



POST.SPACES

APPENDIX II: SYSTEMS DOCUMENTATION

1 Code Documentation

The Unity platform means that much of the design and programming work takes place across various objects, variables, and linkages, not to mention plugin support with other libraries and assets. Much of the programme structure is difficult to document due to the component hierarchies and connections between the various game objects and prefab instances. However, these are the bulk of the raw scripts that have been manually coded to support the VR environment. Note that due to time limitations they have not been fully cleaned, organised and refactored; this appendix exists for reference and as a proof of work.

Due to the several changes in direction over the course of the year, there have been several different systems and custom scripts that have since been scrapped or superseded; those scripts are not included here to avoid more confusion. All scripts here are active in the final release, but there might be some minor ones which I have left out.

Contents

1	Code Documentation	1
2	Network Scripts	4
2.1	Network Manager	4
2.2	Network Player Spawner	7
2.3	Network Avatar	8
2.4	Avatar Random Customiser	12

2.5	Grabbable Photon Ownership	13
3	Player and XR Scripts	14
3.1	Player Controller	14
3.2	Movement Provider	15
3.3	XR Controller Manager	20
3.4	Teleportation Anchored Area	24
3.5	Comment Frame Teleporter	25
3.6	Custom Teleportation Provider	26
4	Reddit General Scripts	28
4.1	Reddit Controller	28
4.2	Reddit Central Space	34
4.3	Reddit Central Sphere	40
4.4	Reddit Cube Spawner	41
4.5	Reddit Node Block	43
4.6	Reddit Node Sphere	44
4.7	Subreddit Star	46
4.8	Reddit Info Panel	47
4.9	Geometry Torus	48
5	Reddit Post Scripts	52
5.1	Reddit Post Controller	52
5.2	Reddit Comment Frame	65
5.3	Reddit Comment Child	71
5.4	Vote Controller	73

5.5	Refresh Comments	76
6	General Support Scripts	78
6.1	Scroll Surface	78
6.2	Frame Dynamic	80
6.3	Frame Dynamic Planes	83
6.4	Text Frame	84
6.5	Material Manager	87
6.6	Lib Color	89
6.7	ColorRandomiser	89
6.8	ColorChanger	90

2 Network Scripts

Network management has been built using the Photon suite of network tools, allowing both object and variable synchronization as well as an additional plugin that works together for voice communication management. Most operations are performed client-side and the relevant data regarding synced objects, players, etc are sent over the network and objects are moved or instantiated on the other client.

2.1 Network Manager

The Network Manager handles most of the general network syncing operations, such as joining/leaving the network and updating player stats (i.e. which subreddit they are currently in).

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Pun;
5 using Photon.Realtime;
6 using ExitGames.Client.Photon;
7 using UnityEngine.XR.Interaction.Toolkit;
8
9
10
11 public class NetworkManager : MonoBehaviourPunCallbacks
12 {
13     public GameObject rig;
14     public RedditController redditController;
15     private string localSubreddit { get; set; }
16     private ExitGames.Client.Photon.Hashtable playerProperties;
17     private Dictionary<Player, NetworkAvatar> playerAvatarDict = new
18         Dictionary<Player, NetworkAvatar>();
19     // Start is called before the first frame update
20     void Start()
```

```

20     {
21         rig = GameObject.Find("XR Rig");
22         playerProperties = PhotonNetwork.LocalPlayer.CustomProperties;
23         ConnectToServer();
24     }
25
26     void ConnectToServer()
27     {
28         PhotonNetwork.ConnectUsingSettings();
29         Debug.Log("Try Connect To Server...");
30     }
31
32     public override void OnConnectedToMaster()
33     {
34         Debug.Log("Connected To Server.");
35
36         base.OnConnectedToMaster();
37         RoomOptions roomOptions = new RoomOptions();
38         roomOptions.MaxPlayers = 16;
39         roomOptions.IsVisible = true;
40         roomOptions.isOpen = true;
41
42         PhotonNetwork.JoinOrCreateRoom("all", roomOptions, TypedLobby.Default);
43     }
44
45     public override void OnJoinedRoom()
46     {
47         Debug.Log("Joined a Room: " + PhotonNetwork.CurrentRoom);
48         base.OnJoinedRoom();
49     }
50
51     public override void OnPlayerEnteredRoom(Player newUser)
52     {
53         Debug.Log("A new player joined the room.");
54         base.OnPlayerEnteredRoom(newUser);
55     }
56
57     public override void OnDisconnected(DisconnectCause cause)
58     {
59         Debug.Log("PUN DISCONNECTED: " + cause);
60         Invoke("ConnectToServer", 20f);
61     }
62
63
64     public override void OnPlayerPropertiesUpdate (Player targetPlayer,
65         ExitGames.Client.Photon.Hashtable changedProps)
66     {
67         Debug.Log("player properties update: " + targetPlayer + " - " +
changedProps);

```

```

67
68     if(changedProps.ContainsKey(" subreddit"))
69     {
70         if(targetPlayer != PhotonNetwork.LocalPlayer)
71         {
72             if(changedProps[" subreddit"].ToString() == localSubreddit)
73             {
74                 playerAvatarDict[targetPlayer].EnableSelf();
75                 Debug.Log(targetPlayer + " has entered your subreddit.");
76             }
77             else
78             {
79                 playerAvatarDict[targetPlayer].DisableSelf();
80                 Debug.Log(targetPlayer + " has left your subreddit.");
81             }
82         }
83     }
84
85
86     public void SetPlayerSubreddit(string subreddit)
87     {
88         localSubreddit = subreddit;
89         rig.GetComponent<XRRig>().MatchRigUpRigForward(new Vector3(0, 3, 0),
90             new Vector3(1, 0, 0));
91         rig.transform.position = new Vector3(0, -3, 0);
92
93         //Set network player property
94         if (!playerProperties.ContainsKey(" subreddit"))
95         {
96             playerProperties.Add(" subreddit", subreddit);
97             Debug.Log("Property added");
98         }
99         playerProperties[" subreddit"] = subreddit;
100        PhotonNetwork.LocalPlayer.SetCustomProperties(playerProperties);
101
102        //Initialise posts
103        redditController.Initialise(subreddit);
104    }
105
106    public void AddPlayerAvatar(Player player, NetworkAvatar avatar)
107    {
108        if(!playerAvatarDict.ContainsKey(player))
109        {
110            playerAvatarDict.Add(player, avatar);
111            Debug.Log("Added " + playerAvatarDict[player] + " to list of
112                player avatars.");
113        }
114    }

```

```
114 }
```

2.2 Network Player Spawner

The Network Player Spawner is responsible for spawning in player avatars.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Pun;
5
6 public class NetworkPlayerSpawner : MonoBehaviourPunCallbacks
7 {
8     private GameObject spawnedPlayerPrefab;
9     private Color shirtColor;
10    private Color pantsColor;
11    private object[] avatarInitData = new object[6];
12
13    void Start()
14    {
15        //Randomise clothes
16        //shirt color
17        avatarInitData[0] = Random.Range(80, 230);
18        avatarInitData[1] = Random.Range(80, 230);
19        avatarInitData[2] = Random.Range(80, 230);
20
21        //pants color
22        avatarInitData[3] = Random.Range(20, 70);
23        avatarInitData[4] = Random.Range(20, 70);
24        avatarInitData[5] = Random.Range(20, 70);
25
26    }
27    public override void OnJoinedRoom()
28    {
29        base.OnJoinedRoom();
30        spawnedPlayerPrefab = PhotonNetwork.Instantiate("Network Avatar",
31            transform.position, transform.rotation, 0, avatarInitData);
32    }
33
34    public override void OnLeftRoom()
35    {
36        base.OnLeftRoom();
```

```
37     PhotonNetwork.Destroy(spawnedPlayerPrefab);
38 }
39 }
```

2.3 Network Avatar

The Network Avatar script is an important script that controls the positioning, animation and behaviour of each player avatar. These avatars are spawned client-side for each player in the server and are synced over the Photon network.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.XR;
5 using Photon.Pun;
6 using Photon.Realtime;
7 using Photon.Voice.Unity;
8
9 public class NetworkAvatar : MonoBehaviour, IPunInstantiateMagicCallback
10 {
11     public Transform head;
12     public List<Transform> meshes;
13     public Transform character;
14     public bool hideSelf = true;
15     public bool hideHead = true;
16     public float cameraFrontOffset;
17     public GameObject speaker;
18     private Vector3 baseVector;
19     private bool parented = false;
20     private GameObject rig;
21     private Animator animator;
22     private NetworkManager networkManager;
23     //private int interval = 1;
24     //private float animUpdateTime = 0;
25
26     private PhotonView photonView;
27
28     // Start is called before the first frame update
29     void Awake()
30     {
31         photonView = GetComponent<PhotonView>();
```

```

32     rig = GameObject.Find("XR_Rig");
33     networkManager = GameObject.Find("NetworkManager").GetComponent<
34         NetworkManager>();
35     animator = GetComponentInChildren<Animator>();
36     if(photonView.IsMine) MapCharacterPosition(character, XRNode.Head);
37 }
38
39 public void OnPhotonInstantiate(PhotonMessageInfo info)
40 {
41     object[] data = info.photonView.InstantiationData;
42     if (data != null)
43     {
44         SetColor(new Color((int)data[0]/255f, (int)data[1]/255f, (int)
45             data[2]/255f), new Color((int)data[3]/255f, (int)data[4]/255
46             f, (int)data[5]/255f));
47     }
48     networkManager.AddPlayerAvatar(photonView.Owner, this);
49 }
50
51 // Update is called once per frame
52 void Update()
53 {
54     if (photonView.IsMine)
55     {
56         if (hideSelf)
57         {
58             foreach (Transform mesh in meshes)
59             {
60                 mesh.gameObject.SetActive(false);
61             }
62         }
63         if (hideHead || hideSelf)
64         {
65             head.gameObject.SetActive(false);
66         }
67         if(rig.GetComponent<MovementProvider>().inCenter && rig)
68         {
69             MapCharacterPosition(character, XRNode.Head);
70         }
71     }
72 }
73
74 void MapPosition(Transform target, XRNode node)
75 {
76     InputDevices.GetDeviceAtXRNode(node).TryGetFeatureValue(CommonUsages
        .devicePosition, out Vector3 position);

```

```

77     InputDevices.GetDeviceAtXRNode(node).TryGetFeatureValue(CommonUsages
78         .deviceRotation, out Quaternion rotation);
79
80     target.position = position;
81     target.rotation = rotation;
82 }
83
84 void MapCharacterPosition(Transform characterModel, XRNode node)
85 {
86     InputDevices.GetDeviceAtXRNode(node).TryGetFeatureValue(CommonUsages
87         .devicePosition, out Vector3 position);
88     InputDevices.GetDeviceAtXRNode(node).TryGetFeatureValue(CommonUsages
89         .deviceRotation, out Quaternion rotation);
90
91     Transform rigTransform = rig.GetComponent<Transform>();
92
93     if(rigTransform.rotation.eulerAngles.x == 0)
94     {
95         Quaternion snapTurnOffset = rigTransform.rotation;
96         Vector3 characterControllerOffset = rigTransform.position;
97         //Vector3 headPosition = new Vector3(characterControllerOffset.x
98             , characterControllerOffset.y, characterControllerOffset.z);
99         Vector3 headPosition = new Vector3(characterControllerOffset.x +
100             (snapTurnOffset * position).x, characterControllerOffset.y,
101             characterControllerOffset.z + (snapTurnOffset * position).z
102             );
103         //Vector3 headRotation = new Vector3(0, rotation.eulerAngles.y +
104             (snapTurnOffset.eulerAngles.y), 0);
105         Vector3 headRotation = new Vector3(0, (snapTurnOffset.
106             eulerAngles.y), 0);
107         Vector3 headPositionOffset = Quaternion.Euler(headRotation) *
108             new Vector3(cameraFrontOffset, 0, 0);
109         //Debug.Log(headPositionOffset);
110
111         characterModel.position = headPosition - headPositionOffset;
112         characterModel.rotation = Quaternion.LookRotation(rigTransform.
113             right, rigTransform.up);
114         //Debug.Log(Quaternion.LookRotation(rigTransform.up,
115             rigTransform.forward));
116         //characterModel.rotation = Quaternion.Euler(headRotation);
117
118         //Parent to XR rig and stop moving dynamically
119         transform.SetParent(rig.transform);
120         baseVector = characterModel.localPosition;
121         parented = true;
122         //Invoke("Parented", 5f);
123     }
124 }
125
126 void MapLocalPosition(Transform characterModel, XRNode node)

```

```

115     {
116         InputDevices.GetDeviceAtXRNode(node).TryGetFeatureValue(CommonUsages
117             .devicePosition, out Vector3 position);
118         Transform rigTransform = rig.GetComponent<Transform>();
119
120         Vector3 headposition = new Vector3(position.x, 0, position.z);
121
122         characterModel.localPosition = baseVector + headposition;
123     }
124
125     private void Parented()
126     {
127         parented = true;
128     }
129
130     public void SetColor(Color shirt, Color pants)
131     {
132         transform.Find("Torso Male Short Sleeves").GetComponent<
133             SkinnedMeshRenderer>().materialSetColor("_BaseColor", shirt);
134         transform.Find("Pants Male").GetComponent<SkinnedMeshRenderer>().
135             materialSetColor("_BaseColor", pants);
136     }
137
138     private void Animate()
139     {
140         // Blend between walk/run
141         //float blend = rig.GetComponent<CharacterController>().velocity.
142         //magnitude;
143         //animator.SetFloat("Move", blend);
144     }
145
146     public void MuteSelf()
147     {
148         speaker.SetActive(false);
149     }
150
151     public void UnmuteSelf()
152     {
153         speaker.SetActive(true);
154     }
155
156     public void DisableSelf()
157     {
158         foreach (Transform mesh in meshes)
159         {
160             mesh.gameObject.SetActive(false);
161         }
162         head.gameObject.SetActive(false);
163         //MuteSelf();
164     }

```

```

161
162     public void EnableSelf()
163     {
164         foreach (Transform mesh in meshes)
165         {
166             mesh.gameObject.SetActive(true);
167         }
168         head.gameObject.SetActive(true);
169         //UnmuteSelf();
170     }
171
172     public Player GetOwner()
173     {
174         return photonView.Owner;
175     }
176
177 }
```

2.4 Avatar Random Customiser

Small script to handle randomisation of avatar shirt colors, so players can tell each other apart.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class AvatarRandomCustomiser : MonoBehaviour
6 {
7
8     // Start is called before the first frame update
9     void Start()
10    {
11
12    }
13
14
15     // Update is called once per frame
16     void Update()
17    {
18
19    }
20
21     public void SetColor(Color shirt, Color pants)
```

```

22     {
23         transform.Find("Torso Male Short Sleeves").GetComponent<
24             SkinnedMeshRenderer>().material.SetColor("_BaseColor", shirt);
25         transform.Find("Pants Male").GetComponent<SkinnedMeshRenderer>().
26             material.SetColor("_BaseColor", pants);
27     }
28 }
```

2.5 Grabbable Photon Ownership

This script handles the transfer of ownership when different players want to move synced objects.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.XR;
5 using UnityEngine.XR.Interaction.Toolkit;
6 using Photon.Realtime;
7 using Photon.Pun;
8
9 [RequireComponent( typeof( PhotonView ) )]
10 public class GrabbablePhotonOwnership : MonoBehaviourPun
11 {
12     private XRGrabInteractable interactable = null;
13     private void Awake()
14     {
15         interactable = GetComponent<XRGrabInteractable>();
16     }
17     private void OnEnable()
18     {
19         interactable.onSelectEnter.AddListener(RequestOwnership);
20     }
21     private void RequestOwnership(XRBaseInteractor interactor)
22     {
23         if( this.photonView.Owner == PhotonNetwork.LocalPlayer )
24         {
25             Debug.Log( "Not requesting ownership. Already mine." );
26             return;
27         }
28
29         this.photonView.RequestOwnership();
30         Debug.Log( "Requesting ownership" );
31     }
}
```

```
32
```

```
33 }
```

3 Player and XR Scripts

The main behaviour for VR/XR comes from the Unity XR plugin and is configured via existing script components and options. However, custom scripts are still needed to support all the additional behaviours available to the player as well as some custom functions for controls and teleportation not supported by the base package.

3.1 Player Controller

Simple script that sets the player's base position, should really be merged into another script because the player rig has lots of different scripts attached to it.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Pun;
5
6
7 public class PlayerController : MonoBehaviour
8 {
9     private Vector3 startPosition;
10    public int numVoiceRooms = 0;
11
12
13    // Start is called before the first frame update
14    void Start()
15    {
16        startPosition = transform.position;
```

```

17     }
18
19     // Update is called once per frame
20     void Update()
21     {
22
23     }
24
25     private void ResetPlayerPosition()
26     {
27         transform.position = startPosition;
28     }
29
30     public void TeleportPlayer(Vector3 target)
31     {
32
33     }
34
35 }
```

3.2 Movement Provider

Main script controlling player movement, gravity, etc.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.XR;
5 using UnityEngine.XR.Interaction.Toolkit;
6
7 public class MovementProvider : LocomotionProvider
8 {
9     public float speed = 1.0f;
10    public float gravityMultiplier = 1.0f;
11    public bool gravity = true;
12    public List<XRController> controllers = null;
13    public TeleportationProvider teleportationProvider;
14    private CharacterController characterController = null;
15    private GameObject head = null;
16    private bool resetClick = true;
17    public bool relativeGravity = true;
18    public bool inCenter = true;
19
20    //CUSTOM GRAVITY VARIABLES
```

```

22     float distanceToGround;
23     Vector3 Groundnormal;
24     bool OnGround = false;
25
26     protected override void Awake()
27     {
28         characterController = GetComponent<CharacterController>();
29         head = GetComponent<XRRig>().cameraGameObject;
30     }
31
32     // Start is called before the first frame update
33     private void Start()
34     {
35         PositionController();
36     }
37
38     private void Update()
39     {
40         // Ground Control (GRAVITY SCRIPT NOT WRITTEN BY ME)
41         RaycastHit hit = new RaycastHit();
42         if (Physics.Raycast(transform.position, -transform.up, out hit, 10))
43         {
44
45             distanceToGround = hit.distance;
46             Groundnormal = hit.normal;
47
48             if (distanceToGround <= 0.2f)
49             {
50                 OnGround = true;
51             }
52             else
53             {
54                 OnGround = false;
55             }
56
57         }
58     }
59 }
60
61     // Update is called once per frame
62     private void FixedUpdate()
63     {
64         PositionController();
65         if (!relativeGravity)
66         {
67             if (gravity)
68                 ApplyGravity();
69             else if (characterController.transform.position.y <= 9)
70                 gravity = true;
71     }

```

```

72     else
73     {
74         if(inCenter)
75             ApplyGravity();
76         //if relative gravity is on
77         //ApplyRelativeGravity();
78     }
79 }
80
81 public void PositionController()
82 {
83     //Get the head in local, playspace ground
84     float headHeight = Mathf.Clamp(head.transform.localPosition.y, 0.5f,
85         2);
86     characterController.height = headHeight;
87
88     //Cut in half, add skin
89     Vector3 newCenter = Vector3.zero;
90     newCenter.y = characterController.height / 2;
91     newCenter.y += characterController.skinWidth;
92
93     //Move the capsule in local space as well
94     newCenter.x = head.transform.localPosition.x;
95     newCenter.z = head.transform.localPosition.z;
96
97     //Apply
98     characterController.center = newCenter;
99 }
100
101 private void CheckForInput()
102 {
103 }
104
105 public void StartMove(Vector2 position)
106 {
107     //Add forward vector
108     Vector3 direction = new Vector3(position.x, 0, position.y);
109     Vector3 headRotation = head.transform.eulerAngles; //new Vector3(0,
110         head.transform.eulerAngles.y, 0);
111     Vector3 bodyRotation = transform.up;
112
113     direction = head.transform.rotation * direction; //raw direction
114         pointing in direction player head
115     direction = direction - bodyRotation * Vector3.Dot(direction,
116         bodyRotation); //direction constrained to XY plane of player, by
117         subtracting scalar projection of normal
118
119     //Apply speed and move
120     Vector3 movement = direction.normalized * speed;

```

