

english_gtdrag

COLLABORATORS

	<i>TITLE :</i> english_gtdrag	
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>
WRITTEN BY		January 1, 2023
<i>SIGNATURE</i>		

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	english_gtdrag	1
1.1	gtdrag - Documentation	1
1.2	gtdrag - Copyright notice	1
1.3	gtdrag - The Author	2
1.4	gtdrag - Credits	2
1.5	gtdrag - Introduction	2
1.6	gtdrag - System requirements	3
1.7	gtdrag - Function Overview	3
1.8	gtdrag.lib/GTD_AddApp	3
1.9	gtdrag.lib/GTD_AddGadget	4
1.10	gtdrag.lib/GTD_AddWindow	5
1.11	gtdrag.lib/GTD_GetDragMsg	6
1.12	gtdrag.lib/GTD_FilterIMsg	7
1.13	gtdrag.lib/GTD_GetIMsg	7
1.14	gtdrag.lib/GTD_PostFilterIMsg	8
1.15	gtdrag.lib/GTD_RemoveApp	8
1.16	gtdrag.lib/GTD_RemoveGadget	9
1.17	gtdrag.lib/GTD_RemoveWindow	9
1.18	gtdrag.lib/GTD_ReplyDragMsg	10
1.19	gtdrag.lib/GTD_ReplyIMsg	10
1.20	gtdrag - Release Notes	10
1.21	gtdrag - Known Bugs	12
1.22	gtdrag - To do list	13

Chapter 1

english_gtdrag

1.1 gtdrag - Documentation

gtdrag - Version 2 Revision 10, 27.6.1997

This package is copyrighted ©1996-97 by
Axel Dörfler
All rights reserved.

Copyright notice

Introduction

System requirements

Functions

Author

Credits

Release Notes

Known Bugs

To do

This is a final release of the package. This doesn't mean that it has implemented all planned features yet, but the API will stay compatible with upcoming releases.

1.2 gtdrag - Copyright notice

The package is ©1996-97 by
Axel Dörfler
. All rights reserved.

You are allowed to copy it to BBS, Aminet and other free

shareware-pools unless all files in this package are provided and unchanged.

The use of gtdrag is at your own risk - I am not liable or responsible for any problems you might have.

1.3 gtdrag - The Author

You can reach me under the following address:

Snail-mail: Axel Dörfler
Heerstraße 53
49492 Westerkappeln
Germany

e-Mail: axeld@ax.westfalen.de
fidonet: 2:2449/413.23@fidonet

If you find a bug or have an idea to implement in gtdrag or if you want to become a beta tester of this library, please write me!

The gtdrag.library is compiled using the SAS/C-Compiler 6.56 on an A4000/040 with OS3.0 (V39).

1.4 gtdrag - Credits

Thanks to:

- Alexander Bartz (0541572992-1@t-online.de) for bug report and support.
- Daniel Rost (d96rost@ios.chalmers.se) for the creation of the E-modules and include files for gtdrag and the translation of the example program.

1.5 gtdrag - Introduction

The gtdrag shared library provides the possibility of the drag&drop-feature for gadtools-applications.

It is thought to close the gap between now and an upcoming version of AmigaOS with this features included. If it comes.

It is definitely not the best you can imagine, but it works good and is easy to use.

You drag an entry of listview when you select it and move the mouse horizontal more than 10 pixels. On other gadget types, you have to move >3 pixels to start dragging. Perhaps I'll implement a special "dragkey" like "alt" or whatsoever.

The drag&drop feature is not limited to a single window or application, every gadget you add, is included. But of course, it is limited to one screen at the same time.

With the version 1.5 gtdrag is implemented in a shared library. This gives several advantages against the old linker library; if someone needs a linker library, I can create one on request.

1.6 gtdrag - System requirements

gtdrag should work on any Amiga with OS 3.0+ (V39 and above). It is not possible to use it under OS 2.04.

But you can also build an application which runs under 2.04 and an installed gtdrag.library - in this case gtdrag works as a gate to gadtools without the dragging capabilities.

