Subject: Re: Drag and Drop between instances [FEATURE REQUEST]
Posted by mirek on Thu, 13 Jan 2011 08:47:33 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Wed, 12 January 2011 19:53Hi Mirek,

Yes that was a typo and should have been DragLeave()

This is my code as it stands just now.

```
void LayerTree::DropInsert(int parent, int ii, PasteClip& d)
{
 AdjustAction(parent, d);
 if(!IsDragAndDropSource() && d.Accept("externalInstance"))
 {
  m_ptr->SetStatus("dropped");

  StringStream ss(d.Get());

  Layer* pLayer = Layer::Load(ss);

  if(pLayer==NULL)
   return;

  if(parent==0)
  {
   m_ptr->AddLayer(pLayer);
   return;
  }

  TreeCtrl::Node node = GetNode(parent);
  LayerOption* ptr = (LayerOption*)~node.ctrl;
  Layer* pParent = ptr->GetLayer();

  if(pParent)
  {
   pParent->AddChild(pLayer);
   m_ptr->SetTree();
   m_ptr->RefreshNow();
  }
  else
  {
   m_ptr->AddLayer(pLayer);
  }
 }
 else if(AcceptInternal<LayerTree>(d, "mytreedrag") )
 {
```

```
  const TreeCtrl &src = GetInternal<LayerTree>(d);
  Vector<int> sel = src.GetSel();
  SaveStateToLayers(); // we need to rebuild the tree
  for(int i=0;i<sel.GetCount();i++)
  {
   Drop(parent,sel[i],ii);
  }
  m_ptr->SetTree();
  SetFocus();
  return;
 }

}

void LayerTree::Drag()
{
 if(m_ptr->AreWeBusy())
  return;

 if(DoDragAndDrop(InternalClip(*this, "mytreedrag"),
                  this->GetDragSample()) == DND_MOVE)
 {
  RemoveSelection();
 }
}


void LayerTree::DragLeave()
{


 int id;

 id = GetCursor();
 if(id<=0) // if id==0 then its the root node which is not a layer
  return;

 // copy to clipboard
 TreeCtrl::Node node = GetNode(id);
 LayerOption* ptr = (LayerOption*)~node.ctrl;
 Layer* pLayer = ptr->GetLayer();

 StringStream ss;

 ss.SetStoring();

 ss.Put("layer");
 int type = int(pLayer->GetType());
```

```
 ss.Put32(type);

 pLayer->Serialize(ss);

 String sLayer(ss);

 String text = pLayer->GetName();

 Size isz = GetTextSize(text.ToWString(), StdFont());

 ImageDraw iw(isz);

 iw.DrawRect(isz, White);

 iw.DrawText(0, 0, text);

 VectorMap<String, ClipData> clip;

 clip.Add("externalInstance", sLayer);

 if(DoDragAndDrop(clip, iw) == DND_MOVE)
 {

 }

 TreeCtrl::DragLeave();

}
```

I am definitely missing something in my understanding of how these methods are triggered and what they do and so of how to get them to do what I want. I would like to leave the internal clip in tact and to only trigger the external clip once I know it is being dropped on another instance although I expect that is not possible.

Thanks,

Nick

Ah, I see. Well, without going into details, I can imagine that this can couse great havoc

DragLeave is intended only for visual feedback in dnd target. It is sort of MouseLeave equivalent (actually, all 'Drag' functions are...)

In any case, all dragged formats have to be known at the moment drag starts. If you do not want to create binary data at that moment, you should rather use

ClipData(const Value& data, String (*render)(const Value& data));

constructor - render is supposed to 'convert' data (which can be basically anything) into your binary binary data, in esence exactly what you are trying to do in DragLeave.