```

117     characterController.Move(movement * Time.deltaTime);
118     XR_TestVariables.XRMoveCount += Vector3.Magnitude(movement * Time.
119         deltaTime);
120 }
121
122 public void StartMove(Vector3 position)
123 {
124     //Add forward vector
125     Vector3 direction = position;
126     Vector3 headRotation = head.transform.eulerAngles; //new Vector3(0,
127         head.transform.eulerAngles.y, 0);
128     Vector3 bodyRotation = transform.up;
129
130     direction = head.transform.rotation * direction; //raw direction
131         pointing in direction player head
132     //direction = direction - bodyRotation * Vector3.Dot(direction,
133         bodyRotation); //direction constrained to XY plane of player, by
134         subtracting scalar projection of normal
135
136     //Apply speed and move
137     Vector3 movement = direction.normalized * speed;
138     characterController.Move(movement);
139     XR_TestVariables.XRMoveCount += Vector3.Magnitude(movement * Time.
140         deltaTime);
141 }
142
143 private void ApplyGravity()
144 {
145     Vector3 gravity = new Vector3(0, Physics.gravity.y *
146         gravityMultiplier, 0);
147     gravity.y *= Time.deltaTime;
148
149     characterController.Move(gravity);
150 }
151
152 private void ApplyRelativeGravity()
153 {
154     //Vector3 sphericalGravDirection = new Vector3( transform.position.
155         normalized.z, 0, transform.position.normalized.y);
156     Vector3 relativeGravDirection = -transform.up;
157     Vector3 gravityTime = -relativeGravDirection * Physics.gravity.y *
158         gravityMultiplier * Time.deltaTime;
159
160     /*
161     if (OnGround == false)
162     {
163         characterController.Move(gravityTime);
164     }
165     */

```

```

158     characterController.Move(gravityTime);
159
160     //Quaternion toRotation = Quaternion.FromToRotation(transform.up,
161     //Groundnormal) * transform.rotation;
162     //transform.rotation = toRotation;
163 }
164
165 private void HighJump(bool jump)
166 {
167     Vector3 highJump = new Vector3(0, 11 - characterController.transform
168         .position.y, 0);
169
170     if (jump)
171     {
172         characterController.Move(highJump);
173         gravity = false;
174     }
175     else
176     {
177         characterController.Move(-highJump);
178         gravity = true;
179     }
180 }
181
182 public void VertMove(float magnitude)
183 {
184     //Debug.Log("VertMove " + magnitude);
185     characterController.Move(transform.up * magnitude * Time.deltaTime);
186     //var moveVector3 = new Vector3(0, magnitude, 0);
187     //StartMove(moveVector3);
188 }
189
190 void OnTriggerEnter(Collider collider)
191 {
192     if(collider.gameObject.name == "CenterSphere")
193     {
194         inCenter = true;
195         Debug.Log("in center!");
196     }
197 }
198
199 void OnTriggerExit(Collider collider)
200 {
201     if(collider.gameObject.name == "CenterSphere")
202     {
203         inCenter = false;
204         Debug.Log("out of center!");
205     }
206 }
```

```
206  
207 }
```

3.3 XR Controller Manager

Custom script I wrote that handles higher-level operations with the XR controllers, allowing behaviour like toggling raycast lines instead of having them on permanently.

```
1 using System.Collections;  
2 using System.Collections.Generic;  
3 using UnityEngine;  
4  
5 using UnityEngine.XR.Interaction.Toolkit.UI;  
6 using UnityEngine.EventSystems;  
7  
8 namespace UnityEngine.XR.Interaction.Toolkit  
9 {  
10     public class XRControllerManager : MonoBehaviour  
11     {  
12         public XRController LeftController;  
13         public XRController RightController;  
14         public XRController TeleportController;  
15         public List<XRController> controllers;  
16         private MovementProvider movementProvider;  
17         private XRRayInteractor TeleportInteractor;  
18         private XRInteractorLineVisual TeleportLine;  
19         private XRRayInteractor GrabRayInteractor;  
20         private XRInteractorLineVisual GrabRayLine;  
21         private GameObject rig;  
22         private GameObject cameraOffset;  
23         private GameObject teleportReticle;  
24         private GameObject teleportPlane;  
25         public bool TeleportMode = true;  
26         //public Dictionary<string, GameObject> spawnPrefabsList = new  
         //Dictionary<string, GameObject>();  
27  
28  
29         // Start is called before the first frame update  
30         void Start()  
31         {  
32             rig = GameObject.Find("XR Rig");  
33             movementProvider = transform.GetComponent<MovementProvider>();
```

```

34     cameraOffset = rig.transform.Find("Camera Offset").gameObject;
35     teleportReticle = cameraOffset.transform.Find(
36         "GroundTeleportReticle").gameObject;
37     teleportPlane = cameraOffset.transform.Find("MagicTeleportPlane"
38         ).gameObject;
39
40     TeleportInteractor = TeleportController.GetComponent<
41         XRRayInteractor>();
42     TeleportLine = TeleportController.GetComponent<
43         XRInteractorLineVisual>();
44     GrabRayInteractor = RightController.GetComponent<XRRayInteractor
45         >();
46     GrabRayLine = RightController.GetComponent<
47         XRInteractorLineVisual>();
48
49     CheckTeleportMode();
50
51     //spawnPrefabsList.Add("anchorBush", Resources.Load("Blobs/
52         BlobAnchorDynamic_Bush", typeof(GameObject)) as GameObject);
53 }
54
55 // Update is called once per frame
56 void FixedUpdate()
57 {
58     CheckForInput();
59     CheckForTeleport();
60     CheckInteractorRay();
61 }
62
63 private bool resetClick, resetClick2 = true;
64
65 private void CheckForInput()
66 {
67     if (TeleportController.inputDevice.TryGetFeatureValue(
68         CommonUsages.secondary2DAxisClick, out bool click))
69     {
70         if (click && resetClick)
71         {
72             TeleportMode = !TeleportMode;
73             CheckTeleportMode();
74             resetClick = false;
75         }
76         else if (!click)
77         {
78             resetClick = true;
79         }
80     }
81
82     if (TeleportController.inputDevice.TryGetFeatureValue(
83         CommonUsages.primary2DAxisClick, out bool click2))

```

```

75         {
76             if (click2 && resetClick2)
77             {
78                 TeleportMode = !TeleportMode;
79                 CheckTeleportMode();
80                 resetClick2 = false;
81             }
82             else if (!click2)
83             {
84                 resetClick2 = true;
85             }
86         }
87
88         /*
89         foreach (XRController controller in controllers)
90         {
91             if (controller.enableInputActions)
92                 CheckForMovement(controller.inputDevice);
93         }
94     */
95
96     CheckForMovement(LeftController.inputDevice);
97 }
98
99     private void CheckForMovement(InputDevice device)
100    {
101        if (device.TryGetFeatureValue(CommonUsages.primary2DAxis, out
102            Vector2 position))
103        {
104            //movementProvider.StartMove(position);
105            //Debug.Log(position);
106        }
107        if (device.TryGetFeatureValue(CommonUsages.secondary2DAxis, out
108            Vector2 position2) && position2.magnitude > 0.2f) //check >
109            0.2f for deadzone
110        {
111            movementProvider.StartMove(position2);
112            //Debug.Log(position2.magnitude);
113        }
114        /*
115        // "High Jump" float function
116        if (device.TryGetFeatureValue(CommonUsages.secondaryButton, out
117            bool click))
118        {
119            if (click && resetClick)
120            {
121                if (characterController.transform.position.y <= 12)
122                {
123                    HighJump(true);
124                    resetClick = false;

```

```

121         }
122         else
123         {
124             HighJump(false);
125             resetClick = false;
126         }
127     }
128     else if (!click && !resetClick)
129         resetClick = true;
130     }
131     */
132 }
133
134 private void CheckForTeleport()
135 {
136     if (TeleportController.inputDevice.TryGetFeatureValue(
137         CommonUsages.triggerButton, out bool triggerValue) &&
138         triggerValue)
139     {
140         TeleportInteractor.enabled = true;
141         TeleportLine.enabled = true;
142         if (TeleportMode == true)
143         {
144             teleportReticle.SetActive(true);
145         }
146         else teleportReticle.SetActive(false);
147     }
148     else if (!triggerValue && TeleportInteractor.enabled)
149     {
150         TeleportInteractor.enabled = false;
151         TeleportLine.enabled = false;
152         teleportReticle.SetActive(false);
153     }
154 }
155
156 private void CheckInteractorRay()
157 {
158     if (RightController.inputDevice.TryGetFeatureValue(CommonUsages.
159         triggerButton, out bool triggerValue) && triggerValue)
160     {
161         GrabRayInteractor.enabled = true;
162         GrabRayLine.enabled = true;
163     }
164     else if (!triggerValue && GrabRayInteractor.enabled)
165     {
166         GrabRayInteractor.enabled = false;
167         GrabRayLine.enabled = false;
168     }
169 }
```

```

168     private void CheckTeleportMode()
169     {
170         //Teleport mode 1 i.e. spatial teleport projectile curve
171         if (TeleportMode == true)
172         {
173             SetTeleportProjectile();
174         }
175         //Teleport mode 2 i.e. straightline select
176         else if (TeleportMode == false)
177         {
178             SetTeleportRaycast();
179         }
180     }
181
182     private void SetTeleportProjectile()
183     {
184         TeleportInteractor.lineType = XRRayInteractor.LineType.
185             ProjectileCurve;
186         //TeleportLine.AttachCustomReticle(teleportReticle);
187         //teleportPlane.SetActive(true);
188         teleportReticle.SetActive(true);
189     }
190
191     private void SetTeleportRaycast()
192     {
193         TeleportInteractor.lineType = XRRayInteractor.LineType.
194             StraightLine;
195         //TeleportLine.RemoveCustomReticle();
196         //teleportPlane.SetActive(false);
197         teleportReticle.SetActive(false);
198     }

```

3.4 Teleportation Anchored Area

Another custom XR script that modifies teleportation behaviour, allowing the player to aim at any part of the collision object but always teleport to a fixed point.

```

1 using System;
2 using System.Collections.Generic;
3 using UnityEditor;
```

```

4 using UnityEngine;
5
6
7 namespace UnityEngine.XR.Interaction.Toolkit
8 {
9     public class TeleportationAnchoredArea : BaseTeleportationInteractable
10    {
11        public GameObject TeleportationAnchor;
12
13        protected override bool GenerateTeleportRequest(XRBaseInteractor
14            interactor, RaycastHit raycastHit, ref TeleportRequest
15            teleportRequest)
16        {
17            teleportRequest.destinationPosition = TeleportationAnchor.
18                transform.position;
19            teleportRequest.destinationUpVector = TeleportationAnchor.
20                transform.up; // use the area transform for data.
21            teleportRequest.destinationForwardVector = transform.forward;
22            teleportRequest.destinationRotation = TeleportationAnchor.
23                transform.rotation;
24
25            GameObject.Find("XR Rig").GetComponent<MovementProvider>().
26                PositionController();
27            return true;
28        }
29    }
30 }
```

3.5 Comment Frame Teleporter

Script that handles the teleporting to a specific spot on the comment frame.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEditor;
4 using UnityEngine;
5
6 namespace UnityEngine.XR.Interaction.Toolkit
7 {
8     public class CommentFrameTeleporter : BaseTeleportationInteractable
9     {
10         private float teleportBufferDistance = 8.0f;
```

```

11     protected override bool GenerateTeleportRequest(XRBaseInteractor
12         interactor, RaycastHit raycastHit, ref TeleportRequest
13         teleportRequest)
14     {
15         teleportRequest.destinationPosition = raycastHit.point - (
16             transform.forward.normalized * teleportBufferDistance) - (
17                 transform.up.normalized * 1);
18         teleportRequest.destinationUpVector = transform.up; // use the
19             area transform for data.
20         teleportRequest.destinationForwardVector = -transform.right;
21         teleportRequest.destinationRotation = Quaternion.identity;//
22             transform.rotation;
23         return true;
24     }
25 }

```

3.6 Custom Teleportation Provider

Custom script instance copied from the default teleportation provider
that adds some additional functions

```

1 using System;
2 using UnityEngine;
3 using System.Collections.Generic;
4
5 namespace UnityEngine.XR.Interaction.Toolkit
6 {
7
8     public class CustomTeleportationProvider : LocomotionProvider
9     {
10
11         // the current teleportation request
12         TeleportRequest m_CurrentRequest;
13         // whether the current teleportation request is valid.
14         bool m_ValidRequest = false;
15
16
17         /// <summary>
18         /// This function will queue a teleportation request within the
19             provider.
20         /// </summary>
21         /// <param name="teleportRequest">The teleportation request</param>

```

```

21     /// <returns>true if successful.</returns>
22     public bool QueueTeleportRequest(TeleportRequest teleportRequest)
23     {
24         m_CurrentRequest = teleportRequest;
25         m_ValidRequest = true;
26         return true;
27     }
28
29     /// <summary>
30     /// Update function for the Teleportation Provider
31     /// </summary>
32     private void Update()
33     {
34         if(m_ValidRequest && BeginLocomotion())
35         {
36             var xrRig = system.xrRig;
37             if (xrRig != null)
38             {
39                 switch (m_CurrentRequest.matchOrientation)
40                 {
41                     case MatchOrientation.None:
42                         xrRig.MatchRigUp(m_CurrentRequest.
43                             destinationUpVector);
44                         break;
45                     case MatchOrientation.Camera:
46                         xrRig.MatchRigUpCameraForward(m_CurrentRequest .
47                             destinationUpVector, m_CurrentRequest .
48                             destinationForwardVector);
49                         break;
50                     //case MatchOrientation.Rig:
51                     //    xrRig.MatchRigUpRigForward(m_CurrentRequest .
52                     //        destinationUpVector, m_CurrentRequest .
53                     //        destinationForwardVector);
54                     //    break;
55                 }
56
57                 Vector3 heightAdjustment = xrRig.rig.transform.up *
58                 xrRig.cameraInRigSpaceHeight;
59
60                 Vector3 cameraDestination = m_CurrentRequest .
61                 destinationPosition + heightAdjustment;
62
63                 xrRig.MoveCameraToWorldLocation(cameraDestination);
64             }
65             EndLocomotion();
66             m_ValidRequest = false;
67         }
68     }
69 }

```

4 Reddit General Scripts

Reddit data is loaded using the Reddit.NET API interface. Probably not the most ideal due to synchronous lag which I haven't been able to solve, but it is relatively easy to understand and implement.

4.1 Reddit Controller

The big, main component script that handles the reddit interface and spawns all the subsequent posts and geometry which then execute their own scripts.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Reddit;
5 using Reddit.Controllers;
6 using Reddit.Controllers.EventArgs;
7 using TMPro;
8 using System.Linq;
9 using System.IO;
10
11 public class RedditController : MonoBehaviour
12 {
13     public RedditClient r;
14     //private string subredditTarget = "AskReddit";
15     private GameObject testPrefab;
16     private GameObject redPostPrefab;
17     private GameObject textTemplate;
18     private GameObject textTemplateColumn;
19     private GameObject textTemplateComment;
20     public GameObject torusTemplate;
21     public GameObject subredditStar;
22     public RedCentralSpaceMonobehaviour centralSpace;
23     public GameObject subredditTitleMesh;
24     public NetworkManager networkManager;
25     public bool generatePosts = true;
26     private string targetSubreddit;
27     private string refreshToken;
28     public List<GameObject> postObjects;
29     private List<Post> allPostList;
```

