ToolsForHomalg

Tools for the homalg project

Version 2020.10-03

December 2014

Mohamed Barakat

Sebastian Gutsche

Markus Lange-Hegermann

(this manual is still under construction)

This manual is best viewed as an HTML document. The latest version is available ONLINE at:

https://homalg-project.github.io/homalg_project/ToolsForHomalg/doc/chap0.html

An OFFLINE version should be included in the documentation subfolder of the package. This package is part of the homalg-project:

http://homalg.math.rwth-aachen.de/index.php/core-packages/toolsforhomalg
Mohamed Barakat
Email: mohamed.barakat@uni-siegen.de
Homepage: https://mohamed-barakat.github.io
Address: Department of Mathematics, University of Siegen, 57072 Siegen, Germany

Sebastian Gutsche
Email: sebastian.gutsche@rwth-aachen.de
Homepage: http://wwwb.math.rwth-aachen.de/~gutsche/
Address: Lehrstuhl B für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

Markus Lange-Hegermann
Email: markus.lange.hegermann@rwth-aachen.de
Homepage: http://wwwb.math.rwth-aachen.de/~markus/
Address: Lehrstuhl B für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany
Copyright

© 2007-2014 by Mohamed Barakat, Sebastian Gutsche, and Markus Lange-Hegermann

This package may be distributed under the terms and conditions of the GNU Public License Version 2 or (at your option) any later version.
# Contents

1. **Introduction**  
   1.1 What is the role of the *ToolsForHomalg* package in the *homalg* project?  

2. **Installation of the *ToolsForHomalg* Package**  

3. **Caches**  
   3.1 Object constructors  
   3.2 Setters, getters  
   3.3 Managing functions  
   3.4 Install functions  

4. **ToDo-list**  
   4.1 Blueprints  
   4.2 ToDo-list entries  
   4.3 Category  
   4.4 Constructor  
   4.5 This is the magic  
   4.6 Methods for all objects  
   4.7 Proof tracking  
   4.8 Maintainance  

5. **ToolsForHomalg automatic generated documentation**  
   5.1 ToolsForHomalg automatic generated documentation of global variables  

6. **Basics**  
   6.1 Categories  

7. **Pointers**  
   7.1 Weak pointer objects  
   7.2 Pointer objects  

8. **Tools**  
   8.1 Functions  
   8.2 Examplefunctions  

9. **Trees**  
   9.1 Trees  

5  
6  
7  
10  
12  
14  
14  
10  
12  
14  
14  
15  
15  
15  
17  
18  
20  
20  
21  
22  
22  
28  
29
References

Index
Chapter 1

Introduction

1.1 What is the role of the ToolsForHomalg package in the homalg project?

1.1.1 ToolsForHomalg provides ...

The package ToolsForHomalg provides for the homalg project [hpa10]:
Chapter 2

Installation of the ToolsForHomalg Package

To install this package just extract the package’s archive file to the GAP pkg directory.

By default the ToolsForHomalg package is not automatically loaded by GAP when it is installed. You must load the package with

LoadPackage( "ToolsForHomalg" );

before its functions become available.

Please, send me an e-mail if you have any questions, remarks, suggestions, etc. concerning this package. Also, I would be pleased to hear about applications of this package.

Mohamed Barakat
Chapter 3

Caches

3.1 Object constructors

Caches are objects which store for a fixed number of keys a value, so they are a map Obj^k -> Obj, while the k is fixed. A cache usually stores the result in a weak pointer list, which means that if the value which the cache should store is not referenced in the system anymore, it will not be remembered by the cache. However, caches can be set to store the value permanently (crisp), or not to store any new value at all (inaktive). In that case, already stored values are still in the cache and can be accessed once the cache is set active again.

3.1.1 CachingObject

⊿ CachingObject([k, ][is_crisp])
⊿ CachingObject(arg)
⊿ CachingObject(arg1, arg2)

Returns: a cache

If no argument is given, the function returns a weak cache with key length one, if an integer k is given, a weak cache with key length k, and if the bool is_crisp is true, a crisp cache with the corresponding length.

3.1.2 CachingObject (for IsObject, IsObject, IsInt)

⊿ CachingObject(object, cache_name, length[, is_crisp])
⊿ CachingObject(arg1, arg2, arg3, arg4)

This methods are not installed, they serve as an interface for InstallMethodWithCacheFromObject.

3.2 Setters, getters

3.2.1 CacheValue (for IsCachingObject, IsObject)

⊿ CacheValue(cache, key)

Returns: stored value
If there is a value stored in the cache for key, which can be a single key for caches with key
length one or a list of keys depending on the key length of the cache, this method returns a list only
containing the value, otherwise an empty list.

3.2.2 SetCacheValue (for IsCachingObject, IsObject, IsObject)

\[ \text{SetCacheValue}(\text{cache}, \text{key}, \text{value}) \]  
(definition)

Sets the value of key of the cache to value.

3.2.3 IsEqualForCache (for IsObject, IsObject)

\[ \text{IsEqualForCache}(\text{obj1}, \text{obj2}) \]  
(definition)

Returns: true or false
This function is used to compare objects for the caches. The standard way is IsIdenticalObj,
and lists are compared recursive with this function. It is possible and recommended to overload this
function as needed.

3.3 Managing functions

3.3.1 SetCachingObjectCrisp

\[ \text{SetCachingObjectCrisp}(\text{cache}) \]  
(function)

Returns: nothing
Sets the caching to crisp, weak, or deactivates the cache completely.

