

**supermodel\_class**

<b>COLLABORATORS</b>
----------------------

	<i>TITLE :</i> supermodel_class		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 26, 2024	

<b>REVISION HISTORY</b>
-------------------------

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>supermodel_class</b>	<b>1</b>
1.1	supermodel_class.doc . . . . .	1
1.2	supermodel.class/--datasheet-- . . . . .	1
1.3	supermodel.class/--datasheet-GlueFunc--upermodel.class/--datasheet-GlueFunc-- . . . . .	2
1.4	supermodel.class/--datasheet-supericclass--l.class/--datasheet-supericclass-- . . . . .	3
1.5	supermodel.class/--History-- . . . . .	3
1.6	supermodel.class/ProcessTagList . . . . .	4
1.7	supermodel.class/SM_IsMemberOf . . . . .	4
1.8	supermodel.class/SM_NewSuperIC . . . . .	5
1.9	supermodel.class/SM_NewSuperModel . . . . .	5
1.10	supermodel.class/SM_SendGlueAttrs . . . . .	6
1.11	supermodel.class/SM_SICMAP . . . . .	7
1.12	supermodel.class/SMTAG_AddTag . . . . .	7
1.13	supermodel.class/SMTAG_AddTags . . . . .	8
1.14	supermodel.class/SMTAG_AllocTags . . . . .	8
1.15	supermodel.class/SMTAG_ClearNumTags . . . . .	9
1.16	supermodel.class/SMTAG_ClearTags . . . . .	10
1.17	supermodel.class/SMTAG_FreeTags . . . . .	10
1.18	supermodel.class/SMTAG_RemTag . . . . .	11
1.19	supermodel.class/SMTAG_TagDone . . . . .	11
1.20	supermodel.class/SMTAG_TagMore . . . . .	12

## Chapter 1

# supermodel\_class

### 1.1 supermodel\_class.doc

```
--datasheet-- ()
--datasheet-GlueFunc-- ()
--datasheet-supericclass-- ()
--History-- ()
ProcessTagList ()
SM_IsMemberOf ()
SM_NewSuperIC ()
SM_NewSuperModel ()
SM_SendGlueAttrs ()
SM_SICMAP
SMTAG_AddTag ()
SMTAG_AddTags ()
SMTAG_AllocTags ()
SMTAG_ClearNumTags ()
SMTAG_ClearTags ()
SMTAG_FreeTags ()
SMTAG_RemTag ()
SMTAG_TagDone ()
SMTAG_TagMore ()
```

### 1.2 supermodel.class/--datasheet--

```
NAME
    supermodel.class -- Model and IC class
```

```
SUPERCLASS
    modelclass
```

```
DESCRIPTION
```

```
METHODS
```

```
ATTRIBUTES
```

---

SMA\_AddMember - calls OM\_ADDMEMBER

SMA\_RemMember - calls OM\_REMMEMBER

SMA\_GlueFunc - Glue code, does tag mapping/conversion, etc.  
See example.  
see --datasheet-GlueFunc--

SMA\_GlueFuncA6 - If your glue code is in a library, set this to  
your Library base.

SMA\_GlueFuncUserData -

SMA\_CacheStringTag - cache string data

icclass Tags supported also

#### NOTES

#### BUGS

Doesn't fail if SMA\_AddMember is NULL during OM\_NEW.  
It will in the future.

#### SEE ALSO

## 1.3 supermodel.class/--datasheet-GlueFunc--upermodel.class/--datasheet-GlueFunc-

-

#### NAME

GlueFunc

#### SYNOPSIS

```
rv GlueFunc(GlueData, TagList, [userdata], [a6])
d0          a0          a1          a2          a6
```

```
ULONG GlueFunc(struct smGlueData *, struct TagItem *, APTR, APTR);
```

#### FUNCTION

GlueFunc is a function you provide to modify a tag list before being sent to the members and ICA\_TARGET of the modelclass.

You may modify TagList and use the SMTAG\_functions to modify TagList, however, don't free TagList.

Call SM\_SendGlueAttrsA() to send your new attributes back to the model class for notification.