```

30     private List<(float, float)> postPropertiesList = new List<(float
31         upvoteScale, float commentWeight)>();
32     private List<string> subredditList;
33
34     //PARAMETERS
35     private float spawnRadius = 100.0f;
36     private int numRings = 3;
37     private int postsPerRing = 18;
38     private int numPosts = 50;
39
40     // Start is called before the first frame update
41     void Start()
42     {
43         //testPrefab = Resources.Load("Blobs/BlobAnchorStatic_Block", typeof(
44             GameObject)) as GameObject;
45
46         redPostPrefab = Resources.Load("Reddit/RedPostPlatform", typeof(
47             GameObject)) as GameObject;
48         textTemplate = Resources.Load("Generic_Text", typeof(GameObject)) as
49             GameObject;
50         textTemplateColumn = Resources.Load("Generic_Text_Column", typeof(
51             GameObject)) as GameObject;
52         textTemplateComment = Resources.Load("Generic_Text_Column_Smaller",
53             typeof(GameObject)) as GameObject;
54
55         targetSubreddit = File.ReadAllText(Application.streamingAssetsPath +
56             "/subreddit.txt");
57         refreshToken = File.ReadAllText(Application.streamingAssetsPath + "/"
58             "refreshtoken.txt");
59
60         r = new RedditClient("DHawrgYdt3OhPg", refreshToken);
61         Debug.Log("Username: " + r.Account.Me.Name);
62
63         GenerateSubreddits();
64         Initialise(targetSubreddit);
65         networkManager.SetPlayerSubreddit(targetSubreddit);
66
67
68         /*for (int i = 0; i < numRings + 1; i++)
69         {
70             GameObject ringDivider = Instantiate(torusTemplate, transform);
71             ringDivider.transform.localPosition = new Vector3(0, i * 30f, 0)
72                 ;
73         }*/
74         //UnityEngine.Debug.Log(r.Account.Me);
75     }
76
77     public void Initialise(string subreddit = "")
78     {
79         DestroyGeometry();
80
81         //InitialiseSubreddits();
82
83         //CreateSubreddits();
84
85         //CreateTorus();
86
87         //CreatePost();
88
89         //CreateComments();
90
91         //CreatePost();
92
93         //CreateComments();
94
95         //CreatePost();
96
97         //CreateComments();
98
99         //CreatePost();
100
101         //CreateComments();
102
103         //CreatePost();
104
105         //CreateComments();
106
107         //CreatePost();
108
109         //CreateComments();
110
111         //CreatePost();
112
113         //CreateComments();
114
115         //CreatePost();
116
117         //CreateComments();
118
119         //CreatePost();
120
121         //CreateComments();
122
123         //CreatePost();
124
125         //CreateComments();
126
127         //CreatePost();
128
129         //CreateComments();
130
131         //CreatePost();
132
133         //CreateComments();
134
135         //CreatePost();
136
137         //CreateComments();
138
139         //CreatePost();
140
141         //CreateComments();
142
143         //CreatePost();
144
145         //CreateComments();
146
147         //CreatePost();
148
149         //CreateComments();
150
151         //CreatePost();
152
153         //CreateComments();
154
155         //CreatePost();
156
157         //CreateComments();
158
159         //CreatePost();
160
161         //CreateComments();
162
163         //CreatePost();
164
165         //CreateComments();
166
167         //CreatePost();
168
169         //CreateComments();
170
171         //CreatePost();
172
173         //CreateComments();
174
175         //CreatePost();
176
177         //CreateComments();
178
179         //CreatePost();
180
181         //CreateComments();
182
183         //CreatePost();
184
185         //CreateComments();
186
187         //CreatePost();
188
189         //CreateComments();
190
191         //CreatePost();
192
193         //CreateComments();
194
195         //CreatePost();
196
197         //CreateComments();
198
199         //CreatePost();
200
201         //CreateComments();
202
203         //CreatePost();
204
205         //CreateComments();
206
207         //CreatePost();
208
209         //CreateComments();
210
211         //CreatePost();
212
213         //CreateComments();
214
215         //CreatePost();
216
217         //CreateComments();
218
219         //CreatePost();
220
221         //CreateComments();
222
223         //CreatePost();
224
225         //CreateComments();
226
227         //CreatePost();
228
229         //CreateComments();
230
231         //CreatePost();
232
233         //CreateComments();
234
235         //CreatePost();
236
237         //CreateComments();
238
239         //CreatePost();
240
241         //CreateComments();
242
243         //CreatePost();
244
245         //CreateComments();
246
247         //CreatePost();
248
249         //CreateComments();
250
251         //CreatePost();
252
253         //CreateComments();
254
255         //CreatePost();
256
257         //CreateComments();
258
259         //CreatePost();
260
261         //CreateComments();
262
263         //CreatePost();
264
265         //CreateComments();
266
267         //CreatePost();
268
269         //CreateComments();
270
271         //CreatePost();
272
273         //CreateComments();
274
275         //CreatePost();
276
277         //CreateComments();
278
279         //CreatePost();
280
281         //CreateComments();
282
283         //CreatePost();
284
285         //CreateComments();
286
287         //CreatePost();
288
289         //CreateComments();
290
291         //CreatePost();
292
293         //CreateComments();
294
295         //CreatePost();
296
297         //CreateComments();
298
299         //CreatePost();
300
301         //CreateComments();
302
303         //CreatePost();
304
305         //CreateComments();
306
307         //CreatePost();
308
309         //CreateComments();
310
311         //CreatePost();
312
313         //CreateComments();
314
315         //CreatePost();
316
317         //CreateComments();
318
319         //CreatePost();
320
321         //CreateComments();
322
323         //CreatePost();
324
325         //CreateComments();
326
327         //CreatePost();
328
329         //CreateComments();
330
331         //CreatePost();
332
333         //CreateComments();
334
335         //CreatePost();
336
337         //CreateComments();
338
339         //CreatePost();
340
341         //CreateComments();
342
343         //CreatePost();
344
345         //CreateComments();
346
347         //CreatePost();
348
349         //CreateComments();
350
351         //CreatePost();
352
353         //CreateComments();
354
355         //CreatePost();
356
357         //CreateComments();
358
359         //CreatePost();
360
361         //CreateComments();
362
363         //CreatePost();
364
365         //CreateComments();
366
367         //CreatePost();
368
369         //CreateComments();
370
371         //CreatePost();
372
373         //CreateComments();
374
375         //CreatePost();
376
377         //CreateComments();
378
379         //CreatePost();
380
381         //CreateComments();
382
383         //CreatePost();
384
385         //CreateComments();
386
387         //CreatePost();
388
389         //CreateComments();
390
391         //CreatePost();
392
393         //CreateComments();
394
395         //CreatePost();
396
397         //CreateComments();
398
399         //CreatePost();
400
401         //CreateComments();
402
403         //CreatePost();
404
405         //CreateComments();
406
407         //CreatePost();
408
409         //CreateComments();
410
411         //CreatePost();
412
413         //CreateComments();
414
415         //CreatePost();
416
417         //CreateComments();
418
419         //CreatePost();
420
421         //CreateComments();
422
423         //CreatePost();
424
425         //CreateComments();
426
427         //CreatePost();
428
429         //CreateComments();
430
431         //CreatePost();
432
433         //CreateComments();
434
435         //CreatePost();
436
437         //CreateComments();
438
439         //CreatePost();
440
441         //CreateComments();
442
443         //CreatePost();
444
445         //CreateComments();
446
447         //CreatePost();
448
449         //CreateComments();
450
451         //CreatePost();
452
453         //CreateComments();
454
455         //CreatePost();
456
457         //CreateComments();
458
459         //CreatePost();
460
461         //CreateComments();
462
463         //CreatePost();
464
465         //CreateComments();
466
467         //CreatePost();
468
469         //CreateComments();
470
471         //CreatePost();
472
473         //CreateComments();
474
475         //CreatePost();
476
477         //CreateComments();
478
479         //CreatePost();
480
481         //CreateComments();
482
483         //CreatePost();
484
485         //CreateComments();
486
487         //CreatePost();
488
489         //CreateComments();
490
491         //CreatePost();
492
493         //CreateComments();
494
495         //CreatePost();
496
497         //CreateComments();
498
499         //CreatePost();
500
501         //CreateComments();
502
503         //CreatePost();
504
505         //CreateComments();
506
507         //CreatePost();
508
509         //CreateComments();
510
511         //CreatePost();
512
513         //CreateComments();
514
515         //CreatePost();
516
517         //CreateComments();
518
519         //CreatePost();
520
521         //CreateComments();
522
523         //CreatePost();
524
525         //CreateComments();
526
527         //CreatePost();
528
529         //CreateComments();
530
531         //CreatePost();
532
533         //CreateComments();
534
535         //CreatePost();
536
537         //CreateComments();
538
539         //CreatePost();
540
541         //CreateComments();
542
543         //CreatePost();
544
545         //CreateComments();
546
547         //CreatePost();
548
549         //CreateComments();
550
551         //CreatePost();
552
553         //CreateComments();
554
555         //CreatePost();
556
557         //CreateComments();
558
559         //CreatePost();
560
561         //CreateComments();
562
563         //CreatePost();
564
565         //CreateComments();
566
567         //CreatePost();
568
569         //CreateComments();
570
571         //CreatePost();
572
573         //CreateComments();
574
575         //CreatePost();
576
577         //CreateComments();
578
579         //CreatePost();
580
581         //CreateComments();
582
583         //CreatePost();
584
585         //CreateComments();
586
587         //CreatePost();
588
589         //CreateComments();
590
591         //CreatePost();
592
593         //CreateComments();
594
595         //CreatePost();
596
597         //CreateComments();
598
599         //CreatePost();
600
601         //CreateComments();
602
603         //CreatePost();
604
605         //CreateComments();
606
607         //CreatePost();
608
609         //CreateComments();
610
611         //CreatePost();
612
613         //CreateComments();
614
615         //CreatePost();
616
617         //CreateComments();
618
619         //CreatePost();
620
621         //CreateComments();
622
623         //CreatePost();
624
625         //CreateComments();
626
627         //CreatePost();
628
629         //CreateComments();
630
631         //CreatePost();
632
633         //CreateComments();
634
635         //CreatePost();
636
637         //CreateComments();
638
639         //CreatePost();
640
641         //CreateComments();
642
643         //CreatePost();
644
645         //CreateComments();
646
647         //CreatePost();
648
649         //CreateComments();
650
651         //CreatePost();
652
653         //CreateComments();
654
655         //CreatePost();
656
657         //CreateComments();
658
659         //CreatePost();
660
661         //CreateComments();
662
663         //CreatePost();
664
665         //CreateComments();
666
667         //CreatePost();
668
669         //CreateComments();
670
671         //CreatePost();
672
673         //CreateComments();
674
675         //CreatePost();
676
677         //CreateComments();
678
679         //CreatePost();
680
681         //CreateComments();
682
683         //CreatePost();
684
685         //CreateComments();
686
687         //CreatePost();
688
689         //CreateComments();
690
691         //CreatePost();
692
693         //CreateComments();
694
695         //CreatePost();
696
697         //CreateComments();
698
699         //CreatePost();
700
701         //CreateComments();
702
703         //CreatePost();
704
705         //CreateComments();
706
707         //CreatePost();
708
709         //CreateComments();
710
711         //CreatePost();
712
713         //CreateComments();
714
715         //CreatePost();
716
717         //CreateComments();
718
719         //CreatePost();
720
721         //CreateComments();
722
723         //CreatePost();
724
725         //CreateComments();
726
727         //CreatePost();
728
729         //CreateComments();
730
731         //CreatePost();
732
733         //CreateComments();
734
735         //CreatePost();
736
737         //CreateComments();
738
739         //CreatePost();
740
741         //CreateComments();
742
743         //CreatePost();
744
745         //CreateComments();
746
747         //CreatePost();
748
749         //CreateComments();
750
751         //CreatePost();
752
753         //CreateComments();
754
755         //CreatePost();
756
757         //CreateComments();
758
759         //CreatePost();
760
761         //CreateComments();
762
763         //CreatePost();
764
765         //CreateComments();
766
767         //CreatePost();
768
769         //CreateComments();
770
771         //CreatePost();
772
773         //CreateComments();
774
775         //CreatePost();
776
777         //CreateComments();
778
779         //CreatePost();
780
781         //CreateComments();
782
783         //CreatePost();
784
785         //CreateComments();
786
787         //CreatePost();
788
789         //CreateComments();
790
791         //CreatePost();
792
793         //CreateComments();
794
795         //CreatePost();
796
797         //CreateComments();
798
799         //CreatePost();
800
801         //CreateComments();
802
803         //CreatePost();
804
805         //CreateComments();
806
807         //CreatePost();
808
809         //CreateComments();
810
811         //CreatePost();
812
813         //CreateComments();
814
815         //CreatePost();
816
817         //CreateComments();
818
819         //CreatePost();
820
821         //CreateComments();
822
823         //CreatePost();
824
825         //CreateComments();
826
827         //CreatePost();
828
829         //CreateComments();
830
831         //CreatePost();
832
833         //CreateComments();
834
835         //CreatePost();
836
837         //CreateComments();
838
839         //CreatePost();
840
841         //CreateComments();
842
843         //CreatePost();
844
845         //CreateComments();
846
847         //CreatePost();
848
849         //CreateComments();
850
851         //CreatePost();
852
853         //CreateComments();
854
855         //CreatePost();
856
857         //CreateComments();
858
859         //CreatePost();
860
861         //CreateComments();
862
863         //CreatePost();
864
865         //CreateComments();
866
867         //CreatePost();
868
869         //CreateComments();
870
871         //CreatePost();
872
873         //CreateComments();
874
875         //CreatePost();
876
877         //CreateComments();
878
879         //CreatePost();
880
881         //CreateComments();
882
883         //CreatePost();
884
885         //CreateComments();
886
887         //CreatePost();
888
889         //CreateComments();
890
891         //CreatePost();
892
893         //CreateComments();
894
895         //CreatePost();
896
897         //CreateComments();
898
899         //CreatePost();
900
901         //CreateComments();
902
903         //CreatePost();
904
905         //CreateComments();
906
907         //CreatePost();
908
909         //CreateComments();
910
911         //CreatePost();
912
913         //CreateComments();
914
915         //CreatePost();
916
917         //CreateComments();
918
919         //CreatePost();
920
921         //CreateComments();
922
923         //CreatePost();
924
925         //CreateComments();
926
927         //CreatePost();
928
929         //CreateComments();
930
931         //CreatePost();
932
933         //CreateComments();
934
935         //CreatePost();
936
937         //CreateComments();
938
939         //CreatePost();
940
941         //CreateComments();
942
943         //CreatePost();
944
945         //CreateComments();
946
947         //CreatePost();
948
949         //CreateComments();
950
951         //CreatePost();
952
953         //CreateComments();
954
955         //CreatePost();
956
957         //CreateComments();
958
959         //CreatePost();
960
961         //CreateComments();
962
963         //CreatePost();
964
965         //CreateComments();
966
967         //CreatePost();
968
969         //CreateComments();
970
971         //CreatePost();
972
973         //CreateComments();
974
975         //CreatePost();
976
977         //CreateComments();
978
979         //CreatePost();
980
981         //CreateComments();
982
983         //CreatePost();
984
985         //CreateComments();
986
987         //CreatePost();
988
989         //CreateComments();
990
991         //CreatePost();
992
993         //CreateComments();
994
995         //CreatePost();
996
997         //CreateComments();
998
999         //CreatePost();
1000
1001         //CreateComments();
1002
1003         //CreatePost();
1004
1005         //CreateComments();
1006
1007         //CreatePost();
1008
1009         //CreateComments();
1010
1011         //CreatePost();
1012
1013         //CreateComments();
1014
1015         //CreatePost();
1016
1017         //CreateComments();
1018
1019         //CreatePost();
1020
1021         //CreateComments();
1022
1023         //CreatePost();
1024
1025         //CreateComments();
1026
1027         //CreatePost();
1028
1029         //CreateComments();
1030
1031         //CreatePost();
1032
1033         //CreateComments();
1034
1035         //CreatePost();
1036
1037         //CreateComments();
1038
1039         //CreatePost();
1040
1041         //CreateComments();
1042
1043         //CreatePost();
1044
1045         //CreateComments();
1046
1047         //CreatePost();
1048
1049         //CreateComments();
1050
1051         //CreatePost();
1052
1053         //CreateComments();
1054
1055         //CreatePost();
1056
1057         //CreateComments();
1058
1059         //CreatePost();
1060
1061         //CreateComments();
1062
1063         //CreatePost();
1064
1065         //CreateComments();
1066
1067         //CreatePost();
1068
1069         //CreateComments();
1070
1071         //CreatePost();
1072
1073         //CreateComments();
1074
1075         //CreatePost();
1076
1077         //CreateComments();
1078
1079         //CreatePost();
1080
1081         //CreateComments();
1082
1083         //CreatePost();
1084
1085         //CreateComments();
1086
1087         //CreatePost();
1088
1089         //CreateComments();
1090
1091         //CreatePost();
1092
1093         //CreateComments();
1094
1095         //CreatePost();
1096
1097         //CreateComments();
1098
1099         //CreatePost();
1100
1101         //CreateComments();
1102
1103         //CreatePost();
1104
1105         //CreateComments();
1106
1107         //CreatePost();
1108
1109         //CreateComments();
1110
1111         //CreatePost();
1112
1113         //CreateComments();
1114
1115         //CreatePost();
1116
1117         //CreateComments();
1118
1119         //CreatePost();
1120
1121         //CreateComments();
1122
1123         //CreatePost();
1124
1125         //CreateComments();
1126
1127         //CreatePost();
1128
1129         //CreateComments();
1130
1131         //CreatePost();
1132
1133         //CreateComments();
1134
1135         //CreatePost();
1136
1137         //CreateComments();
1138
1139         //CreatePost();
1140
1141         //CreateComments();
1142
1143         //CreatePost();
1144
1145         //CreateComments();
1146
1147         //CreatePost();
1148
1149         //CreateComments();
1150
1151         //CreatePost();
1152
1153         //CreateComments();
1154
1155         //CreatePost();
1156
1157         //CreateComments();
1158
1159         //CreatePost();
1160
1161         //CreateComments();
1162
1163         //CreatePost();
1164
1165         //CreateComments();
1166
1167         //CreatePost();
1168
1169         //CreateComments();
1170
1171         //CreatePost();
1172
1173         //CreateComments();
1174
1175         //CreatePost();
1176
1177         //CreateComments();
1178
1179         //CreatePost();
1180
1181         //CreateComments();
1182
1183         //CreatePost();
1184
1185         //CreateComments();
1186
1187         //CreatePost();
1188
1189         //CreateComments();
1190
1191         //CreatePost();
1192
1193         //CreateComments();
1194
1195         //CreatePost();
1196
1197         //CreateComments();
1198
1199         //CreatePost();
1200
1201         //CreateComments();
1202
1203         //CreatePost();
1204
1205         //CreateComments();
1206
1207         //CreatePost();
1208
1209         //CreateComments();
1210
1211         //CreatePost();
1212
1213         //CreateComments();
1214
1215         //CreatePost();
1216
1217         //CreateComments();
1218
1219         //CreatePost();
1220
1221         //CreateComments();
1222
1223         //CreatePost();
1224
1225         //CreateComments();
1226
1227         //CreatePost();
1228
1229         //CreateComments();
1230
1231         //CreatePost();
1232
1233         //CreateComments();
1234
1235         //CreatePost();
1236
1237         //CreateComments();
1238
1239         //CreatePost();
1240
1241         //CreateComments();
1242
1243         //CreatePost();
1244
1245         //CreateComments();
1246
1247         //CreatePost();
1248
1249         //CreateComments();
1250
1251         //CreatePost();
1252
1253         //CreateComments();
1254
1255         //CreatePost();
1256
1257         //CreateComments();
1258
1259         //CreatePost();
1260
1261         //CreateComments();
1262
1263         //CreatePost();
1264
1265         //CreateComments();
1266
1267         //CreatePost();
1268
1269         //CreateComments();
1270
1271         //CreatePost();
1272
1273         //CreateComments();
1274
1275         //CreatePost();
1276
1277         //CreateComments();
1278
1279         //CreatePost();
1280
1281         //CreateComments();
1282
1283         //CreatePost();
1284
1285         //CreateComments();
1286
1287         //CreatePost();
1288
1289         //CreateComments();
1290
1291         //CreatePost();
1292
1293         //CreateComments();
1294
1295         //CreatePost();
1296
1297         //CreateComments();
1298
1299         //CreatePost();
1300
1301         //CreateComments();
1302
1303         //CreatePost();
1304
1305         //CreateComments();
1306
1307         //CreatePost();
1308
1309         //CreateComments();
1310
1311         //CreatePost();
1312
1313         //CreateComments();
1314
1315         //CreatePost();
1316
1317         //CreateComments();
1318
1319         //CreatePost();
1320
1321         //CreateComments();
1322
1323         //CreatePost();
1324
1325         //CreateComments();
1326
1327         //CreatePost();
1328
1329         //CreateComments();
1330
1331         //CreatePost();
1332
1333         //CreateComments();
1334
1335         //CreatePost();
1336
1337         //CreateComments();
1338
1339         //CreatePost();
1340
1341         //CreateComments();
1342
1343         //CreatePost();
1344
1345         //CreateComments();
1346
1347         //CreatePost();
1348
1349         //CreateComments();
1350
1351         //CreatePost();
1352
1353         //CreateComments();
1354
1355         //CreatePost();
1356
1357         //CreateComments();
1358
1359         //CreatePost();
1360
1361         //CreateComments();
1362
1363         //CreatePost();
1364
1365         //CreateComments();
1366
1367         //CreatePost();
1368
1369         //CreateComments();
1370
1371         //CreatePost();
1372
1373         //CreateComments();
1374
1375         //CreatePost();
1376
1377         //CreateComments();
1378
1379         //CreatePost();
1380
1381         //CreateComments();
1382
1383         //CreatePost();
1384
1385         //CreateComments();
1386
1387         //CreatePost();
1388
1389         //CreateComments();
1390
1391         //CreatePost();
1392
1393         //CreateComments();
1394
1395         //CreatePost();
1396
1397         //CreateComments();
1398
1399         //CreatePost();
1400
1401         //CreateComments();
1402
1403         //CreatePost();
1404
1405         //CreateComments();
1406
1407         //CreatePost();
1408
1409         //CreateComments();
1410
1411         //CreatePost();
1412
1413         //CreateComments();
1414
1415         //CreatePost();
1416
1417         //CreateComments();
1418
1419         //CreatePost();
1420
1421         //CreateComments();
1422
1423         //CreatePost();
1424
1425         //CreateComments();
1426
1427         //CreatePost();
1428
1429         //CreateComments();
1430
1431         //CreatePost();
1432
1433         //CreateComments();
1434
1435         //CreatePost();
1436
1437         //CreateComments();
1438
1439         //CreatePost();
1440
1441         //CreateComments();
1442
1443         //CreatePost();
1444
1445         //CreateComments();
1446
1447         //CreatePost();
1448
1449         //CreateComments();
1450
1451         //CreatePost();
1452
1453         //CreateComments();
1454
1455         //CreatePost();
1456
1457         //CreateComments();
1458
1459         //CreatePost();
1460
1461         //CreateComments();
1462
1463         //CreatePost();
1464
1465         //CreateComments();
1466
1467         //CreatePost();
1468
1469         //CreateComments();
1470
1471         //CreatePost();
1472
1473         //CreateComments();
1474
1475         //CreatePost();
1476
1477         //CreateComments();
1478
1479         //CreatePost();
1480
1481         //CreateComments();
1482
1483         //CreatePost();
1484
1485         //CreateComments();
1486
1487         //CreatePost();
1488
1489         //CreateComments();
1490
1491         //CreatePost();
1492
1493         //CreateComments();
1494
1495         //CreatePost();
1496
1497         //CreateComments();
1498
1499         //CreatePost();
1500
1501         //CreateComments();
1502
1503         //CreatePost();
1504
1505         //CreateComments();
1506
1507         //CreatePost();
1508
1509         //CreateComments();
1510
1511         //CreatePost();
1512
1513         //CreateComments();
1514
1515         //CreatePost();
1516
1517         //CreateComments();
1518
1519         //CreatePost();
1520
1521         //CreateComments();
1522
1523         //CreatePost();
1524
1525         //CreateComments();
1526
1527         //CreatePost();
1528
1529         //CreateComments();
1530
1531         //CreatePost();
1532
1533         //CreateComments();
1534
1535         //CreatePost();
1536
1537         //CreateComments();
1538
1539         //CreatePost();
1540
1541         //CreateComments();
1542
1543         //CreatePost();
1544
1545         //CreateComments();
1546
1547         //CreatePost();
1548
1549         //CreateComments();
1550
1551         //CreatePost();
1552
1553         //CreateComments();
1554
1555         //CreatePost();
1556
1557         //CreateComments();
1558
1559         //CreatePost();
1560
1561         //CreateComments();
1562
1563         //CreatePost();
1564
1565         //CreateComments();
1566
1567         //CreatePost();
1568
1569         //CreateComments();
1570
1571         //CreatePost();
1572
1573         //CreateComments();
1574
1575         //CreatePost();
1576
1577         //CreateComments();
1578
1579         //CreatePost();
1580
1581         //CreateComments();
1582
1583         //CreatePost();
1584
1585         //CreateComments();
1586
1587         //CreatePost();
1588
1589         //CreateComments();
1590
1591         //CreatePost();
1592
1593         //CreateComments();
1594
1595         //CreatePost();
1596
1597         //CreateComments();
1598
1599         //CreatePost();
1600
1601         //CreateComments();
1602
1603         //CreatePost();
1604
1605         //CreateComments();
1606
1607         //CreatePost();
1608
1609         //CreateComments();
1610
1611         //CreatePost();
1612
1613         //CreateComments();
1614
1615         //CreatePost();
1616
1617         //CreateComments();
1618
1619         //CreatePost();
1620
1621         //CreateComments();
1622
1623         //CreatePost();
1624
1625         //CreateComments();
1626
1627         //CreatePost();
1628
1629         //CreateComments();
1630
1631         //CreatePost();
1632
1633         //CreateComments();
1634
1635         //CreatePost();
1636
1637         //CreateComments();
1638
1639         //CreatePost();
1640
1641         //CreateComments();
1642
1643         //CreatePost();
1644
1645         //CreateComments();
1646
1647         //CreatePost();
1648
1649         //CreateComments();
1650
1651         //CreatePost();
1652
1653         //CreateComments();
1654
1655         //CreatePost();
1656
1657         //CreateComments();
1658
1659         //CreatePost();
1660
1661         //CreateComments();
1662
1663         //CreatePost();
1664
1665         //CreateComments();
1666
1667         //CreatePost();
1668
1669         //CreateComments();
1670
1671         //CreatePost();
1672
1673         //CreateComments();
1674
1675         //CreatePost();
1676
1677         //CreateComments();
1678
1679         //CreatePost();
1680
1681         //CreateComments();
1682
1683         //CreatePost();
1684
1685         //CreateComments();
1686
1687         //CreatePost();
1688
1689         //CreateComments();
1690
1691         //CreatePost();
1692
1693         //CreateComments();
1694
1695         //CreatePost();
1696
1697         //CreateComments();
1698
1699         //CreatePost();
1700
1701         //CreateComments();
1702
1703         //CreatePost();
1704
1705         //CreateComments();
1706
1707         //CreatePost();
1708
1709         //CreateComments();
1710
1711         //CreatePost();
1712
1713         //CreateComments();
1714
1715         //CreatePost();
1716
1717         //CreateComments();
1718
1719         //CreatePost();
1720
1721         //CreateComments();
1722
1723         //CreatePost();
1724
1725         //CreateComments();
1726
1727         //CreatePost();
1728
1729         //CreateComments();
1730
1731         //CreatePost();
1732
1733         //CreateComments();
1734

```

