Script reference

Youtube tutorials | *Forum* | *Spline mesh deform documentation*
Spline plus Documentation | *Spline plus Scripting reference*
Email: elseforty@gmail.com

API for path generation during runtime

- **Create Spline plus**

```
    public static SPData CreateSplinePlus(Vector3 pos)
```

this creates a spline+ game object at a Vector3 position , the SPData object is returned by this function so you can catch it and used it as a parameter for other functions like functions below,

- **Create Node**

```
    public static Node CreateNode(SPData SPData, Vector3  position)
```

so what this does is generating a node object from a Vector3 so it can be used to draw your Spline+ branches.
This function returns the Node Object created,

a Node class holds node data such as node transforms.

```
    public class Node
    {
    public Transform Point;
    public Transform Point1;
    public Transform Point2;
    public NodeType _Type = NodeType.Smooth; // smooth or free

    public int SpeedFactor = 100;// varies from 0 to 100
    public int NormalFactor = 0;// varies from 0 to 1
    public Vector3 Normal;
    public Vector3 Tangent;
}
```

- **Duplicate Node**

```
    public static Node DuplicateNode(SPData SPData, Node originNode)
```

this duplicates a node object , the parameters you need to provide is the node you want to duplicate and the SPData object.
This function returns the duplicate Node Object,

Example:

```
Node duplicateNode=SplinePlusAPI.DuplicateNode(SPData,originNode);
```

- **Add new branch at node**

```
    public static int AddNewBranchAtNode(SPData SPData, Node Node)
```

Example:

```
 int branchKey= SplinePlusAPI.AddNewBranchAtNode( SPData,Node);
```

this creates a new branch at a node , you need to provide the SPData object and the node which will form the origin of the newly created branch, this function returns the key of the newly created branch

You can access that branch like this

```
var branch =SPData.DictBranches[branchKey];
```

- **Add node at end of branch**

```
    public static void AddNodeAtEndOfBranch(SPData SPData,int branchKey, Node Node)
```

this adds a node at the end of a branch , the branch needs to be unwelded,this function can be used after calling "`AddNewBranchAtNode()`" for an example , add extra nodes to the newly created branch,
you need to provide the `SPData` object , the branch key and the node object you want to add to the branch,
nodes are created using this function "`CreateNode()`"

Example:

```
 SplinePlusAPI.AddNodeAtEndOfBranch( SPData,0,Node0);
```

- **Connect two nodes**

```
    public static int ConnectTwoNodes(SPData SPData, Node node1, Node node2)
```

Example:

```
Int branchKey = SplinePlusAPI.ConnectTwoNodes(SPData, node1, node2);
```

this creates a branch between two nodes , the int returned is the newly created branch key ,
you need to provide the `SPData` object and first and second node.

- **Smooth shared node**

```
   public static void SmoothSharedNode(SPData SPDatas, SharedNodesData sharedNode, float
radius)
```

This smoothen a particular shared node,
You need to provide the SPData object the shared node Object and the radius value;

Example:

```
 SplinePlusAPI.SmoothAllSharedNodes(SPDatas,sharedNode,0.2f);
```

- **Smooth all shared nodes**

```
   public static void SmoothAllSharedNodes(SPData SPDatas,float radius)
```

this method smoothen all the shared nodes found in an SPData object ,
you need to provide the `SPDatas` object and the Radius value .

Example:

```
 SplinePlusAPI.SmoothAllSharedNodes(SPDatas,0.2f);
```

- example code for the generation of a square spline shape with chamfered edges:

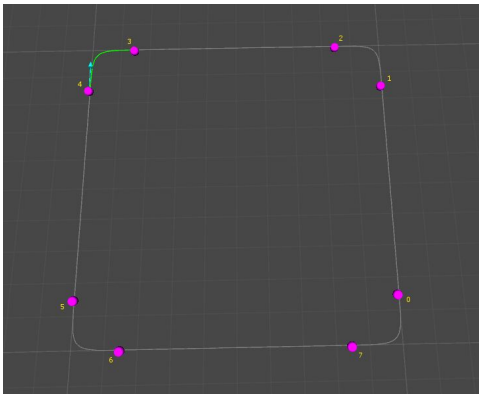
```
void Start () {

    var SPData=SplinePlusAPI.CreateSplinePlus(Vector3.zero);

    var node1= SplinePlusAPI.CreateNode(SPData, new Vector3(0, 0, 0));
    var node2 = SplinePlusAPI.CreateNode(SPData, new Vector3(10, 0, 0));
    var node3 = SplinePlusAPI.CreateNode(SPData, new Vector3(10, 0, 10));
    var node4 = SplinePlusAPI.CreateNode(SPData, new Vector3(0, 0, 10));

        SplinePlusAPI.ConnectTwoNodes(SPData, node1, node2);
       SplinePlusAPI.ConnectTwoNodes(SPData, node2, node3);
       SplinePlusAPI.ConnectTwoNodes(SPData, node3, node4);
       SplinePlusAPI.ConnectTwoNodes(SPData, node4, node1);

        SplinePlusAPI.SmoothAllSharedNodes(SPData, 0.5f);
    }
```



