }  
16 참조 2개  
17 public float guard { get; set; }  
18 참조 2개  
19 public int hp { get; set; }  
20 참조 2개  
21 public int gold { get; set; }  
22 참조 2개  
23 public int clearCount { get; set; }  
24 참조 2개  
25 public Item[] items { get; set; } = System.Array.Empty<Item>();  
26 참조 2개  
27 public Item[] shopItems { get; set; } = System.Array.Empty<Item>();  
28 참조 0개
```

SaveData.cs

```
13 public string? job { get; set; }  
14 참조 2개  
15 public float attack { get; set; }  
16 참조 2개  
17 public float guard { get; set; }  
18 참조 2개  
19 public int hp { get; set; }  
20 참조 2개  
21 public int gold { get; set; }  
22 참조 0개  
23 public float critical { get; set; }  
24 참조 2개  
25 public int clearCount { get; set; }  
26 참조 2개  
27 public Item[] items { get; set; } = System.Array.Empty<Item>();  
28 참조 2개  
29 public Item[] shopItems { get; set; } = System.Array.Empty<Item>();  
30 참조 0개  
31 public SaveData() { }  
32 참조 1개  
33 public SaveData(SaveContext saveContext)
```

SaveData.cs

3. 캐릭터 관련 새로운 데이터 타입 추가

3. SaveData 생성자 에 관련 코드 추가

```
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
public SaveData() {  
    참조 1개  
    public SaveData(GameContext gameContext)  
    {  
        name = gameContext.ch.name;  
        job = gameContext.ch.job;  
        attack = gameContext.ch.defaultAttack;  
        guard = gameContext.ch.defaultGuard;  
        hp = gameContext.ch.hp;  
        gold = gameContext.ch.gold;  
        clearCount = gameContext.ch.clearCount;  
        items = gameContext.ch.inventory.items!.ToArray();  
        shopItems = gameContext.shop.items!.ToArray();  
    }  
}
```

SaveData.cs

```
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
public SaveData(GameContext gameContext)  
{  
    name = gameContext.ch.name;  
    job = gameContext.ch.job;  
    attack = gameContext.ch.defaultAttack;  
    guard = gameContext.ch.defaultGuard;  
    hp = gameContext.ch.hp;  
    gold = gameContext.ch.gold;  
    critical = gameContext.ch.critical;  
    clearCount = gameContext.ch.clearCount;  
    items = gameContext.ch.inventory.items!.ToArray();  
    shopItems = gameContext.shop.items!.ToArray();  
}
```

SaveData.cs

3. 캐릭터 관련 새로운 데이터 타입 추가

4. Character 생성자 에 관련 코드 추가

```
23 public Character(SaveData saveData)
24 {
25     this.name = saveData.name;
26     this.job = saveData.job;
27     this.defaultAttack = saveData.attack;
28     this.defaultGuard = saveData.guard;
29     this.hp = saveData.hp;
30     this.gold = saveData.gold;
31     this.clearCount = saveData.clearCount;
32     this.inventory = new Inventory(new List<Item>(saveData.items));
33     this.critical = saveData.critical;
34 }
35
36 참조 0개
37 public Character(string name, string job, float attack, float guard, int hp, int gold, int clearCount, Inventory inventory, float critical)
38 {
39     this.name = name;
40     this.job = job;
41     this.defaultAttack = attack;
42     this.defaultGuard = guard;
43     this.hp = hp;
44     this.gold = gold;
45     this.clearCount = clearCount;
46     this.inventory = inventory;
47     this.critical = critical;
48 }
```

3. 캐릭터 관련 새로운 데이터 타입 추가

4. defaultData.json 에 관련 항목 추가(새로운 스탯이 캐릭터에 추가 됐으니 그에 맞는 새로운 초기 data)

```
1 {
2   "name": "Chad",
3   "job": "전사",
4   "attack": 10,
5   "guard": 5,
6   "hp": 100,
7   "gold": 1500,
8   "clearCount": 0,
9   "items": [],
10  "shopItems": [
11    {
12      "key": "noviceArmor",
13      "equiped": false,
```



```
스키마: <선택된 스키마가 없음>
1 {
2   "name": "Chad",
3   "job": "전사",
4   "attack": 10,
5   "guard": 5,
6   "hp": 100,
7   "gold": 1500,
8   "clearCount": 0,
9   "items": [],
10  "critical": 15,
11  "shopItems": [
12    {
13      "key": "noviceArmor",
14      "equiped": false,
15      "name": "수련자 갑옷",
```

defaultData.cs

3. 캐릭터 관련 새로운 데이터 타입 추가

5. (잘 됐는지 확인하기, 다른 scene에 관련 코드 추가)