3.3.2 SetCachingObjectWeak

\[ \text{SetCachingObjectWeak}(\text{arg}) \]  
(function)

3.3.3 DeactivateCachingObject

\[ \text{DeactivateCachingObject}(\text{arg}) \]  
(function)

3.4 Install functions

3.4.1 InstallMethodWithCache

\[ \text{InstallMethodWithCache}(\text{Like}, \text{InstallMethod}) \]  
(function)

Installs a method like InstallMethod, but additionally puts a cache layer around it so that the result
is cached. It is possible to give the cache as the option Cache, to use the same cache for more than one
method or store it somewhere to have access to the cache.
3.4.2 InstallMethodWithCrispCache

▷ InstallMethodWithCrispCache(arg)  (function)

Like InstallMethodWithCache, but with a crisp cache.

3.4.3 InstallMethodWithCacheFromObject

▷ InstallMethodWithCacheFromObject(Like, InstallMethod)  (function)

This works just like InstallMethodWithCache, but it extracts the cache via the CachingObject method from one of its arguments. The CachingObject must then be implemented for one of the arguments, and the option ArgumentNumber can specify which option to be used. As second argument for CachingObject a string is used, which can identify the cache. Standard is the name of the operation, for which the method is installed, but it can be specified using the CacheName option.

3.4.4 FunctionWithCache

▷ FunctionWithCache(func)  (function)

    Returns: a function

    Creates a cached function out of a given function func. If the option Cache is a cache, this cache is used. If the option Cache is the string crisp, a crisp cache is used. All other values for this option lead to a single weak cache.
Chapter 4

ToDo-list

4.1 Blueprints

4.1.1 ToDoListEntryToMaintainEqualAttributes (for IsList, IsList, IsList)

\(\text{ToDoListEntryToMaintainEqualAttributes} (\text{indicator}, \text{objects}, \text{attributes})\) (operation)

Returns: a todo list entry

The first argument is the indicator. It is a list of sources like in ToDoListEntry. Each entry SP has to be a threetuple. First entry of SP has to an object, for which the second entry of SP, which has to be the name of an attribute, must become known. Once the attribute is known to the object, it will be compared to the third entry of the list. This can be a value, which is compared directly a function, which is launched and its return value is compared, or a list, consisting of a function and arguments, so the return value of the function with given arguments is compared. If there is no third entry in SP, it is only looked up if the value is known. Once all entries in indicator are processed like this, and all returned true in the comparator, a list of ToDoListEntryForEqualAttributes is installed. They are installed for the two entries of the list objects which can either be the objects itself or a list containing a function and arguments, which return value is used. For each entry in attributes such an entry is installed. Such an entry can be the name of an attribute, if both objects in objects should share the value between attributes with the same name, or a list of two names, if the attributes do not have the same name.

4.1.2 ToDoListEntryToMaintainFollowingAttributes (for IsList, IsList, IsList)

\(\text{ToDoListEntryToMaintainFollowingAttributes} (\text{indicator}, \text{objects}, \text{attributes})\) (operation)

Returns: a todo list entry

This function creates a ToDoListEntry which can install several ToDoListEntries. The first two arguments, indicator and objects except that there will be only ToDoListEntries installed between the two objects in objects. Each entry in attributes can either be a string which means that the attribute with the given name will be set from the first to the second object in objects once it is known. The third argument attributes is a list of attributes that will be propagated by ToDoListEntries. Each entry TP can either be a list consisting of a DescriptionOfImplication string and one of the following or just one of the following lists: It can be a string, which means that the Attribute with the given name will be propagated from the first to the second object. It can be a list, consisting of
two entries, where the first entry is a list of sources like in ToDoListEntry and the second might be a
function which will be launched once the first part is fulfilled. It can also be a threetuple which will
serve as second to fourth argument of ToDoListEntry. Or it can be a string, which will set the attribute
named like this of the first object to the one named in the second object.

4.1.3 ToDoListEntry (for IsList, IsList)

\[ \text{ToDoListEntry(source, target_list)} \] (operation)

Returns: a todo list entry

This function allows to create more than one ToDoListEntry with identical list of sources at one
time. First argument is a list of sources like in the other ToDoListEntry functions Second argument
is a list of threetuples, which serve as second to fourth argument of ToDoListEntry or a function,
which serves as second argument for ToDoListEntry or a tuple with a description string and one of the
above.

4.1.4 ToDoList_this_object

\[ \text{ToDoList_this_object} \] (global variable)

Represents the objects for which the blueprint is created in the arguments.

4.1.5 ToDoListEntryToMaintainEqualAttributesBlueprint (for IsObject, IsList, Is-
List, IsList)

\[ \text{ToDoListEntryToMaintainEqualAttributesBlueprint(filter, indicator, objects, attributes)} \] (operation)

Returns: nothing

This function installs an immediate method which can install ToDoListEntryToMaintainEqualAt-
tributes. First argument must be a filter, and once the filter becomes true the ToDoListEntryToMain-
tainEqualAttributes is installed with the second to fourth argument as first to third. In those attributes,
at any point, the variable ToDoList_this_object can be used. When the entry is installed this will be
replaced with the object to which the filters became known, i.e. the one which triggered the immediate
method.

