SMASER

TRIPhighway 3.5 Reference Guide

TRIP
Product Documentation

End User License Agreement

All rights to this software, its documentation and logotypes of the TRIP product family and software (altogether “Software”) supplied by Smaser AG (Smaser) are exclusively owned by Smaser.

The transfer of this Software, solutions or parts thereof requires the prior written agreement of Smaser. Furthermore, the customer has the right to use licensed Software and / or process solutions supplied by Smaser to the extent specified in his contract with Smaser.

The free-to-use non-commercial version doesn’t require a prior written agreement with Smaser but such customers, organizations and/or third parties agree by using the software and / or solution of Smaser to be strongly obliged to keep all rights to this software, documentation and logotypes of the TRIP product family absolutely uninfringed and protected.

Table of Contents

Introduction

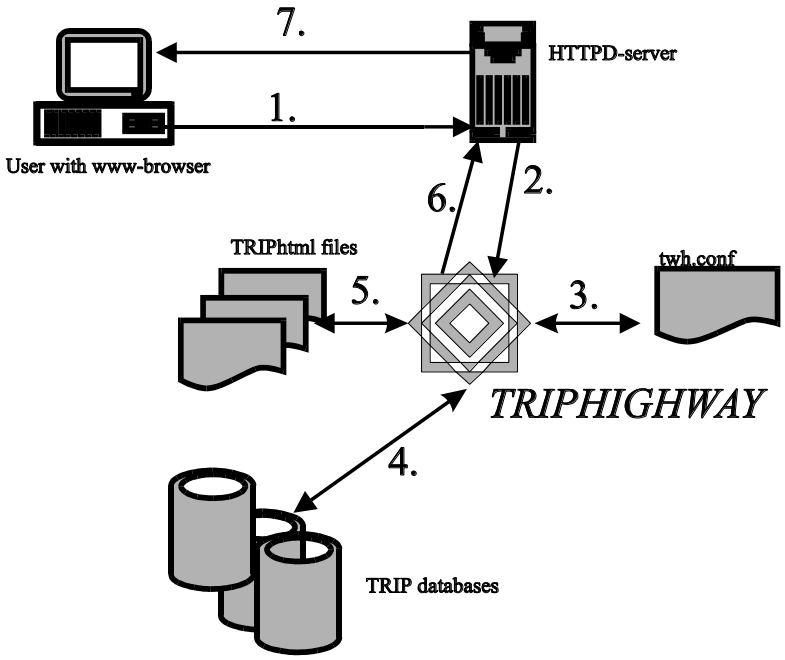
About TRIPhighway

TRIPhighway is a CGI-program. The HTTP server calls TRIPhighway with the parameters supplied through a POST or GET HTTP request - These originate from a simple link or an HTML form. TRIPhighway reads its configuration and TRIPhtml files, performs a database query, replaces the placeholders in TRIPhtml files and sends the result to the client via the HTTP server.

CGI = Common Gateway Interface, an interface for running external programs or gateways under an HTTP server.

TRIPhtml is standard HTML extended with placeholders for information from a TRIP database.

The following picture illustrates a running TRIPhighway system:



An HTTP request is sent from the client to the server. This may be either a GET or a POST request.

The HTTP server starts TRIPhighway and passes it the parameters from the incoming request, along with associated information about the connection.

TRIPhighway reads the configuration file.

TRIPhighway performs the database queries as specified by the parameters passed from the client.

TRIPhighway reads the associated TRIPhtml file and replaces the placeholders with values from the result of the query.

At the same time as the TRIPhtml file is parsed and modified, it is sent to the HTTP server for transportation to the client. The output of TRIPhighway is plain HTML, unless the TRIPhtml file contains non-standard elements.

The HTTP server sends the output from TRIPhighway to the client.

According to the CGI interface transactions are stateless and a single query is regarded as a complete session. Hence, the connection to the TRIP database is initiated and dismantled for each query a client sends.

General functionality

TRIPhighway integrates TRIP databases with the WWW. TRIPhighway does the translation from query result to standard HTML at run time.

TRIPhtml extends standard HTML by adding placeholders for values from the search result from TRIP. In order to maintain compatibility with existing and future features of the HTML language, TRIPhighway does not pose any restrictions on the data surrounding the placeholders.

There are two kinds of variables used in TRIPhighway:

1. Variables that should be applied before the query is performed (passed from the client at each request).
2. Variables that specify how the output is to be produced. These are located in the TRIPhtml files and are also called placeholders.

The first kind of variable gets its values from a TRIP database or the TRIPhighway program. Variables for the query must be set before the query is run. This is usually done with an HTML form or in a <a href=...> tag.

A TRIPhtml file specifies how the search result from the TRIP database is transferred to a WWW-client. In TRIPhtml, standard HTML elements define the layout of your document and TRIPhighway variables give the content.

This approach makes it possible to use any layout in HTML documents. TRIPhtml language gives layout independence in HTML documents by use of standard HTML and TRIP features. It is also possible to use a TRIP output format and to mix TRIPhtml and a TRIP output format in the same HTML document.

Use of TRIPhighway requires good knowledge of TRIP and the HTML language.

TRIPhighway automatically converts Latin-1 (ISO8859-1) characters to SGML entities.

You can use both GET and POST methods with TRIPhighway.

TRIPhighway variables are case insensitive.

TRIPhighway doesn't contain extensive error checking, e.g. there is no verification that <TRIPLOOP> and <TRIPIF> end correctly. All error messages and TRIP messages are transferred to the WWW-client.