```
참조 5개
17 public override void DrawScene()
18 {
19     ClearScene();
20
21     List<string> dynamicText = new();
22
23     Character ch = gameContext.ch;
24
25     dynamicText.Add($"Lv.{ch.getLevel()}");
26     dynamicText.Add($"{{ch.name}} ({{ch.job}})");
27     dynamicText.Add($"공격력 : {{ch.getTotalAttack}}");
28     dynamicText.Add($"방어력 : {{ch.getTotalGuard}}");
29     dynamicText.Add($"체력 : {{ch.hp}}");
30     dynamicText.Add($"Gold : {{ch.gold}}G");
31
32     ((DynamicView)viewMap[ViewID.Dynamic]).SetText(d
33     ((SpriteView)viewMap[ViewID.Sprite]).SetText(
34
35     Render();
36 }
37
참조 2개
```

StatusScene.cs

```
참조 5개
17 public override void DrawScene()
18 {
19     ClearScene();
20
21     List<string> dynamicText = new();
22
23     Character ch = gameContext.ch;
24
25     dynamicText.Add($"Lv.{ch.getLevel()}");
26     dynamicText.Add($"{{ch.name}} ({{ch.job}})");
27     dynamicText.Add($"공격력 : {{ch.getTotalAttack}}");
28     dynamicText.Add($"방어력 : {{ch.getTotalGuard}}");
29     dynamicText.Add($"체력 : {{ch.hp}}");
30     dynamicText.Add($"Gold : {{ch.gold}}G");
31     dynamicText.Add($"Critical : {{ch.critical}}");
32
33     ((DynamicView)viewMap[ViewID.Dynamic]).SetText(d
34     ((SpriteView)viewMap[ViewID.Sprite]).SetText(sce
35
36     Render();
37 }
38
참조 2개
19 public override string respond(int i)
```

StatusScene.cs

4. 새로운 Game내 클래스 추가(Monster?)

4. 새로운 Game내 클래스 추가(Monster?)

이 부분은 살짝 사진이 많이 필요해서 영상으로 하는 게 나을 듯?

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace TextRPG.Context
8 {
9     [Serializable]
10     참조 12개
11     internal class DungeonData
12     {
13         참조 5개
14         public string title { get; set; } = "";
15         참조 5개
16         public float recommendArmor { get; set; }
17         참조 20개
18         public int reward { get; set; }
19         참조 0개
20         public DungeonData(string title, float recommendArmor, int reward)
21         {
22             this.title = title;
23             this.recommendArmor = recommendArmor;
24             this.reward = reward;
25         }
26     }
27 }
```

DungeonData.cs

```
스키마: <선택된 스키마가 없음>
1
2
3
4 {
5     "title": "쉬운 던전",
6     "recommendArmor": 5,
7     "reward": 1000
8 },
9
10 {
11     "title": "일반 던전",
12     "recommendArmor": 11,
13     "reward": 1700
14 },
15
16 {
17     "title": "어려운 던전",
18     "recommendArmor": 17,
19     "reward": 2500
20 }
21 }
```

DungeonData.json

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace TextRPG.Context
8 {
9     [Serializable]
10     참조 21개
11     internal class GameContext
12     {
13         참조 49개
14         public Character ch { get; set; }
15         참조 14개
16         public Shop shop { get; set; }
17         참조 6개
18         public List<DungeonData> dungeonList { get; set; } = new List<DungeonData>();
19         참조 4개
20         public DungeonData? enteredDungeon { get; set; } = null;
21         참조 6개
22         public int prevHp { get; set; }
23         참조 10개
24         public int curHp { get; set; }
25         참조 4개
26         public int prevGold { get; set; }
27         참조 4개
28         public int curGold { get; set; }
29         참조 1개
30         public GameContext(SaveData saveData, List<DungeonData> dungeonData)
31         {
32             ch = new(saveData);
33             shop = new(new List<Item>(saveData.shopItems));
34             this.dungeonList = new List<DungeonData>(dungeonData);
35         }
36     }
37 }
```

GameContext.cs

4. 새로운 Game내 클래스 추가(Monster?)

```
81  
82 //Save 외의 동적 데이터 불러오기  
83 var dungeonDataJson = File.ReadAllText(JsonPath.dungeonData.JsonPath);  
84 var dungeonData = JsonSerializer.Deserialize<List<DungeonData>>(dungeonDataJson);  
85  
86
```

MainCtrl.cs

여기까지 수정했으면 이제부터 Scene들에서 해당 새로운 클래스들을 불러올 수 있습니다

```
public override void DrawScene()  
{  
    ClearScene();  
    List<string> dynamicText = new();  
    List<DungeonData> dungeonList = gameContext.dungeonList;  
    for (int i = 0; i < dungeonList.Count; i++) {  
        dynamicText.Add($"{i + 1}. {dungeonList[i].title} #방어력 {dunge  
    }  
    ((DynamicView)viewMap[ViewID.Dynamic]).SetText(dynamicText.ToArray());  
    ((SpriteView)viewMap[ViewID.Sprite]).SetText(sceneText.spriteText!);  
}
```

DungeonSelectScene.cs