4.1.6 ToDoListEntryToMaintainFollowingAttributesBlueprint (for IsObject, IsList, Is-
List, IsList)

\[ \text{ToDoListEntryToMaintainFollowingAttributesBlueprint(arg1, arg2, arg3, arg4)} \] (operation)

Returns: nothing

The same as ToDoListEntryToMaintainEqualAttributesBlueprint for ToDoListEntryToMaintain-
FollowingAttributes.

4.1.7 ToDoListEntryBlueprint (for IsObject, IsList, IsList)

\[ \text{ToDoListEntryBlueprint(arg1, arg2, arg3)} \] (operation)

Returns: nothing

The same as ToDoListEntryToMaintainEqualAttributesBlueprint for ToDoListEntry.
4.2 ToDo-list entries

4.2.1 AddToToDoList (for IsToDoListEntry)

\[\text{AddToToDoList}(E)\] (operation)

Adds the ToDo-list entry \(E\) to the ToDo-lists of it's source objects and creates a new one, if this is needed. This function might be called with lists of entries.

4.2.2 SourcePart (for IsToDoListEntry)

\[\text{SourcePart}(\text{entry})\] (operation)

Returns: a list

Returns the a list of source parts of the ToDo-list entry \(\text{entry}\). This is a triple of an object, a name of a filter/attribute, and a value to which the attribute has to be set to activate the entry.

4.2.3 TargetPart (for IsToDoListEntry)

\[\text{TargetPart}(\text{entry})\] (operation)

Returns: a list

Returns the target part of the ToDo-list entry \(\text{entry}\). This is a triple of an object, a name of a filter/attribute, and a value to which the specific filter/attribute should be set. The third entry of the list might also be a function to which return value the attribute is set.

4.2.4 ProcessAToDoListEntry (for IsToDoListEntry)

\[\text{ProcessAToDoListEntry}(\text{arg})\] (operation)

Returns: a boolean

Processes a ToDo-list entry, i.e. sets the information given in TargetPart if the definitions in SourcePart are fulfilled. Returns a function if the entry could be processed, false if not, and fail if SourcePart or TargetPart weren’t available anymore.

4.2.5 ToDoListEntry (for IsList, IsObject, IsString, IsObject)

\[\text{ToDoListEntry}(\text{arg1}, \text{arg2}, \text{arg3}, \text{arg4})\] (operation)

Returns: a ToDoListEntry

The first argument must be a list consisting of two, three or four-tuples where the first entry must be the object to which the attribute given as a string in the second entry must be known to process this entry. The second entry can also be a list of strings, in that case all the attributes given as names must be known. Also, in this case, only two entries in this tuple are allowed. The third part can be a value or a list, consisting of a function followed by arguments which will be computed by the time the attribute given as second entry becomes known to the first entry. If the second part is only a string, and there is a third entry in the tuple the attribute is compared to the third entry. One can set a comparing function as fourth entry, which must take two entries and return false or true. If the value of the attribute matches the (computed) value in the third entry for all members of the list in the first argument the attribute given as third argument, also by name, of the second argument is set to the value of the fourth argument. This can also be a list which has to be computed, or a function, which return value is used in this case.
4.2.6 ToDoListEntry (for IsList, IsFunction)

\[ \text{ToDoListEntry}( \text{arg1}, \text{arg2} ) \]  
\textbf{Returns:} a ToDoListEntry  
The first argument is a list of three-tuples like above. Once all preconditions become fulfilled the function given as second argument is launched.

4.2.7 SetTargetValueObject (for IsToDoListEntry, IsObject)

\[ \text{SetTargetValueObject}( \text{entry}, \text{value} ) \]  
\textbf{Returns:} nothing  
If the given value of the target part is the return value of a function this command sets the target value of the entry to a function. This is done to keep proof tracking available.

4.2.8 SetTargetObject (for IsToDoListEntry, IsObject)

\[ \text{SetTargetObject}( \text{entry}, \text{obj} ) \]  
\textbf{Returns:} nothing  
If the target object, i.e. the first entry of the target part, was given as a function, this method can set this entry to the return value computed in ProcessToDoListEntry. This happens atomically, do not worry about it.

4.2.9 ToDoListEntryWithContraposition (for IsObject, IsString, IsBool, IsObject, IsString, IsBool)

\[ \text{ToDoListEntryWithContraposition}( \text{sobj}, \text{source\_prop}, \text{sval}, \text{tobj}, \text{target}, \text{tval} ) \]  
\textbf{Returns:} a ToDoListEntry  
Creates a ToDoListEntry which also installs a contraposition. The arguments \text{source\_prop} and \text{target} need to be strings which name a property, and \text{sval} and \text{tval} need to be boolean values, i.e. true or false.

4.2.10 DescriptionOfImplication (for IsToDoListEntry)

\[ \text{DescriptionOfImplication}( \text{arg} ) \]  
\textbf{Returns:} a list  
Has to be set to a string, which describes the reason for the conclusion. If the ToDo-list entry is displayed, the given string will be displayed with a because before it.

4.2.11 ToDoListEntryForEqualAttributes (for IsObject, IsString, IsObject, IsString)

\[ \text{ToDoListEntryForEqualAttributes}( \text{arg1}, \text{arg2}, \text{arg3}, \text{arg4} ) \]  
\textbf{Returns:} a ToDoListEntry  
Creates a ToDoListEntry for two equal attributes, which means that both values of the two attributes will be propagated in both directions.
4.2.12 ToDoListEntryForEquivalentAttributes (for IsObject, IsString, IsObject, IsObject, IsObject, IsObject)

Returns: a ToDoListEntry

Creates a ToDoListEntry for two equivalent attributes, which means that both values of the two attributes will be propagated in both directions. Please note that this one does NOT implement contrapositions.