#### NOTES

There is room for 50 tags in TagList

#### BUGS

#### SEE ALSO

## 1.4 supermodel.class/--datasheet-supericlass--l.class/--datasheet-supericlass--

### NAME

supermodel.class -- IC class

### SUPERCLASS

icclass

### DESCRIPTION

An interconnect object, creates a data path between a supermodel object and other objects (usually gadgets)

### METHODS

### ATTRIBUTES

SICA\_Model - set by model class when addmembered.

ICA\_TARGET - target object

You may set either ICA\_MAP or SICA\_TargetMap or (SICA\_ToTargetMap and SICA\_FromTargetMap)

ICA\_MAP - map tags

The supplied taglist is cloned and applied to the ic object.  
Another taglist is built by reversing the tags and applied to the targeted object.

SICA\_TargetMap -

The supplied taglist is cloned and applied to the targeted object.  
Another taglist is built by reversing the tags and applied to the ic object.

SICA\_ToTargetMap - Applied to tags sent from the  
ic object to the targeted object.

SICA\_FromTargetMap - Applied to tags sent from the  
targeted object to, likely, the modelobject

### NOTES

if ICA\_TARGET is valid,  
the ic object will set the Target's ICA\_TARGET and ICA\_MAP.

### BUGS

### SEE ALSO

## 1.5 supermodel.class/--History--

### To Do

#### 44.1 -

- \* GlueCode was freeing memory twice.
- \* Added SMA\_CacheStringTag - Anytime this tag is passed through

```

        the model class, the string supplied is cached, and ti_Data
        is updated to point to the cache.
44.2 -
    * Included version string
44.3 -
    * Now failes during OM_NEW if any SMA_AddMember is null.
    * Documentation update.
    * ICA_MAP support in superic was changed

```

## 1.6 supermodel.class/ProcessTagList

### NAME

ProcessTagList -- Macro to process a taglist

### SYNOPSIS

```
ProcessTagList (TagList, Tag, TState)
```

```
TState=TagList;
while (Tag=NextTagItem(&TState))
```

### EXAMPLE

```

void SomeFunc(struct TagItem *TagList)
{
    struct TagItem *tag, *tstate;

    ProcessTagList (TagList, tag, tstate)
    {
        seitch (tag->ti_Tag)
        {
            case GA_Left:
                ...
                break;
            etc...
        }
    }
}

```

## 1.7 supermodel.class/SM\_IsMemberOf

### NAME

SM\_IsMemberOf -- Check if Object belongs to a Class

### SYNOPSIS

```
memberof = SM_IsMemberOf (Object, ClassPtr, ClassID)
```

```
BOOL SM_IsMemberOf (Object *, Class *, STRPTR);
```

### FUNCTION

Determines if the Object is a member of the Class specified.

### INPUTS

Object - Object to check.  
 ClassPtr - (Class) May be NULL.  
 ClassID - (STRPTR) May be NULL.

#### RESULT

BOOL, non-zero on succes.

#### NOTES

Stolen from someone on the BOOPSI mailing list...

Here's some IsMemberOf() code I whipped up quickly. Might be nice if we all posted useful little BOOPSI snippets like this... maybe even collected them together on a web site. (I volunteer NOT to maintain this site :)

## 1.8 supermodel.class/SM\_NewSuperIC

#### NAME

SM\_NewSuperIC -- Allocate SuperIC object

#### SYNOPSIS

```
icobject=SM_NewSuperICA(TagList)
a0                                a0

Object *SM_NewSuperICA(struct TagItem *);

icobject=SM_NewSuperIC(Tags, ...)
a0                                a0

Object *SM_NewSuperIC(Tag, ...);
```

#### FUNCTION

Allocates an SuperIC class object

#### INPUTS

TagList - see --datasheet-supericclass--

#### RESULT

Pointer to an ic object on success, if ICA\_TARGET is valid, the ic object will set the Target's ICA\_TARGET and ICA\_MAP.

## 1.9 supermodel.class/SM\_NewSuperModel

#### NAME

SM\_NewSuperModel -- Allocate SuperModel object

#### SYNOPSIS

```
model = SM_NewSuperModel( Tag1, Data1, TAg2, ...)

Object *SM_NewSuperModel(Tag Tags, ...);
```



## FUNCTION

Allocate model object.