```

71     targetSubreddit = subreddit;
72     centralSpace.Initialise(r.Subreddit(targetSubreddit));
73     allPostList = GetTopPosts(targetSubreddit, numPosts);
74     Debug.Log("Total posts:" + allPostList.Count);
75     ProcessPostData();
76
77     int oneRingCount = Mathf.FloorToInt(postsPerRing);
78
79     //Calculate which posts in the ring first
80     List<int> ringBreakpoints = new List<int>();
81     ringBreakpoints.Add(0);
82     float sumWeights = postPropertiesList.Sum(x => x.Item2 + 1);
83     //Debug.Log("sum weights :" + sumWeights);
84
85     float totalWeight = 0;
86     for (int p = 0; p < allPostList.Count; p++)
87     {
88         if(totalWeight >= sumWeights/(numRings))
89         {
90             totalWeight = 0;
91             ringBreakpoints.Add(p);
92             //Debug.Log("breakpoint: " + p);
93         }
94         totalWeight += postPropertiesList[p].Item2 + 1;
95     }
96     if(ringBreakpoints.Count < numRings + 1)
97     {
98         ringBreakpoints.Add(allPostList.Count - 1);
99     }
100    //Debug.Log(ringBreakpoints.Count);
101
102    for (int i = 0; i < numRings; i++)
103    {
104        //Debug.Log("Generating ring - " + ringBreakpoints[i] + "-" +
105        ringBreakpoints[i+1]);
106        if(generatePosts) GeneratePostGeometry(allPostList.GetRange(
107            ringBreakpoints[i], ringBreakpoints[i+1]-ringBreakpoints[i])
108            ,
109            postPropertiesList.GetRange(
110                ringBreakpoints[i], ringBreakpoints
111                [i+1]-ringBreakpoints[i]), 15f + (i
112                * 30f));
113        //Debug.Log("Ring " + i + " Generated");
114        /*
115        if ((i + 1) * postsPerRing <= postList.Count)
116        {
117            GeneratePostGeometry(postList.GetRange(i * postsPerRing,
118                postsPerRing), 10f + (i * 30f));
119        }
120        */

```

```

114     }
115 }
116
117 public void DestroyGeometry()
118 {
119     foreach (GameObject post in postObjects)
120     {
121         GameObject.Destroy(post);
122     }
123 }
124
125 // Update is called once per frame
126 void Update()
127 {
128
129 }
130
131 public List<Post> GetTopPosts(string subreddit, int num)
132 {
133     return r.Subreddit(subreddit).Posts.Hot.GetRange(0, num);
134 }
135
136 public void GeneratePostGeometry(List<Post> postList, List<(float, float
137     )> postProperties, float yoffset)
138 {
139     int numObjects = postList.Count;
140     var postWeights = new List<float>();
141
142     for (int i = 0; i < numObjects; i++)
143     {
144         //var weightCalc = Mathf.RoundToInt(Mathf.Clamp((postList[i].
145             Listing.NumComments / 200f), 0f, 1f) * 5);
146         postWeights.Add(1f + postProperties[i].Item2);
147     }
148
149     //postWeights = postList.Select(x => (5 + Mathf.RoundToInt(Mathf.
150         Clamp((x.UpVotes / 40000f), 0f, 1f) * 5)).ToList<int>();
151     float totalWeight = postWeights.Sum();
152     //Debug.Log(totalWeight);
153
154     var cumulativeAngle = 0f;
155
156     for (int i = 0; i < numObjects; i++)
157     {
158         Post currentPost = postList[i];
159         string postTitle = currentPost.Title;
160
161         //Circle geometry math
162         float circumference = 2 * Mathf.PI * spawnRadius;

```

```

161     float postAngleShare = postWeights[i] / totalWeight;
162     float angle = postAngleShare * 2 * Mathf.PI;
163     int postLengthUnits = Mathf.FloorToInt(2 * spawnRadius * Mathf.
164         Sin(angle / 2) / 4) - 1;
165
166     //float angle = i * Mathf.PI * 2 / numObjects;
167     cumulativeAngle += angle / 2; //add half because you actually
168         want the center of the angle
169     float x = Mathf.Cos(cumulativeAngle) * spawnRadius;
170     float z = Mathf.Sin(cumulativeAngle) * spawnRadius;
171     Vector3 pos = transform.position + new Vector3(x, yoffset, z);
172     float angleDegrees = -cumulativeAngle * Mathf.Rad2Deg;
173     Quaternion rot = Quaternion.Euler(270, angleDegrees, 90);
174
175
176     //Instantiate platform geometry
177     GameObject currentPlatform = Instantiate(redPostPrefab, pos, rot
178         );
179     currentPlatform.GetComponent<RedPostMonobehaviour>().
180         InitialisePost(currentPost, currentPost.Comments,
181             postLengthUnits, postProperties[i].Item1);
182
183     cumulativeAngle += angle / 2; //add half again to get the
184         actual cumulative angle
185
186     postObjects.Add(currentPlatform);
187
188 }
189
190 public void ProcesspostData()
191 {
192     postPropertiesList.Clear();
193     for(int i = 0; i < allPostList.Count; i++)
194     {
195         Post tempPost = allPostList[i];
196         int maxUpvotes = allPostList.Max(x => x.UpVotes);
197         int maxComments = allPostList.Max(x => x.Listing.NumComments);
198
199         float scaledUpvotes = (float)tempPost.UpVotes/(float)maxUpvotes;
200         float scaledComments = tempPost.Listing.NumComments/(float)
201             maxComments;
202
203         scaledUpvotes = Mathf.Ceil(scaledUpvotes * 100) / 100f;
204
205         postPropertiesList.Add((scaledUpvotes, scaledComments));
206     }
207 }

```

```

204     public void GenerateSubreddits()
205     {
206         List<Subreddit> subredditList = r.GetSubreddits("popular", 100);
207         int num = subredditList.Count;
208         List<string> subredditNames = subredditList.Select(x => x.Name.
209             ToString()).ToList();
210
211         Vector3 center = new Vector3(0, 0, 0);
212         for (int i = 0; i < num; i++)
213         {
214             //Vector sphere math to position stars at random point on
215             //constrained sphere
216             float thetaAngle = Random.Range(5, 45);
217             float phiAngle = Random.Range(0, 360);
218             Quaternion rotationAngle = Quaternion.Euler(thetaAngle, phiAngle
219                 , 0);
220
221             Vector3 pos = rotationAngle * new Vector3(0, 2000, 0);
222
223             //Vector3 pos = Random.onUnitSphere * 2000.0f;
224
225             Quaternion rot = Quaternion.FromToRotation(Vector3.up, center -
226                 pos);
227             GameObject star = Instantiate(subredditStar, pos, rot);
228
229             //Debug.Log(subredditList[i].Name);
230             string subredditName = subredditList[i].Name;
231             star.GetComponent<SubredditStarMonobehaviour>().InitialiseStar(
232                 subredditName);
233
234         }
235
236         GameObject allstar = Instantiate(subredditStar, new Vector3(0, 2000, 0
237             ), Quaternion.FromToRotation(Vector3.up, new Vector3(0, -2000, 0)));
238         allstar.GetComponent<SubredditStarMonobehaviour>().InitialiseStar("all");
239     }
240
241     public void GenerateOAuth()
242     {
243         //AuthTokenRetrieverLib authTokenRetrieverLib = new
244         //AuthTokenRetrieverLib(appId, appSecret, port);
245     }
246 }
```

4.2 Reddit Central Space

Script controlling the behaviour of the central space, including scaling management.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Reddit;
5 using Reddit.Controllers;
6 using TMPro;
7 using System;
8 using System.Linq;
9
10 public class RedCentralSpaceMonobehaviour : MonoBehaviour
11 {
12
13     public List<GameObject> nodeList = new List<GameObject>();
14     public GameObject centralPlatform;
15     public GameObject subredditTitleMesh;
16     public GameObject boundaryRingPrefab;
17     public GameObject boundarySphere;
18     public float distanceMargin = 10;
19     public float minScale = 20;
20
21     //Boundary Parameters
22     public float ringWidth = 3;
23     public float ringHeight = 1;
24     public int numSteps = 3;
25     public float stepWidth = 1;
26     private GameObject boundaryRing;
27     private List<GameObject> stepList = new List<GameObject>();
28
29     //adjustment helper parameters
30     private float maxDistance;
31     private float targetScale;
32     private bool startTransform = true;
33     private float scaleRate = 0.005f;
34     private float nextUpdateTime = 0.0f;
35     private float updateInterval = 2f;
36
37     //Subreddit Data
38     private Subreddit sr;
39     private string subredditName = "all";
40     //private List<string> infoModerators;
41     private string infoDescription;
42     private string infoSidebar;
```

```

43     private List<string> sidebarPanelStrings;
44     private List<GameObject> sidebarPanelObjects = new List<GameObject>();
45     public GameObject subredditInfoParent;
46     public GameObject sidebarPanelPrefab;
47     private Reddit.Controllers.Internal.Lists infoLists;
48
49     // Start is called before the first frame update
50     void Start()
51     {
52         CreateBoundaryRing();
53         GetAllNodes();
54         InstantScale();
55         //targetScale = centralPlatform.transform.localScale.x;
56     }
57
58     public void Initialise(Subreddit subreddit)
59     {
60         ClearAll();
61
62         if (subreddit.Name != "all")
63         {
64             sr = subreddit.About();
65             subredditName = sr.Name;
66
67             if (sr.Description != null) infoDescription = sr.Description;
68             if (sr.Sidebar != null)
69             {
70                 infoSidebar = sr.Sidebar;
71                 sidebarPanelStrings = new List<string>(infoSidebar.Split(new
72                     string[] { "***" }, StringSplitOptions.None));
73                 GeneratePanels();
74             }
75             if (sr.Lists != null) infoLists = sr.Lists;
76         }
77         else
78         {
79             subredditName = "all";
80         }
81
82         //Set geometry stuff
83         RefreshGeometry();
84     }
85
86     public void RefreshGeometry()
87     {
88         subredditTitleMesh.GetComponent<TextMeshPro>().text = "r/" +
89         subredditName;
90     }

```

```

91
92     public void ClearAll()
93     {
94         if (sidebarPanelStrings != null) sidebarPanelStrings.Clear();
95         DestroyPanels();
96     }
97
98     public void GeneratePanels()
99     {
100        DestroyPanels();
101
102        var cumulativeAngle = Mathf.PI / 6;
103        var radius = 6.5f;
104
105        for (int i = 0; i < sidebarPanelStrings.Count; i++)
106        {
107            float angleIncrease = Mathf.PI / 12;
108
109            cumulativeAngle = -cumulativeAngle; //flip sides back and forth
110
111            float x = Mathf.Cos(cumulativeAngle) * radius;
112            float z = Mathf.Sin(cumulativeAngle) * radius;
113            Vector3 pos = new Vector3(x, -2.25f, z);
114
115            float angleDegrees = -cumulativeAngle * Mathf.Rad2Deg;
116            Quaternion rot = Quaternion.Euler(0, angleDegrees + 90, 0);
117
118            GameObject panel = GameObject.Instantiate(sidebarPanelPrefab,
119                pos, rot);
120            panel.transform.SetParent(subredditInfoParent.transform);
121
122            panel.GetComponent<RedInfoPanelMonobehaviour>().SetText(
123                sidebarPanelStrings[i]);
124
125            sidebarPanelObjects.Add(panel);
126
127            if (i % 2 == 1)
128            {
129                cumulativeAngle += angleIncrease;
130            }
131        }
132
133        public void DestroyPanels()
134        {
135            for (int i = 0; i < sidebarPanelObjects.Count; i++)
136            {
137                GameObject.Destroy(sidebarPanelObjects[i]);
138            }
139        }

```

```

139
140     // Update is called once per frame
141     void Update()
142     {
143
144         if (Time.time > nextUpdateTime)
145         {
146             nextUpdateTime += updateInterval;
147
148             CheckDistances();
149
150             var platformRadius = (maxDistance + distanceMargin + stepWidth *
151                                   numSteps);
152
153             if (platformRadius * 2 > minScale && Mathf.Abs(platformRadius *
154                                         2 - centralPlatform.transform.localScale.x) > 1 &&
155                                         platformRadius < 80)
156             {
157                 ChangeTargetScale(Mathf.Round((maxDistance + distanceMargin
158                                         + stepWidth * numSteps) * 2));
159             }
160
161             scaleRate = 0.01f * Mathf.Clamp01(Mathf.Abs(platformRadius * 2 -
162                                         centralPlatform.transform.localScale.x) / (distanceMargin *
163                                         2));
164         }
165
166         if (startTransform)
167         {
168             if (centralPlatform.transform.localScale.x - targetScale < -1f)
169             {
170                 centralPlatform.transform.localScale += new Vector3(1, 0, 1)
171                               * scaleRate;
172                 float sphereScale = centralPlatform.transform.localScale.x +
173                               2 * ringWidth + 2;
174                 boundarySphere.transform.localScale = new Vector3(
175                               sphereScale + 1f, sphereScale + 1f, sphereScale + 1f);
176                 UpdateBoundaryRing();
177             }
178             else if (centralPlatform.transform.localScale.x - targetScale >
179                     1f)
180             {
181                 centralPlatform.transform.localScale += new Vector3(-1, 0, -1)
182                               * scaleRate;
183                 float sphereScale = centralPlatform.transform.localScale.x +
184                               2 * ringWidth + 2;
185                 boundarySphere.transform.localScale = new Vector3(
186                               sphereScale + 1f, sphereScale + 1f, sphereScale + 1f);
187                 UpdateBoundaryRing();
188             }
189         }
190     }

