BAB VII

IMPLEMENTASI GAME

Dengan level selesai diimplementasikan, sekarang kita bisa mempersiapkan komponen komponen dalam level yang membuat Splatted layak dipanggil sebuah game. Sama seperti bab 6, bab ini akan menjelaskan seluruh kode yang dipakai untuk komponen game dan penjelasan mengapa kode tersebut seperti berikut. Di bab ini hanya dijelaskan komponen yang digunakan dalam level dengan komponen – kompenen lain hanya disinggung atau dijelaskan sekilas agar bab ini tidak terlalu panjang.

1. Bola Salju

Dalam subbab ini akan dijelaskan aspek – aspek yang berhubungan dengan bola salju, dari cara memunculkan salju di arena hingga cara kerja bola spesial, dimulai dengan kompenen yang mengatur semua hal yang behubungan dengan bola salju : Snow Ball Manager.

Segmen Program 7.X Class Snow Ball Manager

1. public class SnowBallManager : MonoBehaviour{
2. public static SnowBallManager Instance;
3. public GameObject snowballscontainer;
4. [SerializeField] GameObject snowball;
5. [SerializeField] float respawnTime;
6. [SerializeField] int respawnAmount;
7. [SerializeField] ColorManager colManager;
8. float currentrespawnTimer;
9. private void Awake(){
10. if (Instance == null)
11. Instance = this;
12. }
13. // Start is called before the first frame update
14. void Start(){
15. currentrespawnTimer = respawnTime;
16. }
17. public void destroyball(int index){
18. GameObject ball = snowballscontainer.transform.GetChild(index).gameObject;

Segmen Program 7.X (Lanjutan)

1. Coordinate ballcoor = AStarAlgorithm.vectorToCoordinate(ball.transform.position);
2. if(SetObjects.getMap(true) != null)
3. SetObjects.setMap(ballcoor.yCoor, ballcoor.xCoor, 0);
4. Destroy(ball);
5. }
6. public void Update(){
7. if (currentrespawnTimer <= 0){
8. currentrespawnTimer = respawnTime;
9. putballs();
10. }
11. currentrespawnTimer -= Time.deltaTime;
12. }
13. void putballs(){
14. int x, y;
15. GameObject ballz;
16. for (int i = 0; i < respawnAmount; i++){
17. x = Mathf.RoundToInt(UnityEngine.Random.Range(0, SetObjects.getWidth() - 2));
18. y = Mathf.RoundToInt(UnityEngine.Random.Range(0, SetObjects.getHeight() - 2));
19. if (SetObjects.getMap(false)[y, x] == 0){
20. ballz = Instantiate(snowball, new Vector3(x + 1.5f, -y - 0.5f), Quaternion.identity);
21. ballz.transform.SetParent(snowballscontainer.transform ,true);
22. //Debug.Log("Bola ke-" + i + " = " + x + " " + y);
23. SetObjects.setMap(y, x, 4);
24. }
25. }
26. }
27. public void addBallinVector(Vector2 v){
28. GameObject ballz;
29. ballz = Instantiate(snowball,snowballscontainer.transform );
30. ballz.transform.position = v;
31. }
32. public bool deleteclosestball(Transform objecttransform, float rangetreshold){
33. bool isdeleted = false;
34. int index = getNearestBallIndex(objecttransform, rangetreshold);
35. if (index >= 0){
36. destroyball(index);
37. isdeleted = true;
38. }
39. return isdeleted;
40. }

Segmen Program 7.X (Lanjutan)

1. public GameObject getClosestBall(Transform objecttransform, float rangetreshold){
2. int index = getNearestBallIndex(objecttransform, rangetreshold);
3. return snowballscontainer.transform.GetChild(index).gameObject;
4. }
5. public int getNearestBallIndex(Transform objectTracked){
6. float closestrange = 999, range;
7. int i = 0, index = -1;
8. foreach (Transform ballz in snowballscontainer.transform){
9. range = Vector2.Distance(ballz.position, objectTracked.position);
10. if (range < closestrange && (ballz.GetComponent<PowerUp>() == null || ballz.GetComponent<PowerUp>().isActive())){
11. closestrange = range;
12. index = i;
13. }
14. i++;
15. }
16. return index;
17. }
18. public int getNearestBallIndex(Transform objectTracked, float range){
19. float closestrange = 999, currrange;
20. int i = 0, index = -1;
21. foreach (Transform ballz in snowballscontainer.transform){
22. currrange = Vector2.Distance(ballz.position, objectTracked.position);
23. if (currrange < range && currrange < closestrange && (ballz.GetComponent<PowerUp>() == null || ballz.GetComponent<PowerUp>().isActive())){
24. closestrange = currrange;
25. index = i;
26. }
27. i++;
28. }
29. if (index > -1 && Vector2.Distance(snowballscontainer.transform.GetChild(index).gameObject.transform.position, objectTracked.position) < range)
30. return index;
31. else
32. return -1;
33. }
34. public int getIndexfromSnowball(GameObject go){
35. int i = 0;
36. foreach (Transform item in snowballscontainer.transform){
37. if (item.gameObject == go)
38. return i;
39. i++;
40. }

Segmen Program 7.X (Lanjutan)

1. return -1 ;
2. }
3. public GameObject getBallfromIndex(int index){
4. try{
5. if (snowballscontainer.transform.childCount > 0)
6. return snowballscontainer.transform.GetChild(index).gameObject;
7. return null;
8. }
9. catch (System.Exception){
10. return null;
11. }
12. }
13. public bool isAnyBallNear(Vector2 position){
14. foreach (Transform item in snowballscontainer.transform){
15. if (Vector2.Distance(position, item.position) < 1 && (item.GetComponent<PowerUp>() == null || item.GetComponent<PowerUp>().isActive()))
16. return true;
17. }
18. return false;
19. }
20. public int getBallAmount(){
21. return snowballscontainer.transform.childCount;
22. }
23. }

Penjelasan untuk masing masing segmen program yang panjang diatas :

* 2 : Instance ini adalah sebuah static SnowballManager yang memperbolehkan setiap layar memiliki hanya 1 SnowballManager, dan memudahkan akses dari komponen game lain untuk mengakses snowballManager.
* 3 : GameObject yang akan menampung semua bola salju.
* 5 : Referensi GameObject gumpalan salju yang akan dispawn
* 6 : Periode sebuah gumpalan salju dipanggil.
* 7 : Menentukan berapa banyak gumpalan salju yang dispawn saat waktunya nge-spawn
* 8 : Referensi ke komponen Color Manager.
* 9 : Variabel untuk mencatat timer sebelum bola salju di-spawn
* 11 – 14 : Saat Gameobject nyala, maka akan dicek apakan Instance SnowBallManager kosong, bila kosong maka Instance akan diisi dengan SnowBallManager sekarang.
* 17 – 19 : Di frame pertama, currentRespawnTimer akan diisi dengan respawnTime, menandakan bila timer telah dimulai.
* 21 – 27 : Fungsi ini meminta parameter index dan bongkahan salju yang ke – index akan dihapus. Fungsi ini dipakai untuk mengambil sebuah bola, dimana mengambil bola dari bongkahan salju, bongkahan tersebut akan hilang.
* 29 – 35 : Setiap frame fungsi update ini akan dipanggil, dan seluruh isi update akan dijalankan.
* 30 – 33 : Bila timer sudah habis, maka reset timer dan taruh gumapalan salju.
* 34 : Kurangi waktu di timer hingga mendekati 0.

1. Karakter Player
2. Karakter Musuh
3. Sistem Timer