4.3 Category

4.3.1 IsToDoList (for IsObject)

Returns: true or false

This is the category of ToDo-lists. Every ToDo-list is an object of this category, which basically contains the ToDo-lists.

4.4 Constructor

4.4.1 NewToDoList

Returns: nothing

Creates a new empty ToDo-list.

4.5 This is the magic

4.5.1 Process_A_ToDo_List_Entry

Returns: a boolean

Gets a ToDo-list entry, which is a pair of a list of strings and a weak pointer object, and processes it. If the action was done, it returns true, if not, it returns false, and it returns fail if the action is not possible anymore due to deleted objects.

4.5.2 ProcessToDoList (for IsObject)

Returns: nothing

This is the magic! This attribute is never set. Creating a ToDo-list entry installs an ImmediateMethod for this attribute for the specific category of the object to which ToDo-list is added, and the filter the entry contains. It is then triggert if the filters become applicable, so the ToDo-list is processed.
4.6 Methods for all objects

4.6.1 ToDoList (for IsObject)

\[ \text{ToDoList}(\text{arg}) \]  
\textbf{Returns:} A ToDo-list  
Returns the ToDo-list of an object, or creates a new one.

4.7 Proof tracking

This is a way to track proofs from ToDoLists. Not only for debugging, but also for knowing how things work together.

4.7.1 TraceProof (for IsObject, IsString, IsObject)

\[ \text{TraceProof}(\text{obj}, \text{name}, \text{val}) \]  
\textbf{Returns:} a tree  
If the object \text{obj} has the attribute \text{name}, and its value is \text{val}, and the knowledge has been obtained trough ToDoList-entries, this method traces the way the property was set, and returns a tree which describes the full way of how the attribute became known.

4.8 Maintainance

4.8.1 ActivateToDoList (for IsObject)

\[ \text{ActivateToDoList}(\text{arg}) \]  
\textbf{Returns:} nothing  
This operation activates ToDoLists for the argument.

4.8.2 ActivateToDoList

\[ \text{ActivateToDoList}() \]  
\textbf{Returns:} nothing  
This operation activates ToDoLists for all objects.

4.8.3 DeactivateToDoList (for IsObject)

\[ \text{DeactivateToDoList}(\text{arg}) \]  
\textbf{Returns:} nothing  
This operation deactivates ToDoLists for the argument.

4.8.4 DeactivateToDoList

\[ \text{DeactivateToDoList}() \]  
\textbf{Returns:} nothing  
This operation deactivates ToDoLists for all objects. Note that it is not possible to activate ToDoList for a single object while they are not activated. ToDoListEntries will yet be stored for all
objects that can have ToDoLists. All objects created while ToDoLists are deactivated have by default no ToDoList.

**4.8.5 ActivateWhereInfosInEntries**

\[ \text{ActivateWhereInfosInEntries}(\text{arg}) \]

*Returns:* nothing

Stores the result of Where(100) in an entry if the entry is triggered. This is not activated by default, since it might slow down the system.

**4.8.6 DeactivateWhereInfosInEntries**

\[ \text{DeactivateWhereInfosInEntries}(\text{arg}) \]

*Returns:* nothing

Deactives the storage of the result of Where(100) if an entry is triggered. This is the default.
Chapter 5

ToolsForHomalg automatic generated documentation

5.1 ToolsForHomalg automatic generated documentation of global variables

5.1.1 HOMALGTOOLS

\[ \text{HOMALGTOOLS} \]

A central place for configurations. ChapterInfo Basics, Variables
Chapter 6

Basics

6.1 Categories

6.1.1 IsStructureObjectOrObjectOrMorphism (for IsAttributeStoringRep)

\[ \text{IsStructureObjectOrObjectOrMorphism}(\text{arg}) \]

\textbf{Returns:} true or false

This is the super super GAP-category which will include the GAP-categories IsStructureObjectOrObject and IsHomalgObjectOrMorphism

6.1.2 IsStructureObjectOrObject (for IsStructureObjectOrObjectOrMorphism)

\[ \text{IsStructureObjectOrObject}(\text{arg}) \]

\textbf{Returns:} true or false

This is the super GAP-category which will include the GAP-categories IsHomalgRing, IsHomalgModule, IsHomalgRingOrModule and IsHomalgComplex

6.1.3 IsStructureObject (for IsStructureObjectOrObject)

\[ \text{IsStructureObject}(\text{arg}) \]

\textbf{Returns:} true or false

This is the super GAP-category which will include the GAP-categories IsHomalgRing we need this GAP-category to define things like Hom(M,R) as easy as Hom(M,N) without distinguishing between structure objects (e.g. rings) and objects (e.g. modules)

6.1.4 IsStructureObjectMorphism (for IsAttributeStoringRep)

\[ \text{IsStructureObjectMorphism}(\text{arg}) \]

\textbf{Returns:} true or false

This is the super GAP-category which will include the GAP-categories IsHomalgRingMap, etc.

6.1.5 IsHomalgRingOrModule (for IsStructureObjectOrObject)

\[ \text{IsHomalgRingOrModule}(\text{arg}) \]

\textbf{Returns:} true or false
this is the super GAP-category which will include the GAP-categories IsHomalgRing, IsHomalgModule:
Chapter 7

Pointers

7.1 Weak pointer objects

7.1.1 IsContainerForWeakPointers (for IsComponentObjectRep)

\[\text{IsContainerForWeakPointers(arg)}\]  
\text{(filter)}

\text{Returns: true or false}

The category for weak pointer objects

7.1.2 ContainerForWeakPointers

\[\text{ContainerForWeakPointers(arg)}\]  
\text{(function)}

\text{Returns: a list which can store weak pointers}

The constructor for lists of weak pointers.