```

```

176         }
177     else
178     {
179         startTransform = false;
180     }
181 }
182 }
183
184 void CheckDistances()
185 {
186     float tempMax = minScale/4;
187     Vector3 centerCoord = new Vector3(transform.position.x, 0, transform
188         .position.z);
189
190     nodeList.RemoveAll(i => i == null);
191
192     foreach (GameObject i in nodeList)
193     {
194         Vector3 tempCoord = new Vector3(i.transform.position.x, 0, i.
195             transform.position.z);
196         float tempDist = Vector3.Distance(centerCoord, tempCoord);
197         if (tempDist > tempMax)
198         {
199             tempMax = tempDist;
200         }
201     }
202     maxDistance = tempMax;
203 }
204
205 void ChangeTargetScale(float target)
206 {
207     targetScale = target;
208     startTransform = true;
209 }
210
211 void UpdateBoundaryRing()
212 {
213     boundaryRing.GetComponent<GeometryTorus>().SetParameters(
214         centralPlatform.transform.localScale.x / 2 + ringWidth / 2,
215         ringWidth, 64, 4, 0.5f, true);
216     boundaryRing.transform.localScale = new Vector3(1, ringHeight /
217         ringWidth, 1);
218
219     for (int i = 0; i < stepList.Count; i++)
220     {
221         stepList[i].GetComponent<GeometryTorus>().SetParameters(
222             centralPlatform.transform.localScale.x / 2 - (i * stepWidth
223             * 1.3f) - ringWidth / 2, stepWidth, 64, 4, 0.5f, true);
224     }
225 }

```

```

218         stepList[i].transform.localScale = new Vector3(1, (1 - (float)(i
219             + 1) / (numSteps + 1)) * ringHeight / stepWidth, 1);
220     }
221 }
222 void CreateBoundaryRing()
223 {
224     boundaryRing = GameObject.Instantiate(boundaryRingPrefab);
225     boundaryRing.GetComponent<GeometryTorus>().SetParameters((
226         centralPlatform.transform.localScale.x / 2) + (ringWidth / 2),
227         ringWidth, 64, 4, 0.5f, true);
228     boundaryRing.transform.localScale = new Vector3(1, ringHeight /
229         ringWidth, 1);
230     boundaryRing.transform.position = this.transform.position;
231     boundaryRing.transform.SetParent(this.transform);
232
233     for (int i = 0; i < numSteps; i++)
234     {
235         GameObject stepObject = GameObject.Instantiate(
236             boundaryRingPrefab);
237         stepObject.GetComponent<GeometryTorus>().SetParameters(
238             centralPlatform.transform.localScale.x / 2 - (i * stepWidth
239                 * 1.3f) - ringWidth / 2, stepWidth, 64, 4, 0.5f, true);
240         stepObject.transform.localScale = new Vector3(1, (1 - (float)(i
241             + 1) / (numSteps + 1)) * ringHeight / stepWidth, 1);
242         stepObject.transform.position = this.transform.position;
243         stepObject.transform.SetParent(this.transform);
244
245         stepList.Add(stepObject);
246     }
247 }
248
249 public void InstantScale()
250 {
251     CheckDistances();
252
253     var platformRadius = (maxDistance + distanceMargin + stepWidth *
254         numSteps);
255
256     if (platformRadius * 2 > minScale && Mathf.Abs(platformRadius * 2 -
257         centralPlatform.transform.localScale.x) > 1 && platformRadius <
258         80)
259     {
260         ChangeTargetScale(Mathf.Round((maxDistance + distanceMargin +
261             stepWidth * numSteps) * 2));
262     }
263
264     centralPlatform.transform.localScale = new Vector3(targetScale, 0.3f
265         , targetScale);

```

```

255     float sphereScale = centralPlatform.transform.localScale.x + 2 *
256         ringWidth + 2;
257     boundarySphere.transform.localScale = new Vector3(sphereScale + 1f,
258             sphereScale + 1f, sphereScale + 1f);
259
260     UpdateBoundaryRing();
261
262     public void GetAllNodes()
263     {
264         nodeList.Clear();
265         nodeList = GameObject.FindGameObjectsWithTag("NodeBlock").ToList();
266         Debug.Log(nodeList.Count);
267     }
268
269     public void NodeListAdd(GameObject node)
270     {
271         nodeList.Add(node);
272     }
273
274     public void NodeListRemove(GameObject node)
275     {
276         nodeList.Remove(node);
277     }
278
279 }
```

4.3 Reddit Central Sphere

Script controlling the collision events for the sphere area to determine if player is still in center area.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class RedCenterSphereCollision : MonoBehaviour
6 {
7     // Start is called before the first frame update
8     void Start()
9     {
10 }
```

```

12
13     // Update is called once per frame
14     void Update()
15     {
16
17     }
18 }
```

4.4 Reddit Cube Spawner

Script for the pedestal that spawns the node-block.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Realtime;
5 using Photon.Pun;
6
7 public class RedCubeSpawner : MonoBehaviour
8 {
9     private bool touchingNode;
10    public RedCentralSpaceMonobehaviour centralControl;
11    public ColorChanger colorChanger;
12    private List<Collision> collidingNodes = new List<Collision>();
13    private List<GameObject> nodeBlockList = new List<GameObject>();
14    // Start is called before the first frame update
15    void Start()
16    {
17
18    }
19
20    // Update is called once per frame
21    void Update()
22    {
23
24    }
25
26    public void SpawnCube()
27    {
28        if (!IsTouchingNode())
29        {
30            object[] id = new object[1];
31            id[0] = Random.Range(1, 255);
32        }
33    }
34 }
```

```

33         GameObject cube = PhotonNetwork.Instantiate("Reddit/NodeBlock",
34             transform.position + new Vector3(0, 1, 0), transform.
35             rotation, 0, id);
36         nodeBlockList.Add(cube);
37         centralControl.NodeListAdd(cube);
38     }
39
40     void OnCollisionEnter(Collision collision)
41     {
42         if(collision.gameObject.CompareTag("NodeBlock"))
43         {
44             collidingNodes.Add(collision);
45             colorChanger.SetColor(1);
46         }
47
48         void OnCollisionExit(Collision collision)
49         {
50             if(collision.gameObject.CompareTag("NodeBlock"))
51             {
52                 collidingNodes.Remove(collision);
53                 if(!IsTouchingNode())
54                 {
55                     colorChanger.SetColor(0);
56                 }
57             }
58         }
59
60         public bool IsTouchingNode()
61         {
62             if(collidingNodes.Count > 0)
63             {
64                 return true;
65             }
66             else
67             {
68                 return false;
69             }
70         }
71
72         public void HoverIn()
73         {
74             if(!IsTouchingNode())
75             {
76                 colorChangerSetColor(2);
77             }
78         }
79
80         public void HoverOut()

```

```

81     {
82         if(!IsTouchingNode())
83         {
84             colorChanger.SetColor(0);
85         }
86     }
87
88
89 }
```

4.5 Reddit Node Block

Script controlling the node block - just simple sync functions and destroy if it falls off the map.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Pun;
5 using Photon.Realtime;
6 using Photon.Voice.Unity;
7
8 public class RedNodeBlock : MonoBehaviour, IPunInstantiateMagicCallback
9 {
10     public int voiceRoomID = 0;
11     public RedNodeSphere sphere;
12
13
14     // Start is called before the first frame update
15     void Start()
16     {
17
18     }
19
20     public void OnPhotonInstantiate(PhotonMessageInfo info)
21     {
22         object[] data = info.photonView.InstantiationData;
23         if (data != null)
24         {
25             voiceRoomID = (int)data[0];
26             sphere.voiceRoomID = voiceRoomID;
27         }
28         else
29         {
```

```

30         voiceRoomID = this.GetComponent<PhotonView>().ViewID;
31         sphere.voiceRoomID = voiceRoomID;
32     }
33 }
34 }
35
36 // Update is called once per frame
37 void Update()
38 {
39     if(transform.position.y <= -100)
40     {
41         Destroy();
42     }
43 }
44
45 void ConnectToRoom()
46 {
47 }
48
49
50 public void Destroy()
51 {
52
53     PhotonNetwork.Destroy(this.GetComponent<PhotonView>());
54 }
55 }
```

4.6 Reddit Node Sphere

Handles collision events for the projected sphere around each node block.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using Photon.Pun;
5 using Photon.Realtime;
6 using Photon.Voice.Unity;
7 using Photon.Voice.PUN;
8
9 public class RedNodeSphere : MonoBehaviour
10 {
11     public int voiceRoomID = 0;
12     public PhotonVoiceNetwork voiceNetwork;
```

```
13 public Recorder recorder;
14 public PlayerController playerController;
15 private List<GameObject> collidingPlayers = new List<GameObject>();
16
17 // Start is called before the first frame update
18 void Start()
19 {
20     playerController = GameObject.Find("XR Rig").GetComponent<
21         PlayerController>();
22     voiceNetwork = GameObject.Find("VoiceManager").GetComponent<
23         PhotonVoiceNetwork>();
24     recorder = GameObject.Find("VoiceManager").GetComponent<Recorder>();
25 }
26
27 // Update is called once per frame
28 void Update()
29 {
30 }
31
32 void OnTriggerEnter(Collider collider)
33 {
34     if(collider.gameObject.CompareTag("Player"))
35     {
36         playerController.numVoiceRooms += 1;
37
38         byte[] room = new byte[1];
39         room[0] = (byte)voiceRoomID;
40         voiceNetwork.Client.OpChangeGroups(null, room);
41         recorder.InterestGroup = room[0];
42
43         Debug.Log("Joined group " + room[0]);
44     }
45 }
46
47 void OnTriggerExit(Collider collider)
48 {
49     if(collider.gameObject.CompareTag("Player"))
50     {
51         playerController.numVoiceRooms -= 1;
52
53         byte[] room = new byte[1];
54         room[0] = (byte)voiceRoomID;
55         voiceNetwork.Client.OpChangeGroups(room, null);
56
57         if(playerController.numVoiceRooms == 0 && recorder.InterestGroup
58             != 0)
59         {
60             recorder.InterestGroup = 0;
61             Debug.Log("Recorder joined back group 0.");
62         }
63     }
64 }
```

```

60             Debug.Log("Disconnected from group " + room[0]);
61         }
62     }
63 }

```

4.7 Subreddit Star

Script to handle the behaviour of individual subreddit links.

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using TMPro;
5  using Photon.Pun;
6  using UnityEngine.XR.Interaction.Toolkit;
7
8  public class SubredditStarMonobehaviour : MonoBehaviour
9  {
10     public GameObject titleMesh;
11     private string subreddit;
12     //private RedditController redditController;
13     private NetworkManager networkManager;
14
15     // Start is called before the first frame update
16     void Start()
17     {
18         //InitialiseStar("");
19         //redditController = GameObject.Find("RedditManager").GetComponent<
20             //RedditController>();
21         networkManager = GameObject.Find("NetworkManager").GetComponent<
22             //NetworkManager>();
23         HideTitle();
24     }
25
26     public void InitialiseStar(string subredditName)
27     {
28         subreddit = subredditName;
29         //Debug.Log("subreddit = " + subreddit);
30         titleMesh.GetComponent<TextMeshPro>().text = string.Concat("r/" ,
31             subreddit);
32     }
33
34     public void ChangeSubreddit()
35     {

```

```

33         //networkManager.rig.GetComponent<XRRig>().MatchRigUpRigForward(new
34             Vector3(0, 3, 0), new Vector3(1, 0, 0));
35         networkManager.SetPlayerSubreddit(subreddit);
36     }
37
38     public void ShowTitle()
39     {
40         //titleMesh.SetActive(true);
41         titleMesh.GetComponent<TextMeshPro>().fontSize = 15;
42     }
43
44     public void HideTitle()
45     {
46         //titleMesh.SetActive(false);
47         titleMesh.GetComponent<TextMeshPro>().fontSize = 6;
48     }
49
50     // Update is called once per frame
51     void Update()
52     {
53         titleMesh.transform.rotation = Camera.main.transform.rotation;
54     }
55 }
```

4.8 Reddit Info Panel

Handles the info panel boards

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using TMPro;
5 using LogicUI.FancyTextRendering;
6 using NaughtyAttributes;
7
8 public class RedInfoPanelMonobehaviour : MonoBehaviour
9 {
10
11     public GameObject mainText;
12     private Camera XRcamera;
13     public Canvas cameraCanvas;
14
15     // Start is called before the first frame update
16     void Start()
```

```

17     {
18         XRcamera = GameObject.Find("VR Camera Controller").GetComponent<
19             Camera>();
20         cameraCanvas.GetComponent<Canvas>().worldCamera = XRcamera;
21         Invoke("FitContents", 0.1f);
22     }
23
24     public void Initialise()
25     {
26         Invoke("FitContents", 0.1f);
27     }
28
29     // Update is called once per frame
30     void Update()
31     {
32     }
33
34     public void SetText(string str)
35     {
36         mainText.GetComponent<MarkdownRenderer>().Source = str;
37         Invoke("FitContents", 0.1f);
38     }
39
40     void FitContents()
41     {
42         RectTransform textRect = mainText.GetComponent<RectTransform>();
43
44         textRect.localPosition = new Vector3(0, - textRect.sizeDelta.y / 2,
45             0);
46     }
47 }
```

4.9 Geometry Torus

Script that generates a torus-shape dynamically at runtime, because otherwise scaling a pre-made torus would be uniform and not give the expected results. Used for the giant rings as well as the cylindrical boundaries of the center space.

```
1 /**

```

```

2  * Based on a script by Steffen (http://forum.unity3d.com/threads/torus-in-unity.8487/) (in $primitives_966_104.zip, originally named "Primitives.cs")
3  *
4  * Editted by Michael Zoller on December 6, 2015.
5  * It was shortened by about 30 lines (and possibly sped up by a factor of 2
6  * ) by consolidating math & loops and removing intermediate Collections.
7  */
8
9 [RequireComponent(typeof(MeshFilter), typeof(MeshRenderer))]
10 public class GeometryTorus : MonoBehaviour
11 {
12
13     public float segmentRadius = 1f;
14     public float tubeRadius = 0.1f;
15     public int numSegments = 32;
16     public int numTubes = 12;
17     public float tubeRotation = 0.0f;
18     public bool hardEdges = false;
19
20     void Start()
21     {
22         RefreshTorus();
23     }
24
25     public void SetParameters(float circleRadius, float thickness, int
26         circleResolution, int tubeResolution, float tubeRotate, bool
27         hardShading)
28     {
29         segmentRadius = circleRadius;
30         tubeRadius = thickness;
31         numSegments = circleResolution;
32         numTubes = tubeResolution;
33         tubeRotation = tubeRotate;
34         hardEdges = hardShading;
35
36         RefreshTorus();
37     }
38
39     public void RefreshTorus()
40     {
41         // Total vertices
42         int totalVertices = numSegments * numTubes;
43
44         // Total primitives
45         int totalPrimitives = totalVertices * 2;
46
47         // Total indices
48         int totalIndices = totalPrimitives * 3;

```

```

47
48     // Init the mesh
49     Mesh mesh = new Mesh();
50
51     // Init the vertex and triangle arrays
52     Vector3[] vertices = new Vector3[totalVertices];
53     int[] triangleIndices = new int[totalIndices];
54
55     // Calculate size of a segment and a tube
56     float segmentSize = 2 * Mathf.PI / (float)numSegments;
57     float tubeSize = 2 * Mathf.PI / (float)numTubes;
58
59     // Create floats for our xyz coordinates
60     float x, y, z;
61
62     // Begin loop that fills in both arrays
63     for (int i = 0; i < numSegments; i++)
64     {
65         // Find next (or first) segment offset
66         int n = (i + 1) % numSegments; // changed segmentList.Count to
67                                         numSegments
68
69         // Find the current and next segments
70         int currentTubeOffset = i * numTubes;
71         int nextTubeOffset = n * numTubes;
72
73         for (int j = 0; j < numTubes; j++)
74         {
75             // Find next (or first) vertex offset
76             int m = (j + 1) % numTubes; // changed currentTube.Count to
77                                         numTubes
78
79             // Find the 4 vertices that make up a quad
80             int iv1 = currentTubeOffset + j;
81             int iv2 = currentTubeOffset + m;
82             int iv3 = nextTubeOffset + m;
83             int iv4 = nextTubeOffset + j;
84
85             // Calculate X, Y, Z coordinates.
86             x = (segmentRadius + tubeRadius * Mathf.Cos((j +
87                 tubeRotation) * tubeSize)) * Mathf.Cos(i * segmentSize);
88             z = (segmentRadius + tubeRadius * Mathf.Cos((j +
89                 tubeRotation) * tubeSize)) * Mathf.Sin(i * segmentSize);
90             y = tubeRadius * Mathf.Sin((j + tubeRotation) * tubeSize);
91
92             // Add the vertex to the vertex array
93             vertices[iv1] = new Vector3(x, y, z);
94
95             // "Draw" the first triangle involving this vertex
96             triangleIndices[iv1 * 6] = iv1;

```

```

93         triangleIndices[iv1 * 6 + 1] = iv2;
94         triangleIndices[iv1 * 6 + 2] = iv3;
95         // Finish the quad
96         triangleIndices[iv1 * 6 + 3] = iv3;
97         triangleIndices[iv1 * 6 + 4] = iv4;
98         triangleIndices[iv1 * 6 + 5] = iv1;
99     }
100 }
101 mesh.vertices = vertices;
102 mesh.triangles = triangleIndices;
103
104 mesh.RecalculateBounds();
105 mesh.RecalculateNormals(); // added on suggestion of Eric5h5 &
106 //joaeba in the forum thread
107
108 //HARD SHADING
109 if(hardEdges)
110 {
111     Vector3[] oldVerts = mesh.vertices;
112     int[] triangles = mesh.triangles;
113     Vector3[] newVertices = new Vector3[triangles.Length];
114     for (int i = 0; i < triangles.Length; i++)
115     {
116         newVertices[i] = oldVerts[triangles[i]];
117         triangles[i] = i;
118     }
119     mesh.vertices = newVertices;
120     mesh.triangles = triangles;
121     mesh.RecalculateBounds();
122     mesh.RecalculateNormals();
123 }
124
125 mesh.Optimize();
126 MeshFilter mFilter = GetComponent<MeshFilter>(); // tweaked to
127 // Generic
128 mFilter.mesh = mesh;
129 MeshCollider mCollider = GetComponent<MeshCollider>();
130 if(mCollider != null)
131 {
132     mCollider.sharedMesh = mesh;
133 }
134 }
```

5 Reddit Post Scripts

Sub-scripts that handle various aspects of the reddit object behaviours such as posts, comments, scrolling, etc.

5.1 Reddit Post Controller

Script handling all the procedural generation of post space geometry, as well as interactions with interface and other stuff.

