Sons of Thunder Software
SOFTWARE DEVELOPMENT AND CONSULTING SERVICES

Products Services Developer Resources Contact STS About STS

livecode

Handling Drag-and-Drop From Outside Revolution

>>> While we are on the subject, how does one manage to add
>>> drag and drop from desktop application functionality to rev
>>> based application icons and stacks?
>>
>> What are you trying to drag from "outside" into Revolution? Text,
>> images,
>> ... ?
>
> Text and Images would be a start, 3D models would an eventuality.

OK. Here we go:

TEXT

First of all, you should be able to drag raw text from another application to a Rev field automatically with no special setup or intervention whatsoever.

However, if you want to drag text files into a field and have the contents of the text file be put into the field, you can do this:

```
(script of field)
on dragEnter
  set the acceptDrop to true
  pass dragEnter
end dragEnter

on dragDrop
  put the dragData["files"] into tFileList
  if tFileList <> "" then
    put url("file:" & (line 1 of tFileList)) into me
  else
    beep
  end if
end dragDrop
```

Now the script above makes a number of assumptions: (1) that if the user were dragging multiple files, that you only care about the first one, and (2) that the file being dropped is actually a text file.

Let's deal with these one at a time:

1. The `dragData["files"]` contains a return-delimited list of full paths to files that are being dragged into Revolution. You could certainly do a repeat loop through the files to find what you want, or alert the user if there was more than one (`the number of lines of the dragData["files"] > 1`), etc.
2. This is where Rev kind of breaks down. There's no quick way to determine what the type of file being dropped *is*, although there is a laborious way (which I'll detail below after we talk about images). However you could do something simple, like assume the text file needs to have a .txt extension, and do a "filter" on the `dragData["files"]` on ".txt", as in:

   ```
   put the dragData["files"] into tFiles
   filter tFiles with "*.txt"
   ```

IMAGES

Suppose you wanted to drag a graphic from disk into an image object. This is very similar to the text example above.

```
on dragEnter
  set the acceptDrop to true
  pass dragEnter
end dragEnter

on dragDrop
  put the dragData["files"] into tFileList
  if tFileList <> "" then
    set the fileName of me to (line 1 of tFileList)
  else
    beep
  end if
end dragDrop
```

This assumes:

1. that you have an image object already created into which you want to drag the image file from disk*
2. that if the user were dragging multiple files, that you only care about the first one, and
3. that the file being dropped is actually an image file.

> _* You might have an empty image (perhaps through `create image`) or a pre-existing one. If you set the `lockLocation` of the image to TRUE, the incoming image would be scaled up/down to fit the rect of the image object. If the `lockLocation` of the image is FALSE,_

the image object would be scaled to fit the rect of the incoming image. I would recommend turning on the border and turning off the 3D if you want to have an image "drop region".

Now perhaps you don't want to have a pre-existing image object, but just want to import the image when it is dropped on "the card". You can do it this by having only a single object (it doesn't have to be an image) that is the size of the card and is transparent (like a button) that has this code:

```
on dragEnter
  set the acceptDrop to true
  pass dragEnter
end dragEnter

on dragDrop
  put the dragData["files"] into tFileList
  if tFileList <> "" then
    create image
    set the fileName of it to (line 1 of tFileList)
  else
    beep
  end if
end dragDrop
```

Same basic assumptions apply as the earlier code, and of course you might want to size the image object and lock it first, like:

```
create image
set the rect of it to 200,200,400,400
set the lockLocation of it to true
set the fileName of it to (line 1 of tFileList)
```

You get the idea.

Now we get to the fun part - the way to determine if the file(s) being dropped are OK for the field/image/etc. Here's an example of how to bring in text files ONLY for a field, which BTW concatenates the text from all files dropped on the field (watch the line wraps):

```
on dragEnter
  if hasTextFiles(the dragData["files"]) then
    set the acceptDrop to true
    focus me
  end if
end dragEnter

function hasTextFiles pFileList
  -- assumes all files are from the same folder, looks for
  -- any file that is a text file
  -- If found, sets a custom prop to be used during the drop
  -- looks for both file extension and type of file
  put the directory into tOldDir
  set the itemDel to "/"
  put "" into tTextFiles
  repeat for each line tFile in pFileList
    if tTextFiles = "" then
      put tFile into tDir
      delete item -1 of tDir
      if tDir = "" then put "/" into tDir  -- happens at root
      set the directory to tDir
      put the detailed files into tFiles
      replace "," with "/" in tFiles
    end if
    put lineOffset(cr& urlEncode(item -1 of tFile)&"/",cr&tFiles) into tLine
    if tLine <> 0 then  -- which should always be the case
      if the number of items of (line tLine of tFiles) = 10 then
        -- no type/creator
        put "" into tTypeCreator
      else
        put item -1 of line tLine of tFiles into tTypeCreator
      end if
      if (char -4 to -1 of tTypeCreator = "TEXT") or \
        (char -4 to -1 of tFile = ".txt") then put tFile & \
        cr after tTextFiles
    end if
  end repeat
  set the directory to tOldDir
  delete last char of tTextFiles
  set the uTextFiles of me to tTextFiles
  return (tTextFiles <> "")
end hasTextFiles

on dragDrop
  put the uTextFiles of me into tFiles
  repeat for each line tFile in tFiles
   put me & url ("file:" & tFile) & cr into me
  end repeat
end dragDrop
```

You can extend this model for checking for images, etc.

Hope this helps,

Posted on 9/11/04 by Ken Ray to the Use Revolution list

Print this tip

News and Rumors Products Services Developer Resources Contact STS About STS

Copyright ©1997-2013 Sons of Thunder Software, Inc. All rights reserved.
Send all comments to webmaster@sonsothunder.com.