**Playlist Handling**

**PRQ**

* Before running the application, please ensure that Common is installed.
* We added 'WaspMosDataEntry.dll' to the 'bin' folder of the application to eliminate the dependency on the 'DataBuzz' setup.

**Connect and Initialize helper**

* In this we check that kernel controller is connected or not than we get service helper

for further use.

KCHelper kCHelper = new KCHelper(remoteurl);

if(kCHelper.KCConnected)

{

var objHelper = kCHelper.GetService(Services.TemplateManager);

CTemplateManagerHelper TemplateManagerHelper = objHelper as CTemplateManagerHelper;

}

**Load Pool**

CMosDataEntry \_ \_dataEntryControl = new CMosDataEntry();

\_dataEntryControl.IsNLE = false;

\_dataEntryControl.InitialiseObject("", "", "");

\_dataEntryControl.OnDataInstancePostUpdate += \_dataEntryControl \_OnDataInstancePost;

\_dataEntryControl.Dock = DockStyle.Fill;

**Check playlist is present or not**

//Arg1= Playlist slug/ID

//Arg2= IsSlug or not

var data= \_ProgramHelper.GetPlaylistData(playlistslug, true);

**Add Playlist**

//Arg1= Slug of playlist

//Arg2= Description

//Arg3= Playouttype ID //CG: 3711AFD0-E8D3-4939-A990-9554F31F2A80 //standard: "03AD7130-FAF2-4540-834F-24A95201DC44"

//Arg4= CreatedBy

var response = \_ProgramHelper.AddNewPlaylist(playlistslug, "", "03AD7130-FAF2-4540-834F-24A95201DC44", "Wasp");

**Add Group In Playlist**

//Arg1= Playlist Id

//Arg2=Target instance/group ID i.e. instance in which you want to add

//Arg3= Target group if group is present otherwise playlistid

//Arg4= Slug name of group

//Arg5= Description of group

//Arg6= Group type Ex {Group, TimeBase Group, Date TimeBase Group}

//Arg7= Loop counter of group

//Arg8= Visibility of group i.e. Checked or Unchecked

//Arg9= Insert before or not

//Arg10= Metadata xml if required

//Arg11= Approved or not

string[] groupdata= \_PlaylistHelper. AddGroupInPlaylist(\_PlaylistID

, string.Empty

, string.Empty

, Groupslug

, Groupslug

, "Group"

, "-1,0"

, "Checked"

, false

, ""

, true);

**Add Instance in Playlist**

//Root level

//Helper call to add instance in playlist

//Arg1=Playlist id

//Arg2=InstanceId

//Arg3=Target instance/group ID i.e. instance in which you want to add

//Arg4= Target group if group is present otherwise playlistid

\_PlaylistHelper.AddInstanceInPlylstByID(\_PlaylistID, objDraggedInstance.Id, "", "");

//In group

PlaylistHelper.AddInstanceInPlylstByID(\_PlaylistID, objDraggedInstance.Id, data.Id, data.Node);

//Below instance

\_PlaylistHelper.AddInstanceInPlylstByID(\_PlaylistID, objDraggedInstance.Id, data.Id, groupdata.Node);

**Remove Instance/Group**

//Call to remove instance or group from playlist

//Arg1= Playlist id

//arg2= NodeIds

//Arg3=Username

//Args4= Machinename

\_PlaylistHelper.RemoveItem(\_PlaylistID, nodeids, "Wasp","Wasp");

**Initialize RabbitMQ**

IMQReceiver \_receiver = null;

\_receiver = RabbitMQReceiver.GetObject();

\_receiver.MessageReceived += MQMessageReceived;

\_receiver.Initialize(mqdata);

\_receiver.Start();

**Initialize MSMQ**

string smodulename = "WASPI";

smodulename = smodulename.ToLower();

queuename = "waspUpdates." + smodulename;

queuelabelname = "waspUpdates." + smodulename + ".lbl";

MessageReciever \_objMessageReciever = new MessageReciever();

//Arg1 = Ip address in which MSMQ hosted

//Arg2= Port in which MSMQ hosted

//Arg3= Name of the Queue

//Arg4= label of queue

\_objMessageReciever.Initalize(multicastip, multicastport, queuename, queuelabelname);

\_objMessageReciever.OnMsgReceived += new MessageReciever.MessageReceiveEventHandler(m\_objMessageReciever\_OnMsgReceived);