7.1.3 UpdateContainerOfWeakPointers (for IsContainerForWeakPointers)

\[\text{UpdateContainerOfWeakPointers(arg)}\]  
\text{(operation)}

Updates the weak pointers in a container and deletes the empty ones

7.1.4 _AddElmWPObj_ForHomalg

\[\text{_AddElmWPObj_ForHomalg(arg)}\]  
\text{(function)}

Adds a weak pointer of an objects to a weak pointer list.

7.1.5 _AddTwoElmWPObj_ForHomalg

\[\text{_AddTwoElmWPObj_ForHomalg(arg)}\]  
\text{(function)}

Adds a weak pointer which depends on two objects to a list of weak pointers
7.1.6  _ElmWPObj_ForHomalg (for IsContainerForWeakPointers, IsObject, IsObject)

▷ _ElmWPObj_ForHomalg(arg1, arg2, arg3) (operation)

Creates a weak pointer depending on two objects and adds it to the container.

7.2  Pointer objects

7.2.1  IsContainerForPointers (for IsComponentObjectRep)

▷ IsContainerForPointers(arg) (filter)

Returns: true or false

The category for pointer objects

7.2.2  ContainerForPointers

▷ ContainerForPointers(arg) (function)

Returns: a container for pointers

Creates a container for pointers.

7.2.3  UpdateContainerOfPointers (for IsContainerForPointers)

▷ UpdateContainerOfPointers(arg) (operation)

Updates the container of pointers, removes old.

7.2.4  _AddElmPObj_ForHomalg

▷ _AddElmPObj_ForHomalg(arg) (function)

Adds a pointer to an object to a container for pointers.

7.2.5  _AddTwoElmPObj_ForHomalg

▷ _AddTwoElmPObj_ForHomalg(arg) (function)

Adds a pointer to two objects to a container for pointers

7.2.6  _ElmPObj_ForHomalg (for IsContainerForPointers, IsObject, IsObject)

▷ _ElmPObj_ForHomalg(arg1, arg2, arg3) (operation)

Returns: an object

Returns an object which a pointer refers to.
Chapter 8

Tools

8.1 Functions

8.1.1 homalgTotalRuntimes

$$\texttt{homalgTotalRuntimes(arg)}$$

*(function)*

**Returns:** an integer

A tool to compute the runtime of several methods.

8.1.2 AddLeftRightLogicalImplicationsForHomalg

$$\texttt{AddLeftRightLogicalImplicationsForHomalg(arg)}$$

*(function)*

A tool to install equivalence between filters.

8.1.3 LogicalImplicationsForOneHomalgObject

$$\texttt{LogicalImplicationsForOneHomalgObject(arg)}$$

*(function)*

Installs a logical implication for one type with all it’s contrapositions.

8.1.4 LogicalImplicationsForTwoHomalgBasicObjects

$$\texttt{LogicalImplicationsForTwoHomalgBasicObjects(arg)}$$

*(function)*

8.1.5 InstallLogicalImplicationsForHomalgBasicObjects

$$\texttt{InstallLogicalImplicationsForHomalgBasicObjects(arg)}$$

*(function)*

8.1.6 LeftRightAttributesForHomalg

$$\texttt{LeftRightAttributesForHomalg(arg)}$$

*(function)*
8.1.7 InstallLeftRightAttributesForHomalg

\[ \text{InstallLeftRightAttributesForHomalg}(\text{arg}) \] (function)

8.1.8 MatchPropertiesAndAttributes

\[ \text{MatchPropertiesAndAttributes}(\text{arg}) \] (function)

A method to match the properties and attributes of two objects.

8.1.9 InstallImmediateMethodToPullPropertyOrAttribute

\[ \text{InstallImmediateMethodToPullPropertyOrAttribute}(\text{arg}) \] (function)

Installs methods to pull new known properties and attributes from one object to another

8.1.10 InstallImmediateMethodToConditionallyPullPropertyOrAttribute

\[ \text{InstallImmediateMethodToConditionallyPullPropertyOrAttribute}(\text{arg}) \] (function)

Installs methods to pull new known properties and attributes under certain conditions from one object to another.

8.1.11 InstallImmediateMethodToPullPropertyOrAttributeWithDifferentName

\[ \text{InstallImmediateMethodToPullPropertyOrAttributeWithDifferentName}(\text{arg}) \] (function)

Installs an immediate method which can pull a property from one object to another with different names.

8.1.12 InstallImmediateMethodToPullPropertiesOrAttributes

\[ \text{InstallImmediateMethodToPullPropertiesOrAttributes}(\text{arg}) \] (function)

Installs an immediate method to pull several properties or attributes from one object to another.

8.1.13 InstallImmediateMethodToPullTrueProperty

\[ \text{InstallImmediateMethodToPullTrueProperty}(\text{arg}) \] (function)

Installs an immediate method to pull a property if it is true.