## INPUTS

Tags

## RESULT

## EXAMPLE

## NOTES

DisposeObject() returned model when done.

Before targeted objects (ie gadgets) are freed you must either:

1. Dispose() the SuperModel object, which also Dispose()s all SuperIC objects.
2. SetAttr() ICA\_TARGET to NULL on every SuperIC object.

SuperIC objects need to clear the ICA\_MAP and ICA\_TARGET settings of it's targetted object. If the target object nolonger exists, expect bad things to happen.

## BUGS

## SEE ALSO

## 1.10 supermodel.class/SM\_SendGlueAttrs

## NAME

SM\_SendGlueAttrs -- Send attributes from GlueFunc (SMA\_GlueFunc)

## SYNOPSIS

```
unknown = SM_SendGlueAttrs(GlueData, TagList)
d0              a0      a1
```

```
ULONG SM_SendGlueAttrs(struct smGlueData *, struct TagItem *);
```

## FUNCTION

This function sends TagList back to the modelclass for notification of other objects.

ONLY to be called from inside a GlueFunction.

## INPUTS

GlueData -  
TagList -

## RESULT

## EXAMPLE

## NOTES

## BUGS

SEE ALSO

## 1.11 supermodel.class/SM\_SICMAP

NAME

SM\_SICMAP -- Builds and super ic class object

SYNOPSIS

```
icobject SM_SICMAPA(Target, MapTags)
a0          a0          a1
```

```
Object *SM_SICMAPA(Object *, struct TagItem *);
```

```
icobject=SM_SICMAP(Target, MapTags, ...)
a0          a0          a1
```

```
Object *SM_SICMAP(Object *, Tag, ...);
```

FUNCTION

Simplified way to create and SuperIC object.  
Target is targeted object of the ic object.  
MapTags, target map tags.

INPUTS

RESULT

EXAMPLE

NOTES

BUGS

SEE ALSO

## 1.12 supermodel.class/SMTAG\_AddTag

NAME

SMTAG\_AddTag -- Add a tag to a taglist.

SYNOPSIS

```
ok = SMTAG_AddTag(TagList, Tag, Data)
```

```
BOOL SMTAG_AddTag(struct TagItem *, ULONG, ULONG);
```

FUNCTION

Add a tag pair to a taglist created with SMTAG\_AllocTags()

INPUTS

TagList - TagList created with SMTAG\_AllocTags()  
Tag - ti\_Tag value  
Data - ti\_Data value

---

**RESULT**

non zero if the tag was added.  
failure can be due to under sized taglist.

**EXAMPLE**

see SMTAG\_AllocTags()

**NOTES**

Don't SMTAG\_AddTag TAG\_IGNORE, TAG\_DONE, TAG\_MORE, TAG\_SKIP.  
This function will only effect the specified TagList, and  
not any other lists referenced by TAG\_MORE.  
This function overwrites existing same tags.

**SEE ALSO**

SMTAG\_AllocTags()

## 1.13 supermodel.class/SMTAG\_AddTags

**NAME**

SMTAG\_AddTags -- Add a taglist to a taglist.

**SYNOPSIS**

ok = SMTAG\_AddTags(TagList, Tag, Data)

BOOL SMTAG\_AddTags(struct TagItem \*, ULONG, ULONG);

**FUNCTION**

Add a taglist to a taglist created with SMTAG\_AllocTags()

**INPUTS**

TagList - TagList created with SMTAG\_AllocTags()  
NewTags - Tags to add to TagList

**RESULT**

non zero if the tag was added.  
failure can be due to under sized taglist.

**EXAMPLE**

see SMTAG\_AllocTags()

**NOTES**

This function will only effect the specified TagList, and  
not any other lists referenced by TAG\_MORE.  
This function overwrites existing same tags.

**SEE ALSO**

SMTAG\_AllocTags()

## 1.14 supermodel.class/SMTAG\_AllocTags

---

## NAME

SMTAG\_AllocTags -- Allocate blank Tag List

## SYNOPSIS

taglist = SMTAG\_AllocTags(TagCount)

struct TagItem \*SMTAG\_AllocTags(ULONG);

## FUNCTION

Allocate tag space for use with other SMTAG\_? functions.