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using System.Net;
4  using System.IO;
5  using UnityEngine;
6  using UnityEngine.Networking;
7  using Reddit;
8  using Reddit.Controllers;
9  using TMPro;
10 using Vuplex.WebView;
11 using System.Globalization;
12
13 public class RedPostMonobehaviour : MonoBehaviour
14 {
15
16     //REDDIT POST DATA
17     private Post post;
18     private int numUpvotes;
19     private string postText;
20     private string postURL;
21     private string postTitle;
22     private string postSubreddit;
23     private Comments postComments;
24     private WebClient webClient;
25
26
27     //GEOMETRY
28     public GameObject platformGeometry;
29     public GameObject platformCenter;
30     public GameObject platformLeft;
31     public GameObject platformRight;
32     public GameObject platformSimple;
33     public GameObject platformWall;
```

```

34     public GameObject platformBase;
35     public GameObject platformAreaCollider;
36     public GameObject mainContent;
37     public GameObject mainContentText;
38     public GameObject mainTitle;
39     public GameObject mainTitleText;
40     public GameObject interfaceButtons;
41     public GameObject commentFrame;
42     public GameObject cylinderPlatform;
43     public GameObject externalThumbnail;
44     public GameObject externalTitle;
45     public GameObject externalSubreddit;
46     public GameObject externalUpvotes;
47
48     private List<GameObject> commentFramesList = new List<GameObject>();
49     private List<GameObject> cylinderPlatforms = new List<GameObject>();
50     //private GameObject textTemplateColumn;
51     //private GameObject textTemplateComment;
52     public GameObject imagePlane;
53     public GameObject webViewObject;
54     public GameObject webView;
55     public GameObject webViewFrame;
56     public GameObject webViewLoadingText;
57     public float frameTextureValue = 0f;
58     WebViewPrefab webViewPrefab;
59     private Texture thumbnailTexture;
60     private Texture imageTexture;
61     private Color platformColor;
62
63     //PARAMETERS
64     //private (int votesCeil, int minRadius, int maxRadius) upvoteScaling =
65     //    (20000, 6, 12);
66     private bool instantiateCommentFrames = true;
67     private bool loadComments = true;
68     private bool linkPost = false;
69     private int platformLength;
70     private float platformLengthUnits;
71     private float unitLength = 4f;
72     private float collisionHeight = 0f;
73
74     // Start is called before the first frame update
75     void Start()
76     {
77         //platformGeometry = transform.Find("Platform").gameObject;
78         //platformAreaCollider = transform.Find("Collider").GetComponent<
79             //Collider>();
80         //mainText = transform.Find("MainText").gameObject;
81         //imagePlane = transform.Find("ImagePlane").gameObject;
82         webViewPrefab = webView.GetComponent<WebViewPrefab>();

```

```

82     platformColor = LibColor.RandomiseHSV(new Color(0.5f, 0.5f, 0.7f, 1.
83                                         0f), 0.3f, 0.1f, 0.1f);
84     RefreshGeometry();
85     HideInterior();
86 }
87
88 // Update is called once per frame
89 void Update()
90 {
91     var video = post.Listing.Media;
92 }
93
94 public void InitialisePost(Post p, Comments c, int length, float
95                           upvoteScale)
96 {
97     //POST DATA
98     post = p;
99     numUpvotes = p.UpVotes;
100    postTitle = p.Title;
101    postSubreddit = p.Subreddit;
102    postComments = c;
103
104    //PLATFORM GEOMETRY
105    platformLength = length;
106    platformLengthUnits = platformLength * unitLength;
107    frameTextureValue = upvoteScale;
108
109    if (post.Listing.IsSelf)
110    {
111        postText = ((SelfPost)post).SelfText;
112        if (postText.Length == 0)
113        {
114            //postText = postTitle;
115        }
116    }
117    else
118    {
119        postText = postTitle;
120        postURL = ((LinkPost)post).URL;
121        //Debug.Log("Not self post!");
122        StartCoroutine(GetThumbnailTexture(p.Listing.Thumbnail));
123        StartCoroutine(GetImageTexture(postURL));
124        if (!post.Listing.IsRedditMediaDomain && postURL != null)
125        {
126            //Debug.Log(postURL);
127            linkPost = true;
128        }
129    }

```

```

130
131     RefreshGeometry();
132     //RefreshComments();
133
134 }
135
136 void OnTriggerEnter(Collider collider)
137 {
138     //Debug.Log("Collision");
139     if (collider.gameObject.tag == "Player")
140     {
141         ShowInterior();
142         Debug.Log("Player entered post - refreshing comments...");
143         RefreshComments();
144         if (linkPost) InitialiseWebView(postURL);
145     }
146 }
147
148 void OnTriggerExit(Collider collider)
149 {
150     //Debug.Log("Collision");
151     if (collider.gameObject.tag == "Player")
152     {
153         HideInterior(true);
154     }
155 }
156
157 private void RefreshGeometry()
158 {
159     //Calculate size of platform
160     //platformRadius = upvoteScaling.minRadius + Mathf.Clamp((numUpvotes
161     //    / upvoteScaling.votesCeil), 0, 1) * (upvoteScaling.maxRadius -
162     upvoteScaling.minRadius);
163
164     //Platform geometry
165     platformCenter.transform.localScale = new Vector3(
166         platformLengthUnits / 2, 0.5f, 9); //old transform without frame
167     //prefab
168     platformLeft.transform.localScale = new Vector3(platformLengthUnits,
169         1, 4);
170     platformRight.transform.localScale = new Vector3(platformLengthUnits
171         , 1, 4);
172     platformSimple.transform.localScale = new Vector3(
173         platformLengthUnits, 18, 0.5f);
174
175     platformGeometry.GetComponent<MeshRenderer>().material.color =
176         platformColor;
177
178     //Platform Text
179     //var imagePos = new Vector3(platformLengthUnits / 3, 0, 0);

```

```

172     //var textPos = new Vector3(platformLengthUnits / 3 + 0.2f, 3.5f, 0)
173     ;
174     var rot = Quaternion.identity;
175     //imagePlane.transform.localPosition = imagePos;
176     //mainText.transform.localPosition = textPos;
177
178     mainContentText.GetComponent<TextMeshProUGUI>().text = postText;
179     mainContent.GetComponent<RedTextFrameMonobehaviour>().Initialise();
180     mainTitleText.GetComponent<TextMeshProUGUI>().text = postTitle;
181     mainTitle.GetComponent<RedTextFrameMonobehaviour>().Initialise();
182
183     externalTitle.GetComponent<TextMeshPro>().text = postTitle;
184     externalSubreddit.GetComponent<TextMeshPro>().text = numUpvotes.
185         ToString() + " | r/" + postSubreddit;
186
187     externalTitle.GetComponent<RectTransform>().sizeDelta = new Vector2(
188         platformLengthUnits - 3, 4);
189     externalSubreddit.GetComponent<RectTransform>().sizeDelta = new
190         Vector2(platformLengthUnits - 3, 4);
191     externalUpvotes.GetComponent<RectTransform>().sizeDelta = new Vector
192         2(platformLengthUnits - 3, 4);
193
194     MaterialManager matManager = GameObject.Find("MaterialManager").
195         GetComponent<MaterialManager>();
196     Color frameColor = Color.HSVToRGB(0.1f, 0.7f, 0.8f);
197     Material frameMaterial = matManager.InstantiateMaterialWithEmission(
198         string.Concat("FrameMaterial", frameTextureValue), frameColor, 0
199         .5f + frameTextureValue * 2);
200     platformSimple.GetComponent<FrameDynamic>().ReplaceBaseMaterial(
201         frameMaterial);
202     platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial();
203
204     if (post.Listing.IsSelf)
205     {
206         //Resize title to be larger in absence of thumbnail
207         externalTitle.GetComponent<RectTransform>().sizeDelta = new
208             Vector2(platformLengthUnits - 3, 10);
209         externalTitle.transform.localPosition = new Vector3(0, 1.3f, 1.8
210             f);
211     }
212
213     if (instantiateCommentFrames)
214     {
215         foreach (GameObject i in commentFramesList)
216         {
217             GameObject.Destroy(i);
218         }
219
220         for (int i = 0; i < platformLength; i++)
221         {

```

```

211         float xpos = (platformLengthUnits / 2) - (i * unitLength) -
212             (unitLength / 2);
213
214         GameObject newComment = Instantiate(commentFrame, transform)
215             ;
216         newComment.transform.localPosition = new Vector3(xpos, 1f, -
217             (18f / 2 + 3));
218         newComment.transform.rotation *= Quaternion.Euler(new Vector
219             3(0, 180, 0));
220         commentFramesList.Add(newComment);
221
222         //GameObject newPlatform = Instantiate(cylinderPlatform,
223             transform);
224         //newPlatform.transform.localPosition = new Vector3(xpos, 0.
225             6f, -5.5f);
226         //cylinderPlatforms.Add(newPlatform);
227
228         newComment = Instantiate(commentFrame, transform);
229         newComment.transform.localPosition = new Vector3(xpos, 1f, 1
230             8f / 2 + 3);
231         //newComment.transform.rotation *= Quaternion.Euler(new
232             Vector3(0, 180, 0));
233         commentFramesList.Add(newComment);
234
235         //newPlatform = Instantiate(cylinderPlatform, transform);
236         //newPlatform.transform.localPosition = new Vector3(xpos, 0.
237             6f, 5.5f);
238         //cylinderPlatforms.Add(newPlatform);
239     }
240
241     instantiateCommentFrames = false;
242 }
243
244     RefreshCollisionHeight(20f);
245
246     Invoke("RetrieveVotes", 1.0f);
247     RefreshImage();
248     RefreshThumbnail();
249 }
250
251     private void ShowInterior()
252 {
253     platformSimple.SetActive(false);
254     externalThumbnail.SetActive(false);
255     externalTitle.SetActive(false);
256     externalSubreddit.SetActive(false);
257     externalUpvotes.SetActive(false);

```

```

252     platformCenter.SetActive(true);
253     platformLeft.SetActive(true);
254     platformRight.SetActive(true);
255     platformWall.SetActive(true);
256     platformBase.SetActive(true);
257
258     if (linkPost) webViewObject.SetActive(true);
259     else if (imageTexture != null) imagePlane.SetActive(true);
260     else if (postText != "") mainContent.SetActive(true);
261
262     mainTitle.SetActive(true);
263     interfaceButtons.SetActive(true);
264
265     foreach (GameObject i in commentFramesList)
266     {
267         i.SetActive(true);
268     }
269     foreach (GameObject i in cylinderPlatforms)
270     {
271         i.SetActive(true);
272     }
273 }
274
275 private void HideInterior(bool hideComments = true)
276 {
277     platformSimple.SetActive(true);
278     externalThumbnail.SetActive(true);
279     externalTitle.SetActive(true);
280     externalSubreddit.SetActive(true);
281     externalUpvotes.SetActive(true);
282
283     platformCenter.SetActive(false);
284     platformLeft.SetActive(false);
285     platformRight.SetActive(false);
286     platformWall.SetActive(false);
287     platformBase.SetActive(false);
288
289     imagePlane.SetActive(false);
290     webViewObject.SetActive(false);
291     mainContent.SetActive(false);
292     mainTitle.SetActive(false);
293     interfaceButtons.SetActive(false);
294
295     if (hideComments)
296     {
297         foreach (GameObject i in commentFramesList)
298         {
299             i.SetActive(false);
300         }
301     }

```

```

302         foreach (GameObject i in cylinderPlatforms)
303     {
304         i.SetActive(false);
305     }
306 }
307
308 private void RefreshComments()
309 {
310     if (!loadComments) return;
311
312     int numComments = platformLength * 2;
313     var topComments = RetrieveComments(numComments);
314
315     for (int c = 0; c < commentFramesList.Count; c++)
316     {
317         if (c < topComments.Count)
318         {
319             commentFramesList[c].GetComponent<
320                 RedCommentFrameMonobehaviour>().InitialiseComments(
321                 topComments[c]);
322         }
323         else
324         {
325             //commentFramesList[c].GetComponent<
326                 RedCommentFrameMonobehaviour>().Deactivate();
327             commentFramesList[c].SetActive(false);
328         }
329     }
330
331     loadComments = false;
332
333     RefreshGeometry();
334 }
335
336 public void DeepRefreshAllComments()
337 {
338     if (!loadComments)
339     {
340         for (int c = 0; c < commentFramesList.Count; c++)
341         {
342             if (commentFramesList[c].activeSelf)
343             {
344                 commentFramesList[c].GetComponent<
345                     RedCommentFrameMonobehaviour>().DeepRefreshComments(
346                     );
347             }
348         }
349     }
350 }

```

```

347     private async void InitialiseWebView(string url)
348     {
349         webViewPrefab = webView.GetComponent<WebViewPrefab>();
350
351         //Set initial url of webview and initialise
352         webViewPrefab.InitialUrl = url;
353         webViewPrefab.Init();
354
355         await webViewPrefab.WaitUntilInitialized();
356         //bool initialLoad = true;
357         //Wait until webpage is fully loaded, then check height of page and
            resize webview window accordingly.
358         webViewPrefab.WebView.LoadProgressChanged += async (sender,
            EventArgs) =>
359     {
360         //Resize when finished
361         if (EventArgs.Type == ProgressChangeType.Finished)
362         {
363             var heightString = await webViewPrefab.WebView.
                ExecuteJavaScript("document.documentElement.scrollHeight
                    ");
364             var heightInPixels = float.Parse(heightString, CultureInfo.
                InvariantCulture);
365             float heightInUnityUnits;
366             bool tooLong = false;
367
368             if (webViewPrefab.WebView.Size.x * webViewPrefab.WebView.
                Resolution * heightInPixels > 7000000)
369             {
370                 //Throw error if webpage too large to load
371                 webViewLoadingText.GetComponent<TextMeshPro>().text =
                    "WEBPAGE TOO LONG";
372                 heightInPixels = 7000000 / (webViewPrefab.WebView.Size.x
                    * webViewPrefab.WebView.Resolution);
373                 tooLong = true;
374             }
375
376             ResizeWebView(heightInPixels, out heightInUnityUnits);
377
378             //Hide scrollbar and loading text
379             await webViewPrefab.WebView.ExecuteJavaScript("document.body
                .style.overflow = 'hidden';");
380             if (!tooLong)
381             {
382                 webViewLoadingText.GetComponent<TextMeshPro>().text =
                    "";
383                 webViewLoadingText.gameObject.SetActive(false);
384             }

```

```

386         webViewLoadingText.transform.localPosition = new Vector3(0,
387             -heightInUnityUnits, 0);
388     }
389 }
390 }
391 }
392
393 private void ResizeWebView(float heightInPixels, out float
394     heightInUnityUnits)
395 {
396     //Resize webview window to accommodate full webpage height
397     Debug.Log("Webpage Height: " + heightInPixels);
398     var existingWidth = webViewPrefab.WebView.Size.x;
399     heightInUnityUnits = heightInPixels / webViewPrefab.WebView.
400         Resolution;
401
402     webViewPrefab.Resize(existingWidth, heightInUnityUnits);
403
404     //Resize backing frame accordingly
405     webViewFrame.transform.localScale = new Vector3(existingWidth + 0.2f
406         , heightInUnityUnits + 0.2f, 0.09f);
407     webViewFrame.transform.localPosition = new Vector3(0.051f, -(

408         heightInUnityUnits / 2 - 0.5f), 0);
409
410     RefreshCollisionHeight(heightInUnityUnits);
411 }
412
413 IEnumerator GetImageTexture(string url)
414 {
415     UnityWebRequest www = UnityWebRequestTexture.GetTexture(url);
416     yield return www.SendWebRequest();
417
418     if (www.isNetworkError || www.isHttpError)
419     {
420         //Debug.Log("Image retrieval error");
421     }
422     else
423     {
424         imageTexture = DownloadHandlerTexture.GetContent(www);
425         //Debug.Log("Image downloaded");
426         Invoke("RefreshImage", 5.0f);
427     }
428 }
429
430 IEnumerator GetThumbnailTexture(string url)
431 {
432     UnityWebRequest www = UnityWebRequestTexture.GetTexture(url);

```

```

431     yield return www.SendWebRequest();
432
433     if (www.isNetworkError || www.isHttpError)
434     {
435         //Debug.Log("Thumbnail retrieval error");
436     }
437     else
438     {
439         thumbnailTexture = DownloadHandlerTexture.GetContent(www);
440         //Debug.Log("Thumbnail downloaded");
441         Invoke("RefreshThumbnail", 5.0f);
442     }
443
444 }
445
446 private void RefreshImage()
447 {
448     //SET IMAGE TEXTURE
449     if (imageTexture != null && imageTexture.height > 0 && imageTexture.
450         width > 0)
451     {
452         imagePlane.GetComponent<MeshRenderer>().enabled = true;
453         float imageRatio = (float)imageTexture.height / (float)
454             imageTexture.width;
455         imagePlane.transform.localScale = new Vector3(5f, 5f *
456             imageRatio, 0.1f);
457         imagePlane.transform.localPosition = new Vector3(7f, -(5f *
458             imageRatio - 5f) / 2 - 2, 0);
459         imagePlane.GetComponent<MeshRenderer>().material.SetTexture("_
460             BaseMap", imageTexture);
461         imagePlane.GetComponent<FrameDynamic>().RefreshFrame();
462         //Debug.Log("Setting Texture");
463     }
464     else
465     {
466         imagePlane.GetComponent<MeshRenderer>().enabled = false;
467     }
468 }
469
470 private void RefreshThumbnail()
471 {
472     //SET THUMBNAIL TEXTURE
473     if (thumbnailTexture != null && thumbnailTexture.height > 0 &&
474         thumbnailTexture.width > 0)
475     {
476         externalThumbnail.GetComponent<MeshRenderer>().enabled = true;
477         float thumbnailRatio = (float)thumbnailTexture.height / (float)
478             thumbnailTexture.width;
479         //if (thumbnailRatio == 0) thumbnailRatio = 1;
480         float platformRatio = 12f / platformLengthUnits;

```

```

474         if (thumbnailRatio >= platformRatio)
475     {
476         externalThumbnail.transform.localScale = new Vector3(10f, 10
477             f, 1);
478     }
479     else
480     {
481         externalThumbnail.transform.localScale = new Vector3(
482             platformLengthUnits - 2, (platformLengthUnits - 2) *
483             thumbnailRatio, 1);
484     }
485     externalThumbnail.GetComponent<MeshRenderer>().material.
486         SetTexture("_BaseMap", thumbnailTexture);
487     //Debug.Log("Setting Texture");
488
489     //Resize title to default with thumbnail
490     externalTitle.GetComponent<RectTransform>().sizeDelta = new
491         Vector2(platformLengthUnits - 3, 4);
492     externalTitle.transform.localPosition = new Vector3(0, 1.3f, 4.3
493             f);
494 }
495 else
496 {
497     externalThumbnail.GetComponent<MeshRenderer>().enabled = false;
498
499     //Resize title to be larger in absence of thumbnail
500     externalTitle.GetComponent<RectTransform>().sizeDelta = new
501         Vector2(platformLengthUnits - 3, 10);
502     externalTitle.transform.localPosition = new Vector3(0, 1.3f, 1.8
503             f);
504 }
505
506 private List<Comment> RetrieveComments(int numComments)
507 {
508     //var numComments = 4; //Mathf.CeilToInt(2 * Mathf.PI *
509     //platformRadius / 8);
510     var topComments = postComments.GetComments(sort: "top", 0, 0, true,
511         false, true, 2, 1000);
512     return topComments;
513 }
514
515 public void RefreshCollisionHeight(float yheight)
516 {
517     //only fire if yheight is greater than current height
518     if (yheight > collisionHeight)
519     {
520         yheight = Mathf.Ceil(yheight);
521         platformWall.transform.localScale = new Vector3(
522             platformLengthUnits + 4, 18 + 8, yheight + 20 + 4);

```

```

513         platformWall.transform.localPosition = new Vector3(0, -(yheight
514             + 20)/2), 0);
515         platformBase.transform.localScale = new Vector3(
516             platformLengthUnits + 4, 18 + 8, yheight + 20 + 4);
517         platformBase.transform.localPosition = new Vector3(0, -(yheight
518             + 20 + 2));
519         platformAreaCollider.transform.localScale = new Vector3(
520             platformLengthUnits + 4, yheight + 20 + 20, 25);
521         platformAreaCollider.transform.localPosition = new Vector3(0, -(yheight
522             + 20)/2), 0);
523
524         collisionHeight = yheight;
525     }
526
527 }
528
529 public void Vote(int vote)
530 {
531     if(vote == 1)
532     {
533         post.UpvoteAsync();
534         platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial(1);
535         imagePlane.GetComponent<FrameDynamic>().SetFrameMaterial(1);
536         mainTitle.GetComponent<RedTextFrameMonobehaviour>().
537             frameGeometry.GetComponent<ColorChanger>().SetColor(1);
538         mainContent.GetComponent<RedTextFrameMonobehaviour>().
539             frameGeometry.GetComponent<ColorChanger>().SetColor(1);
540         webViewFrame.GetComponent<ColorChanger>().SetColor(1);
541     }
542     else if(vote == -1)
543     {
544         post.DownvoteAsync();
545         platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial(2);
546         imagePlane.GetComponent<FrameDynamic>().SetFrameMaterial(2);
547         mainTitle.GetComponent<RedTextFrameMonobehaviour>().
548             frameGeometry.GetComponent<ColorChanger>().SetColor(2);
549         mainContent.GetComponent<RedTextFrameMonobehaviour>().
550             frameGeometry.GetComponent<ColorChanger>().SetColor(2);
551         webViewFrame.GetComponent<ColorChanger>().SetColor(2);
552     }
553     else if(vote == 0)
554     {
555         post.UnvoteAsync();
556         platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial(0);

```