8.1.14 InstallImmediateMethodToConditionallyPullTrueProperty

\[ \text{InstallImmediateMethodToConditionallyPullTrueProperty}(\text{arg}) \] (function)

Installs an immediate method which conditionally pulls a property if it is true.
8.1.15 InstallImmediateMethodToPullTruePropertyWithDifferentName

InstallImmediateMethodToPullTruePropertyWithDifferentName(arg) (function)

Installs an immediate method which pulls a property with a different name if it is true.

8.1.16 InstallImmediateMethodToPullTrueProperties

InstallImmediateMethodToPullTrueProperties(arg) (function)

Installs an immediate method which pulls several properties if they are true.

8.1.17 InstallImmediateMethodToPullFalseProperty

InstallImmediateMethodToPullFalseProperty(arg) (function)

Installs an immediate method to pull a property if it is false.

8.1.18 InstallImmediateMethodToConditionallyPullFalseProperty

InstallImmediateMethodToConditionallyPullFalseProperty(arg) (function)

Installs an immediate method which conditionally pulls a property if it is false.

8.1.19 InstallImmediateMethodToPullFalsePropertyWithDifferentName

InstallImmediateMethodToPullFalsePropertyWithDifferentName(arg) (function)

Installs an immediate method which pulls a property with a different name if it is false.

8.1.20 InstallImmediateMethodToPullFalseProperties

InstallImmediateMethodToPullFalseProperties(arg) (function)

Installs an immediate method which pulls several properties if they are false.

8.1.21 InstallImmediateMethodToPushPropertyOrAttribute

InstallImmediateMethodToPushPropertyOrAttribute(arg) (function)

Installs an immediate method to push a property from one object to another.

8.1.22 InstallImmediateMethodToConditionallyPushPropertyOrAttribute

InstallImmediateMethodToConditionallyPushPropertyOrAttribute(arg) (function)

Installs an immediate method to conditionally push a property from one object to another.
8.1.23 **InstallImmediateMethodToPushPropertyOrAttributeWithDifferentName**

\( \text{InstallImmediateMethodToPushPropertyOrAttributeWithDifferentName}(\text{arg}) \) (function)

Installs an immediate method which can push a property from one object to another with different names.

8.1.24 **InstallImmediateMethodToPushPropertiesOrAttributes**

\( \text{InstallImmediateMethodToPushPropertiesOrAttributes}(\text{arg}) \) (function)

Installs an immediate method to push several properties or attributes from one object to another.

8.1.25 **InstallImmediateMethodToPushTrueProperty**

\( \text{InstallImmediateMethodToPushTrueProperty}(\text{arg}) \) (function)

Installs an immediate method to push a property if it is true.

8.1.26 **InstallImmediateMethodToPushTruePropertyWithDifferentName**

\( \text{InstallImmediateMethodToPushTruePropertyWithDifferentName}(\text{arg}) \) (function)

Installs an immediate method which pushes a property with a different name if it is true.

8.1.27 **InstallImmediateMethodToPushTrueProperties**

\( \text{InstallImmediateMethodToPushTrueProperties}(\text{arg}) \) (function)

Installs an immediate method which pushes several properties if they are true.

8.1.28 **InstallImmediateMethodToPushFalseProperty**

\( \text{InstallImmediateMethodToPushFalseProperty}(\text{arg}) \) (function)

Installs an immediate method to push a property if it is false.

8.1.29 **InstallImmediateMethodToPushFalsePropertyWithDifferentName**

\( \text{InstallImmediateMethodToPushFalsePropertyWithDifferentName}(\text{arg}) \) (function)

Installs an immediate method which pushes a property with a different name if it is false.

8.1.30 **InstallImmediateMethodToPushFalseProperties**

\( \text{InstallImmediateMethodToPushFalseProperties}(\text{arg}) \) (function)

Installs an immediate method which push several properties if they are false.
8.1.31 DeclareAttributeWithCustomGetter

\[\text{DeclareAttributeWithCustomGetter}(\text{arg})\]

Installs an attribute with a custom getter function.

8.1.32 AppendToAhomalgTable

\[\text{AppendToAhomalgTable}(\text{arg})\]

Appends an entry to a homalg table.

8.1.33 homalgNamesOfComponentsToIntLists

\[\text{homalgNamesOfComponentsToIntLists}(\text{arg})\]

Returns: a list of integers

Creates a list of integers out of the names of components.

8.1.34 IncreaseExistingCounterInObject

\[\text{IncreaseExistingCounterInObject}(\text{arg})\]

Increases an existing counter in an object.

8.1.35 IncreaseExistingCounterInObjectWithTiming

\[\text{IncreaseExistingCounterInObjectWithTiming}(\text{arg})\]

Increases an existing counter on an object with timing.

8.1.36 IncreaseCounterInObject

\[\text{IncreaseCounterInObject}(\text{arg})\]

Increases a counter in an object and creates one if it not exists.

8.1.37 MemoryToString

\[\text{MemoryToString}(\text{arg})\]

Converts the current memory state to a string.

8.1.38 PrimePowerExponent

\[\text{PrimePowerExponent}(n, p)\]

Returns: A nonnegative integer

Returns the \(p\)-exponent of the integer \(n\), where \(p\) is a rational prime.
8.1.39 ViewList (for IsList)

▷ ViewList(L)
   \textbf{Returns:} nothing
   Apply ViewObj to the list L.

8.1.40 homalgLaTeX (for IsObject)

▷ homalgLaTeX(arg)

8.1.41 IdenticalPosition (for IsList, IsObject)

▷ IdenticalPosition(L, o)
   \textbf{Returns:} a positive integer or fail
   Return the position of the object identical to o in the list L.