## INPUTS

TagCount - Number of blank tags to allocate.

## RESULT

An empty tag space ending with TAG\_DONE, or NULL.

## EXAMPLE

## NOTES

## BUGS

## SEE ALSO

## 1.15 supermodel.class/SMTAG\_ClearNumTags

## NAME

SMTAG\_ClearTags -- Clear a TagList

## SYNOPSIS

void SMTAG\_ClearTags(TagList, TagCount)

SMTAG\_ClearTags(struct TagItem \*, ULONG);

## FUNCTION

Clears the TagList of all data.

## INPUTS

TagList - Allocated with SMTAG\_AllocTags()

TagCount - Number of blank tags to allocate.

## EXAMPLE

see SMTAG\_AllocTags()

## NOTES

This function is called by SMTAG\_AllocTags(), so the taglist is cleared when allocated.

This function will only effect the specified TagList, and not any other lists referenced by TAG\_MORE.

## BUGS

## SEE ALSO

see SMTAG\_AllocTags()

## 1.16 supermodel.class/SMTAG\_ClearTags

### NAME

SMTAG\_ClearTags -- Clear a TagList

### SYNOPSIS

```
void SMTAG_ClearTags(TagList)

SMTAG_ClearTags(struct TagItem *);
```

### FUNCTION

Clears the TagList of all data.

### INPUTS

TagList - Allocated with SMTAG\_AllocTags()

### EXAMPLE

see SMTAG\_AllocTags()

### NOTES

This function will only effect the specified TagList, and not any other lists referenced by TAG\_MORE.

### BUGS

### SEE ALSO

see SMTAG\_AllocTags()

## 1.17 supermodel.class/SMTAG\_FreeTags

### NAME

SMTAG\_FreeTags -- Clear a TagList

### SYNOPSIS

```
void SMTAG_FreeTags(TagList)

SMTAG_FreeTags(struct TagItem *);
```

### FUNCTION

Frees the TagList.

### INPUTS

TagList - Allocated with SMTAG\_AllocTags().

### EXAMPLE

see SMTAG\_AllocTags()

### SEE ALSO

see SMTAG\_AllocTags()

---

## 1.18 supermodel.class/SMTAG\_RemTag

### NAME

SMTAG\_RemTag -- Remove a tag to a taglist.

### SYNOPSIS

```
ok = SMTAG_RemTag(TagList, Tag)
```

```
BOOL SMTAG_RemTag(struct TagItem *, ULONG);
```

### FUNCTION

Find and remove a tag from a taglist.

### INPUTS

TagList - TagList created with SMTAG\_AllocTags()  
Tag - ti\_Tag value

### RESULT

non zero if the tag was found and removed.

### EXAMPLE

see SMTAG\_AllocTags()

### NOTES

Don't SMTAG\_AddTag TAG\_IGNORE, TAG\_DONE, TAG\_MORE, TAG\_SKIP.  
This function will only effect the specified TagList, and  
not any other lists referenced by TAG\_MORE.

### SEE ALSO

SMTAG\_AllocTags()

## 1.19 supermodel.class/SMTAG\_TagDone

### NAME

SMTAG\_TagDone -- End the TagList with TagDone

### SYNOPSIS

```
void SMTAG_FreeTags(TagList)
```

```
SMTAG_FreeTags(struct TagItem *);
```

### FUNCTION

Ends the taglist with TAG\_MORE and link the list to MoreTags

### INPUTS

TagList - Allocated with SMTAG\_AllocTags().

### EXAMPLE

see SMTAG\_AllocTags()

### SEE ALSO

see SMTAG\_AllocTags()

---

## 1.20 supermodel.class/SMTAG\_TagMore

### NAME

SMTAG\_TagMore -- End the TagList with TagMore

### SYNOPSIS

```
void SMTAG_FreeTags(TagList, MoreTags)

SMTAG_FreeTags(struct TagItem *, struct TagItem *);
```

### FUNCTION

Ends the taglist with TAG\_MORE and link the list to MoreTags

### INPUTS

TagList - Allocated with SMTAG\_AllocTags().  
MoreTags - Tags to link

### EXAMPLE

see SMTAG\_AllocTags()

### SEE ALSO

see SMTAG\_AllocTags()

---