1.7 gtdrag - Function Overview

TABLE OF CONTENTS

gtdrag.lib/GTD_AddApp
gtdrag.lib/GTD_AddGadget
gtdrag.lib/GTD_AddWindow
gtdrag.lib/GTD_GetDragMsg
gtdrag.lib/GTD_FilterIMsg
gtdrag.lib/GTD_GetIMsg
gtdrag.lib/GTD_PostFilterIMsg
gtdrag.lib/GTD_RemoveApp
gtdrag.lib/GTD_RemoveGadget
gtdrag.lib/GTD_RemoveWindow
gtdrag.lib/GTD_ReplyDragMsg
gtdrag.lib/GTD_ReplyIMsg

1.8 gtdrag.lib/GTD_AddApp

NAME
 GTD_AddApp -- inits all app. dragging context data

SYNOPSIS

```
result = GTD_AddApp(name,firsttag,...);
D0          A0  A1
```

```
int GTD_AddApp(STRPTR,Tag,...);
```

FUNCTION

This function allocates and sets all needed data.
 You must call it before any other action with this
 library.
 Every name must be unique.

TAGS

GTDA_InternalOnly (BOOL) - if you don't want to react on
 dropped icons of other applications, you should set this
 flag to TRUE. Defaults to FALSE.

RESULT

result - TRUE for success.

SEE ALSO

```
GTD_RemoveApp()
,
GTD_GetIMsg()
```

1.9 gtdrag.lib/GTD_AddGadget

NAME
 GTD_AddGadget -- adds a drag&drop gadget to the internal list

SYNOPSIS

```
GTD_AddGadget(type,gad,win,firsttag,...);
D0  A0  A1  A2
```

```
void GTD_AddGadget(int,struct Gadget *,struct Window *,Tag,...);
```

FUNCTION

This function adds the chosen gadget to the internal list.
 You must call this function, if you want a gadget to have
 the drag&drop-feature.

TAGS

GTDA_Images (BOOL) - if specified, only the image of an ImageNode
 will be dragged. GTDA_Width & GTDA_Height MUST be set. Defaults
 to FALSE.

GTDA_NoDrag (BOOL) - if specified, you cannot drag an item of this
 gadget; you are only able to drop them over it. Defaults to FALSE.

GTDA_Same (BOOL) - for listviews. If specified, source and target of a drag can be the same. Defaults to FALSE.

GTDA_NoPosition (BOOL) - for listviews only. If specified, there is no position highlighting or scrolling. Defaults to FALSE.

GTDA_ItemHeight (short) - specifies the height of a listview entry. Defaults to the screen's font height.

GTDA_RenderHook (struct Hook *) - if provided, the dragged image will be drawn with this listview-callback hook. Otherwise there is only a ClipBlit() proceeded, which will not work in combination with simple refresh windows. You can set a special size with GTDA_Width & GTDA_Height. Defaults to NULL.

GTDA_Object (struct *ImageNode) - for non-listview gadgets: the node that will be dragged. If a render hook is provided, the object must be of a type accepted by the render hook.

GTDA_Mask (ULONG) - the mask value for the gadget. Defaults to 0xffffffff.

GTDA_AcceptMask (ULONG) - if the mask value of the dropped icon and (bit-wise) this value are TRUE, the drag will be accepted. Defaults to 0xffffffff.

GTDA_Width, GTDA_Height (short) - set the size of the image to be dragged.

INPUTS

type - type of the gadget (e.g. LISTVIEW_KIND)
gad - pointer to the gadget (from CreateGadget()).
win - pointer to the window of the gadget

NOTES

If the gadget type is not LISTVIEW_KIND, GTDA_Object and GTDA_Images or GTDA_RenderHook must be set, or DGF_NODRAG will be set.

Furthermore, you have to set the GA_Immediate-attribute to TRUE if want to have a draggable gadget.

The function should only be used with GTDA_NoDrag when using other gadget types than LISTVIEW_KIND or BUTTON_KIND. Otherwise it could make problems (check it out).

You MUST set the GadgetID field of your gadgets!

SEE ALSO

GTD_RemoveGadget ()
,
GTD_RemoveApp ()

1.10 gtdrag.lib/GTD_AddWindow

NAME

GTD_AddWindow -- adds drag&drop notify to a window

SYNOPSIS

```
GTD_AddWindow(win, firsttag, ...);

void GTD_AddWindow(struct Window *, Tag, ...);
```

FUNCTION

This function adds the chosen window to the internal list. You must call this function, if you want to receive drag&drop messages from a window.

TAGS

GTDA_AcceptMask (ULONG) - if the mask value of the dropped icon and (bit-wise) this value are TRUE, the drag will be accepted. Defaults to 0xffffffff.

INPUTS

win - pointer to the window

NOTE

You only receive a DMT_WINDOW message from `GTD_GetDragMsg()` if the object wasn't dropped over a gadget which supports drag&drop.