Followers setup:

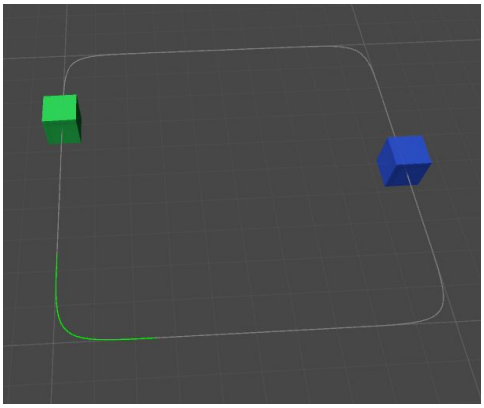
Now as a next chapter of the example above we will try to add followers during runtime,
You can call this function in the Start() after the runtime spline creation code above

```
    void FollowerSettings()
    {
        SPData.Followers[0]._FollowerAnimation = FollowerAnimation.AutoAnimated;
        SPData.Followers[0].FollowerGO = Follower1;
        SPData.Followers[0].Speed = 10;
        SPData.Followers[0].IsActive = true;

        SPData.Followers.Add(new Follower());

        SPData.Followers[1]._FollowerAnimation = FollowerAnimation.AutoAnimated;
        SPData.Followers[1].FollowerGO = Follower2;
        SPData.Followers[1].Speed = 5;
        SPData.Followers[1].IsActive = true;
    }
```

What we did above is adding two followers to the followers list , we assigned their gameObjects, Speed, animation type and we set them as active path followers ,



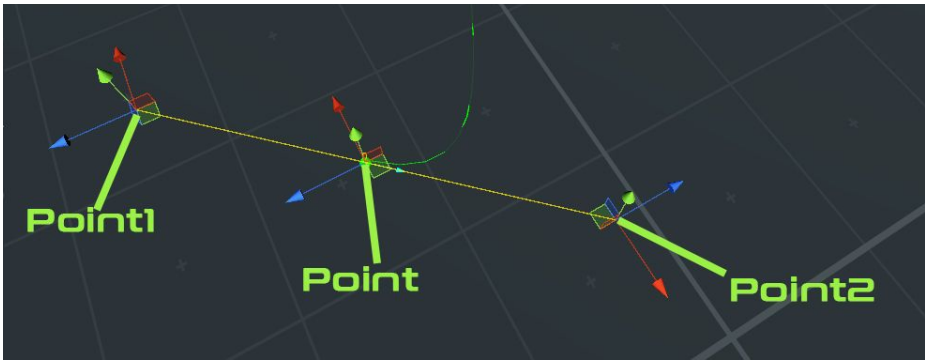
nodes

Node is a class that holds node data

A look at Node class components

```
      public class Node
    {
        public Transform Point;
        public Transform Point1;
        public Transform Point2;

        public NodeType _Type = NodeType .Smooth;  // Smooth or Free
        public float SpeedFactor = 1;
        public float NormalFactor = 1;
        public Vector3 Normal;
        public Vector3 Tangent;
 }
```



Current branch data

Access current branch data " Vertices, Normals, Tangents,nodes"

Code sample here

```
public class YourClassName
{
   public SPData SPData;
   void Start()
   {
       //SPData.Selection._BranchKey is the key of the currently selected branch in the
editor,
       int n =SPData.Selections._BranchKey;
       List<Vector3> branchVertices = SPData.DictBranches[n].Vertices;
       List<Vector3> branchTangents = SPData.DictBranches[n].Tangents;
       List<Vector3> branchNormals = SPData.DictBranches[n].Normals;
       List<Node> branchNodes = SPData.DictBranches[n].Nodes;
   }
}
```