8.1.42 PositionsOfMaximalObjects (for IsList, IsFunction)

▷ PositionsOfMaximalObjects(L, f)
   \textbf{Returns:} a list
   Return the list of positions of maximal objects in L w.r.t. the partial order defined by the binary function f.

8.1.43 MaximalObjects (for IsList, IsFunction)

▷ MaximalObjects(L, f)
   \textbf{Returns:} a list
   Return the sublist of maximal objects in L w.r.t. the partial order defined by the binary function f.

8.1.44 CollectEntries

▷ CollectEntries(list)
   \textbf{Returns:} a list
   returns a new list that contains for each element elm of the list list a list of length two, the first element of this is elm itself and the second element is the number of times elm appears in list until the next different element. The default comparing function is \(\neq\), which can be changed by passing an optional value to \texttt{ComparingFunction}.

8.1.45 MakeShowable

▷ MakeShowable(mime_types, filter)
   \textbf{(function)}
   Installs a method for IsShowable such that IsShowable( mime_type, object ) returns true for any mime_type in the list mime_types and object in the filter filter.
8.1.46 MakeShowableWithLaTeX

\texttt{MakeShowableWithLaTeX(filter)}

Installs a method for \texttt{IsShowable} such that \texttt{IsShowable( "text/latex", object )} and \texttt{IsShowable( "application/x-latex", object )} return true for an object in the filter \texttt{filter}.

8.2 Examplefunctions

8.2.1 ExamplesForHomalg

\texttt{ExamplesForHomalg()} \\
\textbf{Returns:} true or false \\
Runs the examples for homalg if the package is loadable.

8.2.2 ExamplesForHomalg (for IsInt)

\texttt{ExamplesForHomalg(arg)} \\
\textbf{Returns:} true or false \\
Runs the named example for homalg
Chapter 9

Trees

The trees are used in ToDoLists. They are a technical feature, and fairly general, so they also can be used somewhere else.

9.1 Trees

9.1.1 IsTree (for IsObject)

\[ \text{IsTree}(\text{arg}) \]

(\text{filter})

\text{Returns: true or false}

The category of trees. A tree may have a content, a list of successors, a predecessor and it knows if it is a leave of a tree or not.

9.1.2 Content (for IsTree)

\[ \text{Content}(\text{arg}) \]

(\text{attribute})

\text{Returns: object}

The content of the tree. May be any object.

9.1.3 ListOfSuccessors (for IsTree)

\[ \text{ListOfSuccessors}(\text{arg}) \]

(\text{operation})

\text{Returns: a list of trees}

Returns the list of successors of a tree.

9.1.4 Predecessor (for IsTree)

\[ \text{Predecessor}(\text{arg}) \]

(\text{operation})

\text{Returns: a tree or fail}

Returns the predecessor of a tree, or fail if there is none.

9.1.5 ListOfSentinels (for IsTree)

\[ \text{ListOfSentinels}(\text{arg}) \]

(\text{operation})

\text{Returns: a list}

Returns a list of leaves of the tree.
9.1.6 **RemoveHead** (for IsTree)

```markdown
RemoveHead(arg)

Returns: a tree

Returns the first successor of the tree, and adds all other successors of the tree to the tree that is returned. If the tree is a leave, it returns an empty tree. If the tree is empty, it returns the tree itself.
```

9.1.7 **Tree**

```markdown
Tree()

Returns: a tree

Returns an empty tree.
```

9.1.8 **Tree** (for IsObject)

```markdown
Tree(obj)

Returns: a tree

Returns a tree with argument obj.
```

9.1.9 **Add** (for IsTree, IsTree)

```markdown
Add(tree, new_tree)

Returns: nothing

Adds the [list of] tree[s] new_tree as successor to the tree tree.
```

9.1.10 **ContentListFromSentinelToHead** (for IsTree)

```markdown
ContentListFromSentinelToHead(sent)

Returns: a list

Returns a list of the contents of the trees from the leave sent up to the content of the head of the tree.
```

9.1.11 **PostOrder** (for IsTree)

```markdown
PostOrder(arg)

Returns: a list

Returns the contents of the nodes of the tree in post-order.
```
References

Index

ToolsForHomalg, 5
ActivateToDoList, 15
    for IsObject, 15
ActivateWhereInfosInEntries, 16
Add
    for IsTree, IsTree, 30
AddLeftRightLogicalImplicationsFor- Homalg, 22
AddToDoList
    for IsToDoListEntry, 12
AppendToAhomalgTable, 26
CacheValue
    for IsCachingObject, IsObject, 7
CachingObject, 7
    for IsObject, 7
    for IsObject, IsObject, 7
    for IsObject, IsObject, IsInt, 7
    for IsObject, IsObject, IsInt, IsBool, 7
CollectEntries, 27
ContainerForPointers, 21
ContainerForWeakPointers, 20
Content
    for IsTree, 29
ContentListFromSentinelToHead
    for IsTree, 30
DeactivateCachingObject, 8
DeactivateToDoList, 15
    for IsObject, 15
DeactivateWhereInfosInEntries, 16
DeclareAttributeWithCustomGetter, 26
DescriptionOfImplication
    for IsToDoListEntry, 13
ExamplesForHomalg, 28
    for IsInt, 28
FunctionWithCache, 9
homalgLaTeX
    for IsObject, 27
homalgNamesOfComponentsToIntLists, 26
homalgTotalRuntimes, 22
HOMALG_TOOLS, 17

