**NOTES ABOUT HOW .SCENE FILES WORK**All the information we need is contained in the scnSceneResource array; inside of it we can find different entries each of which is responsible for certain elements of the scene players will see. For example, we have the entry named “arry:scnEntryPoint” that allows us to see which are the point of interaction available to be used.  
  
Of the most importance is, though, the “handle:scnSceneGraph”, which allows us to get a graphical representation of what would the quest designer see on his/her end.  
More precisely, the path we should follow for example to see where does a StartNode connect to:  
1. Open “handle:scnSceneGraph”  
2. Open “scnSceneGraph”  
3. Open “array:handlescnSceneGraphNode”  
4. Open “scnStartNode” (this will be numbered differently depending on where it is positioned inside the struct Data array)  
5. Open “array:scnOutputSocket”  
6a. Open “array:scnInputSocketId”  
7a. Here you will be shown different “scnInputSocketId”, just select one of them (doesn’t really matter which)  
8a. Open “scnNodeId” and read the value inside the Uint32 row, that’s the number of the node to which our “scnStartNode” connects to.  
  
6b. If there is no “array:scnInputSocketId” then you will be probably be shown different “scnOutputSocket”; open one of them and you will be fine  
7b. Open “array:scnInputSocketId”  
8b. As in 7a here you will be shown different “scnInputSocketId”; select one of them (again, doesn’t really matter which)  
9b. Open “scnNodeId” and read the value inside the Uint32 row, that’s the number of the node to which our “scnStartNode” connects to.  
  
7c. If there is no “array:scnInputSocketId” in the “scnOutputSocket” you opened just go to the next until you find one and repeat steps from 8b to 9b.  
[NOTE FOR CODING: It’s important that the steps from 7a/b/c forward are repeated for each “scnInputSocketId” until you run out of them; then for the graphical representation it should be enough to display the node for which you completed the process while retaining in memory the information about what connections it has // alternatively you could follow this scheme here]

[Display the StartNode, then do the steps above just for the first scnInputSocketId until you’ve run out of first scnInputSocketId for the others connected node, display the sequence, then switch to the second scnInputSocketId for the StartNode, do the same, then the last an finally change to the second node in the sequence and start the process all over again]

Apparently it is enough to just swap the .scene files with the gold version for getting them to work

**TYPES OF NODES**StartNode  
SceneNode  
PhaseNode  
InputNode  
OutputNode  
DeletionNode  
PauseNode / PauseConditionnode  
InteractiveObjectManagerNode  
DeviceManagerNode  
**FOUND IN .scene FILES**  
ConditionNode  
QuestNode  
HubNode  
EndNode (only one)  
ChoiceNode  
SectionNode  
RandomizerNode  
XorNode  
DialogLineEvent  
**FOUND IN .questphase FILES**  
FactsDBManagerNode  
LogicalHubNode  
ConditionNode  
WorldDataManagerNode  
AudioNode  
TeleportPuppetNode  
UIManagerNode  
RenderFXNode  
LogicalXorNode (as XorNode, it lets just one signal out, but is found inside a PauseNode)  
RewardManagerNode  
CheckpointNode  
LogicalHubNode (as HubNode, it lets different signal out at the same time, but is found inside a PauseNode)  
CharacterManagerNode  
PhoneManagerNode  
CutControlNode  
SwitchNode (one input, many outputs)