A look at Branches class components

```
    public class Branches
    {
        public List<Vector3> Vertices = new List<Vector3>();
        public List<Vector3> Tangents = new List<Vector3>();
        public List<Vector3> Normals = new List<Vector3>();
        public List<Node> Nodes = new List<Node>();

        public List<float> SpeedFactor= new List<float>();
        public float[] VertexDistance;
        public float BranchDistance;
    }
```

Access branch nodes

Access current branch nodes data

Code sample here

```
public class YourClassName
{
   public SPData SPData;
   void Start()
   {
       var branch = SPData.DictBranches[0];

//access first node of the branch with key 0
       Node Node= branch.Nodes[0] ;


   }
}
```

Follower

follower object that holds Follower data you define to follow the spline,
in spline+ you have the ability to define multiple followers for one Spline+ gameObject,
to access the list of all the follower you have on a Spline+ object use this code

```
public class YourClassName
{
   public SPData SPData;
   void Start()
   {
       List<Follower> listOfAllFollowers=SPData.Followers;
       Follower firstFollowerOnList = listOfAllFollowers[0];
       GameObject firstFollowerGameObject=  firstFollowerOnList.FollowerGO;
       float firstFollowerSpeed= firstFollowerOnList.Speed;
       Bool firstFollowerState= firstFollowerOnList.IsActive;
      FollowerAnimation firstFollowerAnimType = firstFollowerOnList._FollowerAnimation;
       Int firstFollowerCurrentbranchKey= firstFollowerOnList._BranchKey;
       int firstFollowerProgree= firstFollowerOnList.Progress;
   }
}
```

Follower data

```
DistanceData DataExtraction(SPData SPData,Follower Follower);
```

a function that returns a class "DistanceData" which contains position and rotation data also as the index of the first vertex on the segment that the follower is on ,

you need to provide the SPData object also as the follower object to the function

Code sample here

```
public class YourClassName
{
    public SPData SPData;
   DistanceDataClass DistanceDataClass = new DistanceDataClass();
   void Start()
   {

      DistanceData distanceData = DistanceDataClass.DataExtraction(SPData,   SPData.Followers[0]);
       Vector3 currentFollowerPos =distanceData.Position;
       Quaternion currentFollowerRot =distanceData .Rotation;
       int currentFollowerBranchVertexIndex= distanceData .Index;
   }
}
```

example of the components of the class DistanceData:

```
public class DistanceData
{
  public int Index = 0;
  public Vector3 Position;
  public Quaternion Rotation;
}
```

Selections

Elements that you can access in edit mode in case you want to add more customized features based on your project need .

Code sample here

```
public class YourClassName
{
      public SPData SPData;
   void Start()
   {
       int currentFollowerIndex =SPData.Selections._Follower;
       int currentbranchKey=SPData.Selections._BranchKey;
       Node currentNode =SPData.Selections._Node;
       int currentSharedNodeIndex =SPData.Selections._SharedNodeIndex;
       int currentLocalNodeIndex =SPData.Selections._LocalNodeIndex;//local index on the branch
   }
}
```

Follower Branch

public void GoToBranch(int key) is used to switch follower from branch to another branch , you need to provide next branch key as parameter

Code sample here

```
public class YourClassName
{
    public SPData SPData;
   void Start()
   {
       SPData.Selections._Follower = 5;
       SPData.GoToNewBranch(2);
     //switch the fifth follower in the followers list to the branch with key2
     // same as
       Follower follower = SPData.Followers[5];
       follower._BranchKey = 2;
       follower.Progress = 0;
   }
}
```

Follower Speed

public void SetSpeed(float Speed)

Code sample here

```
using SplinePlusAll;
public class YourClassName
{
    public SPData SPData;
   void Start()
   {
       SPData.Selections._Follower = 5;
       SPData.SetSpeed(2);
       //Change the speed of the fifth follower on the followers list to 2
     // same as
       Follower follower = SPData.Followers[5];
       follower.Speed = 5;
   }
}
```

Follower progress

```
public void SetProgress(float Progress)
```

Code sample here

```
using SplinePlusAll;
public class YourClassName
{
    public SPData SPData;
   void Start()
   {
       SPData.Selections._Follower = 5;
       SPData.SetProgress(45.4f);
       //Change the progress of the fifth follower on followers list to 45.4f
     // same as
       Follower follower = SPData.Followers[5];
       follower.Progress = 45.4f;
   }
}
```