SEE ALSO

```
GTD_RemoveWindow()
,
GTD_RemoveApp()
```

1.11 gtdrag.lib/GTD_GetDragMsg

NAME

GTD_GetDragMsg -- get a DragMsg if possible

SYNOPSIS

```
dm = GTD_GetDragMsg();

struct DrawMsg *GTD_GetDragMsg(void);
```

FUNCTION

If there are drag messages in the queue, this function returns the first one.

RESULT

dm - a pointer to the message (NULL for none)

NOTE

The `dm_SourceApp` field of the returned `DragMsg` is NULL if the `dm_Source` is one of the application's `DragGadgets`.

SEE ALSO

GTD_ReplyDragMsg()

1.12 gtdrag.lib/GTD_FilterIMsg

NAME

GTD_FilterIMsg -- filter an IntuiMessage through gtdrag & gadtools.

SYNOPSIS

```
modmsg = GTD_FilterIMsg(imsg);  
D0          A0
```

```
struct IntuiMessage *GTD_FilterIMsg(imsg);
```

FUNCTION

see gadtools/GT_FilterIMsg().

INPUT

imsg - an IntuiMessage you got from GetMsg().

RESULT

modmsg - pointer to the modified IntuiMessage

SEE ALSO

GTD_PostFilterIMsg()

1.13 gtdrag.lib/GTD_GetIMsg

NAME

GTD_GetIMsg -- get an IntuiMessage, with all necessary processing

SYNOPSIS

```
msg = GTD_GetIMsg(port);
```

```
struct IntuiMessage *GTD_GetIMsg(iport);
```

FUNCTION

Gets an IntuiMessage and does all the drag processing including a call to the equivalent gadtools functions.

INPUT

port - a pointer to the msg port

RESULT

msg - pointer to the IntuiMessage

SEE ALSO

GTD_ReplyIMsg()

1.14 gtdrag.lib/GTD_PostFilterIMsg

NAME

GTD_PostFilterIMsg -- return filtered IntuiMessage from
GTD_FilterIMsg().

SYNOPSIS

```
msg = GTD_PostFilterIMsg(modimg);  
D0          A0
```

```
struct IntuiMessage *GTD_FilterIMsg(img);
```

FUNCTION

see gadtools/GT_FilterIMsg().

INPUT

modimg - an IntuiMessage you got from GTD_FilterIMsg().

RESULT

msg - pointer to the unmodified IntuiMessage

SEE ALSO

GTD_FilterIMsg()

1.15 gtdrag.lib/GTD_RemoveApp

NAME

GTD_RemoveApp -- frees the dragging context data

SYNOPSIS

```
GTD_RemoveApp();
```

```
void GTD_RemoveApp(void);
```

FUNCTION

Frees all the memory allocated by GTD_AddApp(), GTD_AddGadget() and
GTD_AddWindow() for an application.

You should not call it without the GTD_AddApp() call before.

SEE ALSO

```
GTD_AddApp()  
,  
GTD_AddGadget()  
,  
GTD_AddWindow()
```

1.16 gtdrag.lib/GTD_RemoveGadget

NAME
GTD_RemoveGadget -- removes a drag&drop gadget

SYNOPSIS
GTD_RemoveGadget(gad);

void GTD_AddGadget(struct Gadget *);

FUNCTION
This function removes the drag&drop feature from a gadget which was added to it with GTD_AddGadget().

INPUTS
gad - pointer to the gadget (from CreateGadget()).

SEE ALSO

GTD_AddGadget()
,
GTD_RemoveApp()

1.17 gtdrag.lib/GTD_RemoveWindow

NAME
GTD_RemoveWindow -- removes drag&drop notify from a window

SYNOPSIS
GTD_RemoveWindow(win);

void GTD_RemoveWindow(struct Window *);

FUNCTION
Removes drag&drop notify from a window which was added with GTD_AddWindow() before.

INPUTS
win - pointer to the window

SEE ALSO

GTD_AddWindow()
,
GTD_RemoveApp()

1.18 gtdrag.lib/GTD_ReplyDragMsg

NAME
GTD_ReplyDragMsg -- replies a DragMsg

SYNOPSIS
GTD_ReplyDragMsg(dm);

void GTD_ReplyDragMsg(struct DragMsg *);

FUNCTION
Replies a drag msg got from GTD_GetDragMsg().