IdenticalPosition
    for IsList, IsObject, 27
IncreaseCounterInObject, 26
IncreaseExistingCounterInObject, 26
IncreaseExistingCounterInObjectWith- Timing, 26
InstallImmediateMethodToConditionally- PullFalseProperty, 24
InstallImmediateMethodToConditionally- PullPropertyOrAttribute, 23
InstallImmediateMethodToConditionally- PullTrueProperty, 23
InstallImmediateMethodToConditionally- PushPropertyOrAttribute, 24
InstallImmediateMethodToPullFalse- Properties, 24
InstallImmediateMethodToPullFalse- Property, 24
InstallImmediateMethodToPullFalse- PropertyWithDifferentName, 24
InstallImmediateMethodToPull- PropertiesOrAttributes, 23
InstallImmediateMethodToPullProperty- OrAttribute, 23
InstallImmediateMethodToPullProperty- OrAttributeWithDifferentName, 23
InstallImmediateMethodToPullTrue- Properties, 24
InstallImmediateMethodToPullTrue- Property, 23
InstallImmediateMethodToPullTrue- PropertyWithDifferentName, 24
InstallImmediateMethodToPushFalseProperty, 25
InstallImmediateMethodToPushFalseProperty, 25
InstallImmediateMethodToPushFalsePropertyWithDifferentName, 25
InstallImmediateMethodToPushFalsePropertiesOrAttributes, 25
InstallImmediateMethodToPushFalsePropertyOrAttribute, 24
InstallImmediateMethodToPushFalsePropertyOrAttributeWithDifferentName, 25
InstallImmediateMethodToPushTrueProperty, 25
InstallImmediateMethodToPushTrueProperty, 25
InstallImmediateMethodToPushTruePropertyWithDifferentName, 25
InstallLeftRightAttributesForHomalg, 23
InstallLogicalImplicationsForHomalgBasicObjects, 22
InstallMethodWithCache, 8
InstallMethodWithCacheFromObject, 9
InstallMethodWithCrispCache, 9
IsContainerForPointers for IsComponentObjectRep, 21
IsContainerForWeakPointers for IsComponentObjectRep, 20
IsEqualForCache for IsObject, IsObject, 8
IsHomalgRingOrModule for IsStructureObjectOrObject, 18
IsStructureObject for IsStructureObjectOrObject, 18
IsStructureObjectMorphism for IsAttributeStoringRep, 18
IsStructureObjectOrObject for IsStructureObjectOrObjectOrMorphism, 18
IsStructureObjectOrObjectOrMorphism for IsAttributeStoringRep, 18
IsToDoList for IsObject, 14
IsTree for IsObject, 29

LeftRightAttributesForHomalg, 22
ListOfSentinels for IsTree, 29
ListOfSuccessors for IsTree, 29
LogicalImplicationsForOneHomalgObject, 22
LogicalImplicationsForTwoHomalgBasicObjects, 22
MakeShowable, 27
MakeShowableWithLaTeX, 28
MatchPropertiesAndAttributes, 23
MaximalObjects for IsList, IsFunction, 27
MemoryToString, 26
NewToDoList, 14
PositionsOfMaximalObjects for IsList, IsFunction, 27
PostOrder for IsTree, 30
Predecessor for IsTree, 29
PrimePowerExponent, 26
ProcessAToDoListEntry for IsToDoListEntry, 12
ProcessToDoList for IsObject, 14
Process_A_ToDo_List_Entry, 14
RemoveHead for IsTree, 30
SetCacheValue for IsCachingObject, IsObject, IsObject, 8
SetCachingObjectCrisp, 8
SetCachingObjectWeak, 8
SetTargetObject for IsToDoListEntry, IsObject, 13
SetTargetValueObject for IsToDoListEntry, IsObject, 13
SourcePart for IsToDoListEntry, 12
TargetPart for IsToDoListEntry, 12
ToDoList
for IsObject, 15

ToDoListEntry
for IsList, IsFunction, 13
for IsList, IsList, 11
for IsList, IsObject, IsString, IsObject, 12

ToDoListEntryBlueprint
for IsObject, IsList, IsList, 11

ToDoListEntryForEqualAttributes
for IsObject, IsString, IsObject, IsString, 13

ToDoListEntryForEquivalentAttributes
for IsObject, IsString, IsObject, IsObject, IsString, IsObject, 14

ToDoListEntryToMaintainEqualAttributes
for IsList, IsList, IsList, 10

ToDoListEntryToMaintainEqualAttributesBlueprint

ToDoListEntryToMaintainFollowingAttributes
for IsList, IsList, IsList, 10

ToDoListEntryToMaintainFollowingAttributesBlueprint

ToDoListEntryWithContraposition

ToDoList_this_object, 11

TraceProof
for IsObject, IsString, IsObject, 15

Tree, 30
for IsObject, 30

UpdateContainerOfPointers
for IsContainerForPointers, 21

UpdateContainerOfWeakPointers
for IsContainerForWeakPointers, 20

_AddElmPObj_ForHomalg, 21
_AddElmWObj_ForHomalg, 20
_AddTwoElmPObj_ForHomalg, 21
_AddTwoElmWObj_ForHomalg, 20
_ElmPObj_ForHomalg
for IsContainerForPointers, IsObject, IsObject, IsObject, 21

_ElmWObj_ForHomalg
for IsContainerForWeakPointers, IsObject, IsObject, 21

ViewList
for IsList, 27