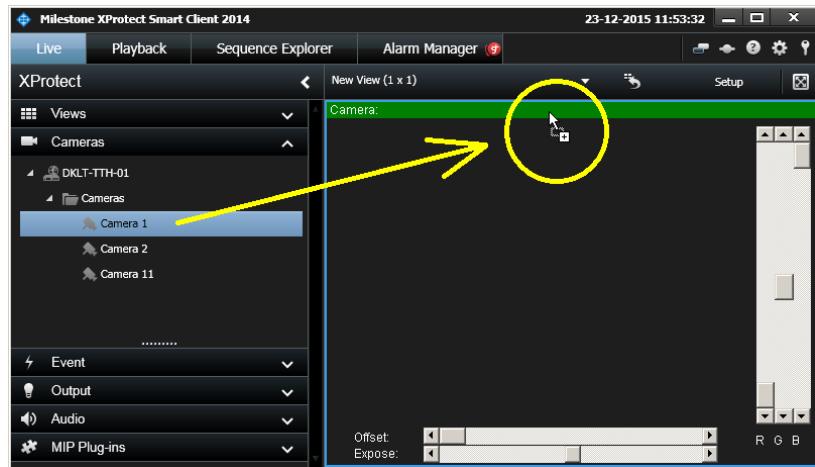


## Drag'n'drop enabled MIP plugin

Drag'n'drop has so far been limited to dragging cameras into views, whereas MIP plugins have been unable to utilize drag'n'drop. This sample shows how to enable dropping cameras onto your MIP plugin.

To demonstrate, the existing sample "RGBEnhanceVideo" has been chosen and extended. Note that the title bar has been chosen for Drag'n'drop, not the actual video area:



### How to implement drag'n'drop

1. Enable dropping on the control where you want to drop off a camera (property `AllowDrop`).
2. Define two event handlers for your control: `OnDragEnter` (preparing the actual drop when the cursor enters the drop area) and `OnDragDrop` (handling the actual drop).
3. Define an (empty) class `VideoOS.RemoteClient.Application.DragDrop.DraggedDeviceIdList`.  
This is necessary because the event data of the `DragEnter` and `DragDrop` events are of this type, and without having the right data type, you cannot extract the GUID of the dragged camera

### The code essentials

```
void pictureBox_OnDragEnter(object sender, System.Windows.Forms.DragEventArgs e)
{
    string[] szDragDataFormats = e.Data.GetFormats();
    // only allow drop if the dragged item is a camera list
    if (szDragDataFormats[0] == "VideoOS.RemoteClient.Application.DragDrop.DraggedDeviceIdList")
    {
        // retrieve the GUID of the dragged camera
        List<Guid> myCameraList = (List<Guid>)e.Data.GetData(szDragDataFormats[0]);
        myCameraGuid = myCameraList[0];

        // allow the Windows drop event to fire
        e.Effect = DragDropEffects.All;
    }
}

void pictureBox_OnDrop(object sender, DragEventArgs e)
{
    Item myDraggedCamera = new Item();
    myDraggedCamera = Configuration.Instance.GetItem(myCameraGuid, Kind.Camera);
    _viewItemManager.SelectedCamera = myDraggedCamera;
}
```

### Download Link

<http://download.milestonesys.com/MIPSdk/Samples/DragAndDropEnabledPlugins/DragAndDropEnabledPlugins.zip>