```

553         imagePlane.GetComponent<FrameDynamic>().SetFrameMaterial(0);
554         mainTitle.GetComponent<RedTextFrameMonobehaviour>().
555             frameGeometry.GetComponent<ColorChanger>().SetColor(0);
556         mainContent.GetComponent<RedTextFrameMonobehaviour>().
557             frameGeometry.GetComponent<ColorChanger>().SetColor(0);
558         webViewFrame.GetComponent<ColorChanger>().SetColor(0);
559     }
560
561     public void RetrieveVotes()
562     {
563         if(post.IsUpvoted)
564         {
565             interfaceButtons.GetComponentInChildren<
566                 VoteControllerMonobehaviour>().SetVote(1);
567             platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial(1);
568             imagePlane.GetComponent<FrameDynamic>().SetFrameMaterial(1);
569             mainTitle.GetComponent<RedTextFrameMonobehaviour>().
570                 frameGeometry.GetComponent<ColorChanger>().SetColor(1);
571             mainContent.GetComponent<RedTextFrameMonobehaviour>().
572                 frameGeometry.GetComponent<ColorChanger>().SetColor(1);
573             webViewFrame.GetComponent<ColorChanger>().SetColor(1);
574         }
575         else if(post.IsDownvoted)
576         {
577             interfaceButtons.GetComponentInChildren<
578                 VoteControllerMonobehaviour>().SetVote(-1);
579             platformSimple.GetComponent<FrameDynamic>().SetFrameMaterial(2);
580             imagePlane.GetComponent<FrameDynamic>().SetFrameMaterial(2);
581             mainTitle.GetComponent<RedTextFrameMonobehaviour>().
582                 frameGeometry.GetComponent<ColorChanger>().SetColor(2);
583             mainContent.GetComponent<RedTextFrameMonobehaviour>().
584                 frameGeometry.GetComponent<ColorChanger>().SetColor(2);
585             webViewFrame.GetComponent<ColorChanger>().SetColor(2);
586         }
587     }
588 }

```

5.2 Reddit Comment Frame

Script handling all the procedural generation and calculation of comment frames, as well as associated interactions.

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using System.Linq;
4  using UnityEngine;
5  using Reddit;
6  using Reddit.Controllers;
7  using TMPro;
8  using UnityEngine.UI;
9  using NaughtyAttributes;
10 using LogicUI.FancyTextRendering;
11
12 public class RedCommentFrameMonobehaviour : MonoBehaviour
13 {
14
15     //PLAYER
16     private GameObject rig;
17     private MovementProvider playerMovementProvider;
18
19     //GAMEOBJECT / GEOMETRY
20     private Camera XRcamera;
21     public GameObject canvas;
22     public GameObject frameGeometry;
23     public GameObject textContainer;
24     public GameObject mainText;
25     public GameObject refreshButton;
26     public GameObject voteArrows;
27     public GameObject subTextPrefab;
28     public GameObject helperContainer;
29     public Material subCommentMaterial;
30     public GameObject verticalFX;
31     public int totalReplies = 0;
32     private RedPostMonobehaviour parentPostController;
33     private ScrollRect scrollRect;
34     private MaterialManager materialManager = null;
35     private List<string> subCommentStrings = new List<string>();
36     private List<int> subCommentDepth = new List<int>();
37     private List<GameObject> subCommentObjectList = new List<GameObject>();
38     private List<Comment> subCommentList = new List<Comment>();
39
40     //REDDIT COMMENT DATA
41     private Comment mainComment = null;
42     private List<Comment> directReplies;
43
44
45     //PARAMETERS
46     private float frameWidth = 2.4f;
47     private float frameMargin = 0.2f;
48     private float yMargin = 0.05f;
49
50

```

```

51     private bool initialised = false;
52     private bool deepRefresh = false;
53     private Vector2 previousVector2 = new Vector2(0, 0);
54
55
56     public void InitialiseComments(Comment c)
57     {
58         mainComment = c;
59
60         if (!initialised)
61         {
62             //directReplies = RetrieveChildComments(c);
63             RetrieveAllChildComments(c.replies, 1);
64             totalReplies = c.NumReplies.Value;
65
66             UpdateText();
67             RetrieveVote();
68             //FitContents();
69             Invoke("FitContents", 0.1f); //need to delay fitcontents so '
69             content fitter' has time to work
70             initialised = true;
71         }
72
73         parentPostController = transform.parent.GetComponent<
74             RedPostMonobehaviour>();
75     }
76
77     // Start is called before the first frame update
78     void Start()
79     {
80         XRcamera = GameObject.Find("VR Camera Controller").GetComponent<
80             Camera>();
81         rig = GameObject.Find("XR Rig");
82         playerMovementProvider = rig.GetComponent<MovementProvider>();
83         canvas.GetComponent<Canvas>().worldCamera = XRcamera;
84         scrollRect = textContainer.GetComponent<RectTransform>();
85
86         materialManager = GameObject.Find("MaterialManager").GetComponent<
86             MaterialManager>();
87
88         //canvas.SetActive(false);
89     }
90
91     // Update is called once per frame
92     void Update()
93     {
94
95     }
96
97     void UpdateText()

```

```

97     {
98         mainText.GetComponent<MarkdownRenderer>().Source = "";
99         mainText.GetComponent<MarkdownRenderer>().Source = mainComment.
100            UpVotes + " | u/" + mainComment.Author + "\n" + mainComment.Body
101            ;
102
103        for (int i = 0; i < subCommentStrings.Count; i++)
104        {
105            GameObject subComment = Instantiate(subTextPrefab, textContainer
106                .transform);
107            //subComment.GetComponent<TextMeshProUGUI>().text =
108            subCommentStrings[i];
109            subComment.GetComponent<RedCommentChildMonobehaviour>().
110                Initialise(subCommentList[i], subCommentStrings[i], this);
111
112            Color subCommentColor = Color.HSVToRGB(0.4f, 0.3f, 0.2f + 0.6f/
113                subCommentDepth[i]);
114            string subCommentColorName = string.Concat("CommentMaterial",
115                subCommentDepth[i]);
116            //Debug.Log(subCommentColorName + " " + subCommentColor + " " +
117                subCommentMaterial);
118
119            if(materialManager == null)
120            {
121                materialManager = GameObject.Find("MaterialManager").
122                    GetComponent<MaterialManager>();
123
124                Material subCommentBackgroundMaterial = materialManager.
125                    InstantiateMaterialWithColor(subCommentColorName,
126                        subCommentColor, subCommentMaterial);
127
128                subComment.transform.Find("ChildBackground").GetComponent<Image>
129                    ().material = subCommentBackgroundMaterial;
130
131                subCommentObjectList.Add(subComment);
132            }
133        }
134    }
135
136    void FitContents()
137    {
138        //canvas.SetActive(true);
139
140
141        var mainRectTransform = mainText.GetComponent<RectTransform>();
142        var mainHeight = mainRectTransform.sizeDelta[1];
143        mainRectTransform.anchoredPosition = new Vector3(0, -
144            mainRectTransform.sizeDelta[1] / 2, 0);
145
146
147        var yOffsetTotal = 0.0f;

```

```

134     for (int i = 0; i < subCommentList.Count; i++)
135     {
136         var subRectTransform = subCommentObjectList[i].GetComponent<
137             RectTransform>();
138         var subHeight = subRectTransform.sizeDelta[1];
139         subRectTransform.anchoredPosition = new Vector3(0.1f, -(mainHeight + (subHeight / 2) + yMargin + yOffsetTotal), 0);
140         yOffsetTotal += subHeight + yMargin;
141     }
142
143     var textContainerRect = textContainer.GetComponent<RectTransform>();
144     var columnHeight = mainHeight + yOffsetTotal + 0.1f;
145     textContainerRect.sizeDelta = new Vector2(frameWidth, columnHeight);
146     textContainerRect.anchoredPosition = new Vector3(0, -columnHeight /
147         2, 0);
148
149     frameGeometry.transform.localScale = new Vector3(frameWidth +
150         frameMargin, columnHeight + frameMargin, 0.1f);
151     frameGeometry.transform.localPosition = new Vector3(0, -columnHeight
152         / 2, 0.1f);
153
154     //Refresh total height
155     parentPostController.RefreshCollisionHeight(yOffsetTotal);
156 }
157
158 List<Comment> RetrieveChildComments(Comment comment)
159 {
160     return comment.Replies;
161 }
162
163 void RetrieveAllChildComments(IList<Comment> comments, int depth)
164 {
165     if (comments != null)
166     {
167         foreach (Comment comment in comments)
168         {
169             subCommentStrings.Add(comment.UpVotes + " | u/" + comment.
170                 Author + "\n" + string.Concat(Enumerable.Repeat<string>(
171                     ">", depth)) + " " + comment.Body);
172             subCommentList.Add(comment);
173             subCommentDepth.Add(depth);
174             //Debug.Log(comment.replies.Count);
175             if (comment != null) RetrieveAllChildComments(comment.replies
176                 , (depth + 1));
177         }
178     }
179 }
180
181 public void DeepRefreshComments()

```

```

176     {
177         if (!deepRefresh)
178         {
179             foreach(GameObject c in subCommentObjectList)
180             {
181                 GameObject.Destroy(c);
182             }
183             subCommentObjectList.Clear();
184             subCommentList.Clear();
185             subCommentDepth.Clear();
186             subCommentStrings.Clear();
187
188             deepRefresh = true;
189
190             RetrieveAllChildComments(mainComment.Comments.GetTop(limit:500),
191                                     1);
192             UpdateText();
193             Invoke("FitContents", 0.1f);
194
195             closeButton.GetComponent<RefreshCommentMonobehaviour>().
196             SetRefreshed(true);
197         }
198     }
199
200     public void Deactivate()
201     {
202         canvas.SetActive(false);
203     }
204
205     public void MovePlayer(Vector2 vector)
206     {
207         var velocity = scrollRect.velocity;
208         playerMovementProvider.VertMove(-velocity.y);
209     }
210
211     public void ResizeVerticalFX(float height)
212     {
213         verticalFX.transform.localScale = new Vector3(2f, height, 1);
214         verticalFX.transform.localPosition = new Vector3(0, -height/2, 0.1f)
215         ;
216     }
217
218     public bool Refreshed()
219     {
220         return deepRefresh;
221     }
222
223     public void Vote(int vote)
224     {

```

```

223     if(vote == 1)
224     {
225         mainComment.UpvoteAsync();
226         frameGeometry.GetComponent<ColorChanger>().SetColor(1);
227     }
228     else if(vote == -1)
229     {
230         mainComment.DownvoteAsync();
231         frameGeometry.GetComponent<ColorChanger>().SetColor(2);
232     }
233     else if(vote == 0)
234     {
235         mainComment.UnvoteAsync();
236         frameGeometry.GetComponent<ColorChanger>().SetColor(0);
237     }
238 }
239
240 public void RetrieveVote()
241 {
242     VoteControllerMonobehaviour voteController = voteArrows.GetComponent
243         <VoteControllerMonobehaviour>();
244     if(mainComment.IsUpvoted)
245     {
246         voteController.SetVote(1);
247         frameGeometry.GetComponent<ColorChanger>().SetColor(1);
248     }
249     else if(mainComment.IsDownvoted)
250     {
251         voteController.SetVote(-1);
252         frameGeometry.GetComponent<ColorChanger>().SetColor(2);
253     }
254 }
255
256 public int GetHiddenComments()
257 {
258     int numComments = subCommentList.Sum(x => x.NumReplies).Value;
259 }
260 }
```

5.3 Reddit Comment Child

Script attached to each child comment for finer control of attributes.

```
1 using System.Collections;
```

```

2  using System.Collections.Generic;
3  using UnityEngine;
4  using Reddit;
5  using Reddit.Controllers;
6  using TMPro;
7  using NaughtyAttributes;
8  using LogicUI.FancyTextRendering;
9
10 public class RedCommentChildMonobehaviour : MonoBehaviour
11 {
12     private RedCommentFrameMonobehaviour parentCommentController;
13     private Comment comment;
14     private string commentString;
15
16     //Geometry
17     public TextMeshProUGUI textMeshComponent;
18     public MarkdownRenderer markdownRendererComponent;
19     public VoteControllerMonobehaviour voteController;
20
21     // Start is called before the first frame update
22     void Start()
23     {
24
25     }
26
27     // Update is called once per frame
28     void Update()
29     {
30
31     }
32
33     public void Initialise(Comment c, string s, RedCommentFrameMonobehaviour
34         parent)
35     {
36         comment = c;
37         commentString = s;
38         parentCommentController = parent;
39
40         UpdateText(commentString);
41         RetrieveVote();
42     }
43
44     public void UpdateText(string s)
45     {
46         markdownRendererComponent.Source = s;
47     }
48
49     public void Vote(int vote)
50     {
51         if(vote == 1)

```

```

51     {
52         comment.UpvoteAsync();
53     }
54     else if(vote == -1)
55     {
56         comment.DownvoteAsync();
57     }
58     else if(vote == 0)
59     {
60         comment.UnvoteAsync();
61     }
62 }
63
64 public void RetrieveVote()
65 {
66     if(comment.IsUpvoted)
67     {
68         voteController.SetVote(1);
69     }
70     else if(comment.IsDownvoted)
71     {
72         voteController.SetVote(-1);
73     }
74 }
75 }
```

5.4 Vote Controller

Script attached to the upvote/downvote arrows that handle the API operation and toggle the arrow colors.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class VoteControllerMonobehaviour : MonoBehaviour
6 {
7     public RedPostMonobehaviour parentPostController;
8     public RedCommentChildMonobehaviour parentCommentChildController;
9     public RedCommentFrameMonobehaviour parentCommentController;
10    public GameObject arrowUpvote;
11    public GameObject arrowDownvote;
12    private int vote = 0;
13 }
```

```

14     // Start is called before the first frame update
15     void Start()
16     {
17
18     }
19
20     public void ToggleUpvote(bool live = true)
21     {
22         if(vote != 1)
23         {
24             vote = 1;
25             arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
26             arrowUpvote.GetComponent<ColorChanger>().SetColor(1);
27
28             if(parentPostController != null && live) parentPostController.
29                 Vote(1);
30             if(parentCommentChildController != null && live)
31                 parentCommentChildController.Vote(1);
32             if(parentCommentController != null && live)
33                 parentCommentController.Vote(1);
34         }
35         else
36         {
37             vote = 0;
38             arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
39             arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
40
41             if(parentPostController != null && live) parentPostController.
42                 Vote(0);
43             if(parentCommentChildController != null && live)
44                 parentCommentChildController.Vote(0);
45             if(parentCommentController != null && live)
46                 parentCommentController.Vote(0);
47         }
48     }
49
50     public void ToggleDownvote(bool live = true)
51     {
52         if(vote != -1)
53         {
54             vote = -1;
55             arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
56             arrowDownvote.GetComponent<ColorChanger>().SetColor(1);
57
58             if(parentPostController != null && live) parentPostController.
59                 Vote(-1);
60             if(parentCommentChildController != null && live)
61                 parentCommentChildController.Vote(-1);
62             if(parentCommentController != null && live)
63                 parentCommentController.Vote(-1);

```

```
55     }
56     else
57     {
58         vote = 0;
59         arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
60         arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
61
62         if(parentPostController != null && live) parentPostController.
63             Vote(0);
64         if(parentCommentChildController != null && live)
65             parentCommentChildController.Vote(0);
66         if(parentCommentController != null && live)
67             parentCommentController.Vote(0);
68     }
69 }
70
71 public void HoverUpvote()
72 {
73     //Debug.Log("hover upvote");
74     if(vote != 1)
75     {
76         arrowUpvote.GetComponent<ColorChanger>().SetColor(2);
77     }
78 }
79
80 public void HoverDownvote()
81 {
82     //Debug.Log("hover downvote");
83     if(vote != -1)
84     {
85         arrowDownvote.GetComponent<ColorChanger>().SetColor(2);
86     }
87 }
88
89 public void HoverOutUpvote()
90 {
91     if(vote == 1)
92     {
93         arrowUpvote.GetComponent<ColorChanger>().SetColor(1);
94     }
95     else
96     {
97         arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
98     }
99 }
100
101 public void HoverOutDownvote()
102 {
103     if(vote == -1)
104     {
```

```

102         arrowDownvote.GetComponent<ColorChanger>().SetColor(1);
103     }
104     else
105     {
106         arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
107     }
108 }
109
110 public void ResetColors()
111 {
112     if(vote == 0)
113     {
114         arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
115         arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
116     }
117     else if(vote == 1)
118     {
119         arrowUpvote.GetComponent<ColorChanger>().SetColor(1);
120         arrowDownvote.GetComponent<ColorChanger>().SetColor(0);
121     }
122     else if(vote == -1)
123     {
124         arrowUpvote.GetComponent<ColorChanger>().SetColor(0);
125         arrowDownvote.GetComponent<ColorChanger>().SetColor(1);
126     }
127 }
128
129 public void SetVote(int i)
130 {
131     vote = i;
132     Invoke("ResetColors", 0.1f);
133 }
134 }
```

5.5 Refresh Comments

Script attached to the refresh arrow that handles the refresh operation.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using TMPro;
5
6 public class RefreshCommentMonobehaviour : MonoBehaviour
7 {
```

```

8
9     public RedPostMonobehaviour parentPostController;
10    public RedCommentFrameMonobehaviour parentCommentFrameController;
11    public GameObject refreshText;
12    public GameObject refreshArrow;
13    private bool refreshed = false;
14
15
16    // Start is called before the first frame update
17    void Start()
18    {
19        if(parentCommentFrameController != null)
20        {
21            //refreshText.GetComponent<TextMeshPro>().text = "Load More
22            //Comments\n" + parentCommentFrameController.
23            GetHiddenComments();
24        }
25        else if(parentPostController != null)
26        {
27            //refreshText.GetComponent<TextMeshPro>().text = "Load All
28            //Comments";
29        }
30        refreshText.SetActive(false);
31    }
32
33    public void Refresh()
34    {
35        if(parentCommentFrameController != null)
36        {
37            RefreshComments();
38            SetRefreshed(true);
39        }
40        else if(parentPostController != null)
41        {
42            RefreshAllPostComments();
43            SetRefreshed(true);
44        }
45    }
46
47    public void RefreshComments()
48    {
49        if(!parentCommentFrameController.Refreshed())
50        {
51            parentCommentFrameController.DeepRefreshComments();
52        }
53    }
54    public void RefreshAllPostComments()
55    {
56        parentPostController.DeepRefreshAllComments();
57    }

```

```

55
56     public void SetRefreshed(bool x)
57     {
58         refreshed = x;
59         refreshArrow.GetComponent<ColorChanger>().SetColor(2);
60         refreshText.SetActive(false);
61         refreshArrow.SetActive(false);
62     }
63
64     public void HoverIn()
65     {
66         if (!refreshed)
67         {
68             refreshText.SetActive(true);
69             refreshArrow.GetComponent<ColorChanger>().SetColor(1);
70         }
71     }
72
73     public void HoverOut()
74     {
75         refreshText.SetActive(false);
76         if (!refreshed)
77         {
78             refreshArrow.GetComponent<ColorChanger>().SetColor(0);
79         }
80     }
81 }
```

6 General Support Scripts

Some general scripts and libraries to handle miscellaneous operations, mostly regarding procedural geometry.