INPUT
dm - a pointer to the drag msg

SEE ALSO

GTD_GetDragMsg()

1.19 gtdrag.lib/GTD_ReplyIMsg

NAME
GTD_ReplyIMsg -- replies an IntuiMessage

SYNOPSIS
GTD_ReplyIMsg(msg);

void GTD_ReplyIMsg(struct IntuiMessage *);

FUNCTION
Replies the msg got from GTD_GetIMsg(). It works like
GT_ReplyIMsg() - but you must call GTD_GetIMsg() before!

INPUT
msg - a pointer to the msg to be replied, maybe NULL.

SEE ALSO

GTD_GetIMsg()

1.20 gtdrag - Release Notes

gtdrag 1.1 (29.9.96)
- first release.

gtdrag 1.4 (3.10.96)
- big changes in structure and tag definitions! (1.2)

- other gadget types than LISTVIEW_KIND will be handled

correctly (like BUTTON_KIND). (1.2)

- added
 GTD_RemoveGadget()
 and
 GTD_RemoveWindow()
 for multi-window support. (1.2)
- checks now for the number of entries in a source-listview.
(1.3)
- the range of a mouse move until a drag happens can now be
changed via GTDA_(LV)DragPixel. (1.3)
- the dragged icon now hides if you are near the source gadget
to prevent graphical damage. (1.4)

gtdrag 1.8 (5.11.96)

- created a shared library, the linker lib is not provided anymore
but I can create one on request. This creation is part of big
changes in the whole application interface. E. g. the GTD_Init()/
Free() function-pair doesn't exist any longer. (1.5)
- implemented GTD_(Post)FilterIMsg()-functions. (1.6)
- checks now the number of entries in the target-listview. (1.6)
- makes the dragged object transparent if it is over a gadget
which accepts drag&drop. (1.6)
- now centers the dragged object if it is an image. (1.7)
- fixed problems with more than one window open at a time. (1.8)
- fixed the GTD_(Post)FilterIMsg()-functions; they replied the
same IntuiMessage two times. (1.8)

gtdrag 2.1 (7.11.96)

- implemented accept and type masks. (1.9)
- implemented GTDA_InternalOnly and GTDA_Same. (1.9)
- changed the handling of the DragMsgs internally. (1.10)
- bumped to V2 to let applications presuppose the extensions
to the last release. (2.1)
- it was possible to get a drag message (DMT_UNKNOWN) if a drag
was interrupted using the right mouse button. (2.1)

gtdrag 2.2 (11.11.96)

- if no DragGadget was defined, this could make problems. (2.2)
- fixed possible problems with the use of Wait() in applications.
(2.2)

- the GTDA_RenderHook got one pixel too much in width and height. (2.2)

gtdrag 2.7 (15.12.96)

- the drag objects are now rendered via BitMaps, they are no BOBs any longer to reduce flickering on gfx-cards. (2.3)
- include the E includes/modules made by Daniel Rost. (2.3)
- fixed small bugs in the render-routine. (2.3)
- a little more highlightning is done. (2.4)
- the render-routines were a bit optimised and changed for use with CyberGraphX. (2.5)
- the message handling is completely changed, now it uses fake events to let gadtools stop processing messages while an object is dragged. (2.6)
- small optimisations. (2.6)
- enhanced highlightning and listview-control. If you move an entry in a listview, you may get another dm_TargetEntry than in previous releases. (2.6)
- added scrolling capabilities to listview-drags. (2.6)
- some minor bug fixes. (2.7)

gtdrag 2.10 (27.6.97)

- a bug in the render-routine could cause a strange listview entry to appear when running under CyberGraphX with huge call-backs. (2.8)
- if the first IDCMP_MOUSEMOVE during a drag did not come from the source gadget, strange effects may happen. (2.8)
- no more "ghost" drags will occur when you release the mouse button while scrolling in a listview. (2.9)
- fixed a bug which could appear if the drag doesn't start over the gadget itself. (2.10)

1.21 gtdrag - Known Bugs

Unfortunately, there are bugs left:

- on CyberGraphX-screens with more than 256 colors the drag object is not rendered correctly.
 - it is much too slow on a graphic-card; I know the reason for that but I don't know how to fix it.
-

1.22 gtdrag - To do list

There are a few things on the "to do"-list:

- better CyberGraphX support (see
known bugs
)
- standards for "interprocess" drag&drop

That's not all, but I obviously forgot the rest.