6.1 Scroll Surface

Key script that handles the reverse-scrolling of the player - uses an invisible helper scroll rectangle that moves the player in the opposite direction when scrolled.

```

1  using System.Collections;
2  using System.Collections.Generic;
3  using System.Linq;
4  using UnityEngine;
5  using Reddit;
6  using Reddit.Controllers;
7  using TMPro;
8  using UnityEngine.UI;

9
10 public class RedScrollView : MonoBehaviour
11 {
12
13     //PLAYER
14     private GameObject rig;
15     private MovementProvider playerMovementProvider;
16
17     //GAMEOBJECT / GEOMETRY
18     private Camera XRcamera;
19     public GameObject cameraCanvas;
20     public GameObject scrollRectContainer;
21     public GameObject scaleReference = null;
22     private ScrollRect scrollRect;
23
24     //PARAMETERS
25     private float frameWidth = 2.4f;
26     private float frameMargin = 0.2f;
27     private float yMargin = 0.2f;
28
29
30     private Vector2 previousVector2 = new Vector2(0, 0);
31
32
33     // Start is called before the first frame update
34     void Start()
35     {
36         XRcamera = GameObject.Find("VR Camera Controller").GetComponent<
37             Camera>();
38         rig = GameObject.Find("XR Rig");
39         playerMovementProvider = rig.GetComponent<MovementProvider>();
40         cameraCanvas.GetComponent<Canvas>().worldCamera = XRcamera;
41         scrollRect = scrollRectContainer.GetComponent<ScrollRect>();
42     }
43
44     // Update is called once per frame
45     void Update()
46     {
47
48     }
49

```

```

50     public void MovePlayer(Vector2 vector)
51     {
52         var velocity = scrollRect.velocity;
53         if(scaleReference != null)
54         {
55             velocity = velocity * scaleReference.transform.localScale.y;
56         }
57         playerMovementProvider.VertMove(-velocity.y);
58     }
59
60     public void SnapPlayer()
61     {
62
63     }
64 }
```

6.2 Frame Dynamic

Creates the 3D frame around certain UI elements to make them less flat.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5
6 public class FrameDynamic : MonoBehaviour
7 {
8     public float frameThickness = 0.1f;
9     public float frameDepth = 0f;
10    public Material frameMaterial;
11    public List<Material> frameMaterials;
12    private List<GameObject> frameObjects;
13    private GameObject frameUpper;
14    private GameObject frameRight;
15    private GameObject frameLower;
16    private GameObject frameLeft;
17
18    // Start is called before the first frame update
19    void Start()
20    {
21        CreateFrame();
22    }
23
24    // Update is called once per frame
25    void Update()
```

```

26     {
27
28 }
29
30     void CreateFrame()
31 {
32     if (frameUpper != null || frameRight != null || frameLower != null
33         || frameLeft != null)
34     {
35         return;
36     }
37
38     frameMaterials.Insert(0, frameMaterial);
39
40     frameUpper = GameObject.CreatePrimitive(PrimitiveType.Cube);
41     frameRight = GameObject.CreatePrimitive(PrimitiveType.Cube);
42     frameLower = GameObject.CreatePrimitive(PrimitiveType.Cube);
43     frameLeft = GameObject.CreatePrimitive(PrimitiveType.Cube);
44
45     frameUpper.transform.SetParent(transform);
46     frameRight.transform.SetParent(transform);
47     frameLower.transform.SetParent(transform);
48     frameLeft.transform.SetParent(transform);
49
50     RefreshFrame();
51 }
52
53     public void RefreshFrame()
54 {
55     if (frameUpper == null || frameRight == null || frameLower == null
56         || frameLeft == null)
57     {
58         CreateFrame();
59     }
60
61     frameUpper.transform.localRotation = Quaternion.identity;
62     frameRight.transform.localRotation = Quaternion.identity;
63     frameLower.transform.localRotation = Quaternion.identity;
64     frameLeft.transform.localRotation = Quaternion.identity;
65
66     frameUpper.transform.localPosition = new Vector3(0, (transform.
       localScale.y / 2 + frameThickness / 2) / transform.localScale.y,
       0);
67     frameRight.transform.localPosition = new Vector3(- (transform.
       localScale.x / 2 + frameThickness / 2) / transform.localScale.x,
       0, 0);
68     frameLower.transform.localPosition = new Vector3(0, - (transform.
       localScale.y / 2 + frameThickness / 2) / transform.localScale.y,
       0);
69     frameLeft.transform.localPosition = new Vector3((transform.
       localScale.x / 2 + frameThickness / 2) / transform.localScale.x,
       0, 0);

```

```

        , 0, 0);

67
68     frameUpper.transform.localScale = new Vector3( (transform.localScale
       .x + 2 * frameThickness) / transform.localScale.x,
       frameThickness / transform.localScale.y, (transform.localScale.z
       + frameDepth) / transform.localScale.z);
69     frameRight.transform.localScale = new Vector3(frameThickness /
       transform.localScale.x, (transform.localScale.y) / transform.
       localScale.y, (transform.localScale.z + frameDepth) / transform.
       localScale.z);
70     frameLower.transform.localScale = new Vector3( (transform.localScale
       .x + 2 * frameThickness) / transform.localScale.x,
       frameThickness / transform.localScale.y, (transform.localScale.z
       + frameDepth) / transform.localScale.z);
71     frameLeft.transform.localScale = new Vector3(frameThickness /
       transform.localScale.x, (transform.localScale.y) / transform.
       localScale.y, (transform.localScale.z + frameDepth) / transform.
       localScale.z);
72
73     SetFrameMaterial(0);
74 }
75
76     public void SetFrameMaterial(int i = 0)
77     {
78         if (frameUpper == null || frameRight == null || frameLower == null
           || frameLeft == null)
79         {
80             CreateFrame();
81         }
82         frameUpper.GetComponent<MeshRenderer>().material = frameMaterials[i]
           ;
83         frameRight.GetComponent<MeshRenderer>().material = frameMaterials[i]
           ;
84         frameLower.GetComponent<MeshRenderer>().material = frameMaterials[i]
           ;
85         frameLeft.GetComponent<MeshRenderer>().material = frameMaterials[i];
86     }
87
88     public void ReplaceBaseMaterial(Material mat)
89     {
90         if (frameUpper == null || frameRight == null || frameLower == null
           || frameLeft == null)
91         {
92             CreateFrame();
93         }
94         frameMaterials[0] = mat;
95     }
96 }
```

6.3 Frame Dynamic Planes

Same as frame dynamic but for planar elements.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5
6 public class FrameDynamicPlanes : MonoBehaviour
7 {
8     public float frameDepth = 0f;
9     public Material frameMaterial;
10    private List<GameObject> frameObjects;
11    private GameObject frameUpper;
12    private GameObject frameRight;
13    private GameObject frameLower;
14    private GameObject frameLeft;
15
16    // Start is called before the first frame update
17    void Start()
18    {
19        CreateFrame();
20    }
21
22    // Update is called once per frame
23    void Update()
24    {
25
26    }
27
28    public void RefreshFrame()
29    {
30        if (frameUpper == null || frameRight == null || frameLower == null
31            || frameLeft == null)
32        {
33            return;
34        }
35        frameUpper.transform.localRotation = Quaternion.Euler(270, 90, 0);
36        frameRight.transform.localRotation = Quaternion.Euler(180, 90, 0);
37        frameLower.transform.localRotation = Quaternion.Euler(90, 90, 0);
38        frameLeft.transform.localRotation = Quaternion.Euler(0, 90, 0);
39
40        frameUpper.transform.localPosition = new Vector3(0, (transform.
localScale.y / 2) / transform.localScale.y, 0);
41        frameRight.transform.localPosition = new Vector3(- (transform.
localScale.x / 2) / transform.localScale.x, 0, 0);
```

```

41     frameLower.transform.localPosition = new Vector3(0, - (transform.
42         localScale.y / 2) / transform.localScale.y, 0);
43     frameLeft.transform.localPosition = new Vector3((transform.
44         localScale.x / 2) / transform.localScale.x, 0, 0);
45
46     frameUpper.transform.localScale = new Vector3( (transform.localScale
47         .x + frameDepth) / transform.localScale.x, 1, 1);
48     frameRight.transform.localScale = new Vector3( (transform.localScale
49         .x + frameDepth) / transform.localScale.x, 1, 1);
50     frameLower.transform.localScale = new Vector3( (transform.localScale
51         .x + frameDepth) / transform.localScale.x, 1, 1);
52     frameLeft.transform.localScale = new Vector3( (transform.localScale.
53         x + frameDepth) / transform.localScale.x, 1, 1);
54
55     frameUpper.GetComponent<MeshRenderer>().material = frameMaterial;
56     frameRight.GetComponent<MeshRenderer>().material = frameMaterial;
57     frameLower.GetComponent<MeshRenderer>().material = frameMaterial;
58     frameLeft.GetComponent<MeshRenderer>().material = frameMaterial;
59 }
60
61 void CreateFrame()
62 {
63     frameUpper = GameObject.CreatePrimitive(PrimitiveType.Quad);
64     frameRight = GameObject.CreatePrimitive(PrimitiveType.Quad);
65     frameLower = GameObject.CreatePrimitive(PrimitiveType.Quad);
66     frameLeft = GameObject.CreatePrimitive(PrimitiveType.Quad);
67
68     frameUpper.transform.SetParent(transform);
69     frameRight.transform.SetParent(transform);
70     frameLower.transform.SetParent(transform);
71     frameLeft.transform.SetParent(transform);
72
73     RefreshFrame();
74 }
75 }
```

6.4 Text Frame

Some stuff to handle text panels.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class RedTextFrameMonobehaviour : MonoBehaviour
```

```

6  {
7
8      public GameObject frameGeometry;
9      public GameObject textContainer;
10     public GameObject mainText;
11     public AlignEnum alignment;
12
13     //PARAMETERS
14     public float frameWidth = 4f;
15     public float frameMargin = 0.2f;
16
17     public enum AlignEnum
18     {
19         Top,
20         Bottom
21     };
22
23     // Start is called before the first frame update
24     void Start()
25     {
26
27     }
28
29     // Update is called once per frame
30     void Update()
31     {
32
33     }
34
35     public void Initialise()
36     {
37         Invoke("FitContents", 0.1f);
38     }
39
40     void FitContents()
41     {
42
43         var mainTextRect = mainText.GetComponent<RectTransform>();
44         var mainHeight = mainTextRect.sizeDelta[1];
45
46         var textContainerRect = textContainer.GetComponent<RectTransform>();
47         textContainerRect.sizeDelta = new Vector2(frameWidth, mainHeight);
48
49         frameGeometry.transform.localScale = new Vector3(frameWidth +
50             frameMargin, mainHeight + frameMargin, 0.1f);
51
52         if(alignment.ToString() == "Bottom")
53     {

```

```

54         textContainerRect.anchoredPosition = new Vector3(0, mainHeight /
55             2, 0);
56         mainTextRect.anchoredPosition = new Vector3(0, mainHeight / 2, 0
57             );
58         frameGeometry.transform.localPosition = new Vector3(0, - (1 -
59             mainHeight / 2), 0.051f);
60         //transform.localPosition += new Vector3(0, 2f, 0); //moves
61         //entire gameobject to correct position
62     }
63     else
64     {
65         textContainerRect.anchoredPosition = new Vector3(0, - mainHeight
66             / 2, 0);
67         mainTextRect.anchoredPosition = new Vector3(0, - mainHeight / 2,
68             0);
69         frameGeometry.transform.localPosition = new Vector3(0, (1 -
70             mainHeight / 2), 0.051f);
71     }
72
73
74
75     /*
76     var yOffsetTotal = 0.0f;
77     for (int i = 0; i < subCommentList.Count; i++)
78     {
79         var subRectTransform = subCommentList[i].GetComponent<
80             RectTransform>();
81         var subHeight = subRectTransform.sizeDelta[1];
82         subRectTransform.anchoredPosition = new Vector3(0, -(mainHeight
83             + (subHeight / 2) + yMargin + yOffsetTotal), 0);
84
85         yOffsetTotal += subHeight + yMargin;
86     }
87
88     var textContainerRect = textContainer.GetComponent<RectTransform>();
89     var columnHeight = mainHeight + yOffsetTotal + yMargin;
90     textContainerRect.sizeDelta = new Vector2(frameWidth, columnHeight);
91     textContainerRect.anchoredPosition = new Vector3(0, -columnHeight /
92         2, 0);
93
94     frameGeometry.transform.localScale = new Vector3(frameWidth +
95         frameMargin, columnHeight + frameMargin, 0.1f);
96     frameGeometry.transform.localPosition = new Vector3(0, -columnHeight
97         / 2, 0.1f);
98
99     //transform.localPosition += new Vector3(0, mainHeight + 0.5f, 0);
100    if(!deepRefresh) transform.localPosition += new Vector3(0, 2f, 0);
101    */
102 }
103 }
```

6.5 Material Manager

Important script handling all dynamic material instances - ensures that procedural material generation goes through a central dictionary and uses the same materials if already instantiated, instead of creating a new instance for 10000s of comments.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using System;
4 using UnityEngine;
5
6 public class MaterialManager : MonoBehaviour
7 {
8
9     public Material defaultMaterial;
10    public Material defaultNeon;
11    public Material skyBox;
12    public int dayInterval;
13    private Dictionary<string, Material> materials = new Dictionary<string,
14                                Material>();
15
16    // Start is called before the first frame update
17    void Start()
18    {
19    }
20
21    void Update()
22    {
23        AdjustDayNight();
24    }
25
26    public Material InstantiateMaterialWithColor(string materialName, Color
27                                                 materialColor, Material material = null)
28    {
29        //Debug.Log("Instantiating material - " + materialName);
30        if(material == null)
31        {
32            material = defaultMaterial;
33        }
34
35        if(!materials.ContainsKey(materialName))
36        {
37            Material newMaterial = Material.Instantiate(material);
```

```

37         newMaterial.SetColor("_BaseColor", materialColor);
38         Debug.Log("Instantiating new material - " + materialName);
39
40         materials.Add(materialName, newMaterial);
41     }
42
43     return materials[materialName];
44 }
45
46 public Material InstantiateMaterialWithEmission(string materialName,
47     Color materialColor, float intensity, Material material = null)
48 {
49     //Debug.Log("Instantiating material - " + materialName);
50     if(material == null)
51     {
52         material = defaultNeon;
53     }
54
55     if(!materials.ContainsKey(materialName))
56     {
57         Color emissionColor = new Vector4(materialColor.r * intensity,
58             materialColor.g * intensity, materialColor.b * intensity, 1.
59             Of);
60
61         Material newMaterial = Material.Instantiate(material);
62         newMaterial.SetColor("_EmissionColor", emissionColor);
63         //Debug.Log("Instantiating new material - " + materialName);
64
65         materials.Add(materialName, newMaterial);
66     }
67
68     return materials[materialName];
69 }
70
71 public void AdjustDayNight()
72 {
73     float currentTime = DateTime.Now.Minute + DateTime.Now.Second/60f;
74     float scaledDay = (Mathf.Abs(currentTime / 59f - 0.5f) - 0.25f) * 3;
75
76     RenderSettings.skybox.SetFloat("_Exposure", exposure);
77 }
78 }
```

6.6 Lib Color

I wanted more stuff in here but it's just a random color function at the moment that takes in HSV values (easier to control result).

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public static class LibColor
6 {
7
8     public static Color RandomiseHSV(Color inputColor, float varianceHue=0.0
9         5f, float varianceSaturation=0.05f, float varianceValue=0.05f )
10    {
11        float[] HSV = { 0, 0, 0 };
12        Color.RGBToHSV(inputColor, out HSV[0], out HSV[1], out HSV[2]);
13        HSV[0] = Random.Range(HSV[0] - varianceHue, HSV[0] + varianceHue);
14        HSV[1] = Random.Range(HSV[1] - varianceSaturation, HSV[1] +
15            varianceSaturation);
16        HSV[2] = Random.Range(HSV[2] - varianceValue, HSV[2] + varianceValue
17            );
18
19
20    }
21 }
```

6.7 ColorRandomiser

Monobehaviour script that makes use of the aforementioned color randomising function.

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class ColorRandomiser : MonoBehaviour
6 {
```

```

7   public Color platformColor;
8   public float varianceHue = 0.05f;
9   public float varianceSaturation = 0.05f;
10  public float varianceValue = 0.05f;
11  private float[] HSV = { 0, 0, 0 };
12
13
14 // Start is called before the first frame update
15 void Start()
16 {
17     Color.RGBToHSV(platformColor, out HSV[0], out HSV[1], out HSV[2]);
18     HSV[0] = Random.Range(HSV[0] - varianceHue, HSV[0] + varianceHue);
19     HSV[1] = Random.Range(HSV[1] - varianceSaturation, HSV[1] +
20         varianceSaturation);
21     HSV[2] = Random.Range(HSV[2] - varianceValue, HSV[2] + varianceValue
22         );
23
24     //Change platform material color
25     Color newColor = Color.HSVToRGB(HSV[0], HSV[1], HSV[2]);
26     GetComponent<MeshRenderer>().material.color = newColor;
27 }
28
29 // Update is called once per frame
30 void Update()
31 {
32 }
```

6.8 ColorChanger

Clever universal script that allows you to set specific materials to an array that is unique to this component instance, then call their indices through another script to change object materials.

```

1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4
5 public class ColorChanger : MonoBehaviour
6 {
7     Material redMat;
8     Material blueMat;
```

```

9     Material greenMat;
10    Material clearMat;
11    Material defaultMat;
12    public List<Material> materialList = new List<Material>();
13
14    void Start()
15    {
16        defaultMat = GetComponent<MeshRenderer>().material;
17        materialList.Insert(0, defaultMat);
18
19        redMat = Resources.Load("Materials/Select-Red", typeof(Material)) as
20            Material;
21        blueMat = Resources.Load("Materials/Select-Blue", typeof(Material))
22            as Material;
23        greenMat = Resources.Load("Materials/Select-Green", typeof(Material))
24            as Material;
25        clearMat = Resources.Load("Materials/Select-Clear", typeof(Material))
26            as Material;
27    }
28
29    public void SetMaterialDefault()
30    {
31        GetComponent<MeshRenderer>().material = defaultMat;
32    }
33
34    public void SetColorRed()
35    {
36        GetComponent<MeshRenderer>().material = redMat;
37    }
38
39    public void SetColorBlue()
40    {
41        GetComponent<MeshRenderer>().material = blueMat;
42    }
43
44    public void SetColorGreen()
45    {
46        GetComponent<MeshRenderer>().material = greenMat;
47    }
48
49    public void SetColor(int colorIndex)
50    {
51        if(colorIndex < materialList.Count)
52        {
53            GetComponent<MeshRenderer>().material = materialList[colorIndex]
54        }
55    }

```

```
54 }  
55 }  
56 }
```