Creating TRIPhtml files

It is possible to create the TRIPhtml files using any ordinary text editor or HTML/SGML editor. It is important, however, that the editor being used does not store control information along with the text of the document.

TRIPhtml variable reference

All TRIPhtml variables start with '${' or '<TRIP' and end with '}' or '>'. Note that the variable names are case insensitive.

TRIPhtml variables in alphabetical order

This is an alphabetically sorted list of all TRIPhtml variables.

* - Must be set before a query (usually in STANDARD html)

@ - Must be set before a query but works only inside TRIPhighway

.

${ANDOR}	*	Relation between search fields
${APPL}	*	Name of application
${BACKWARD}	@	URL to the previous page in current search set
${BASE}	*	The name of the TRIP database
${CCL}	*	Any valid CCL command
${CURBASE}		Current database
${CURRID}		Current record number in search set
${CURPART}		Current part record number in search set
${CURRIS}		Current ordinal record number in search set
${DBERROR}		TRIP error message
${DEBUG}	*	Debug flag
${DIHTML}	*	Display TRIPhtml
${DPYTERM}		Display term
${FINDSTR}		Find/display search order
${FORD}	*	Search type
${FORWARD}	@	URL to the next page in current search set
${FREETEXT}	*	Free text search order
${FROMHTML}	@	Same as ${FROMURL}
${FROMURL}	@	Back to previous URL
${FUZZ}	*	Use fuzzy search in query
${HILITE}	*	Highlight option
${HILITEATTR}	*	Highlight attribute
${HTML}	*	TRIPhtml file
${LANGUAGE}	*	TRIP command/messages language
${MATCHES}		Number of hits in search set
${MAXHITS}	*	Maximum number of records to be shown
${MAXPAGE}	*	Maximum number of records to be shown in a page
${NEXTREC}	@	URL to the next record in search set
${OOHTML}	*	TRIPhtml file when one record in search set
${PASSHTML}	@	Same as ${PASSURL}
${PASSURL}	@	Passes URL
${PREVREC}	@	URL to the previous record in search set
${RECORDS}		Number of records in search set
${SAVEHTML}	*	Save html (not TRIPhtml)
${SNHTML}	*	Next html file in case of syntax error or no hits
${SORT}	*	Sort search set
${SURLCUR}	@	Save current URL to urlstack
${SURLDUM}	@	Save current URL to urlstack when there is no search set
${SURLSLI}	@	Save current URL to urlstack
${THES}	*	Thesaurus database
${THLEVEL}		Thesaurus level

${THW}		TRIPhighway
${TRIPCOL}		Maximum number of columns in current output format
${TRIPCUR}	@	Current record in search set
${TRIPCURX}	@	Current record in search set appended to the CGI-script name
${TRIPFORM}		TRIP's output formatter
<TRIPFORM>		TRIP's output formatter
<TRIPIF>		If test structure
<TRIPLOOP>		Loop structure
<TRIPREREAD>		Start again reading records from the beginning of the search set
<TRIPRIS>		Opens current record from search set
${TRIPROW}		Maximum number of rows in current output format
${TRIPSHOW}	*	TRIP output format
${TRIPSLI}	@	URL to current list of records in search set
${TRIPSTRING}	@	Output a TRIP string field
${TRIPVIEW}	@	Output a TRIP string field using TRIPview HTML export
${TRIP-VARIABLE}	@	Field value from database

Detailed descriptions of TRIPhtml variables

${ANDOR} Relation between search fields

Defines AND/OR operator between search fields. TRIPhighway creates a search order using TRIPhighway variables and puts an AND/OR operator between search fields. Default is AND. The actual value must be AND or OR independent of which language used.

Must be set before query.

Example:

Sets OR operator in HTML form

```
<input type="hidden" name="${ANDOR}" value="OR">
```

${APPL} The name of an application

This tag is used to identify the application. In previous versions the application was identified by the ${BASE} tag but if it is be possible to open a selection of databases in a cluster, which is not pre-defined in TRIP, this new tag must be used to identify the application: I.e. To tell TRIPhighway where all its html pages are located. The name of the application must also be inserted in the thw.conf file in the same manner as other databases. Most important here is to specify the thwdir attribute.

If the ${APPL} tag is used, the individual databases in that application do not have to be specified in the thw.conf file if there are no need for special IP address restriction access for a particular database.

${BACKWARD} URL to the previous page in the current search set

${BACKWARD} goes to the previous page in search set. ${BACKWARD} is used for navigation within record lists. See also ${FORWARD}.

Read-only.

Example:

Link to previous page in search set

```
<TRIPIF PREVRIS>
    <a href="${BACKWARD}">Previous</a>
<TRIPIF END>
```

${BASE} The name of the TRIP database

The name of the TRIP database. See also the TRIPhighway configuration file.

Must be set before query.

Example:

Define the database alice

```
<a href="/cgi-bin/thw?${BASE}=alice&${HTML} =liform">
```

${CCL} Any valid CCL command

All CCL commands are not supported. The sort function has it's own TRIPhtml variable.

Please note that the DEFINE REVERSE command must always be written in English.

Must be set before query.

Example:

Show search set in reverse order

```
<a href="/cgi-bin/thw?${BASE}=alice&person=alice&${HTML}=liform&
   ${CCL}=define reverse">
```

${CURBASE} (or ${CURRBASE}) The current database

Read-only in TRIPhtml document.

Example:

Show current database

```
<b>${CURRBASE}</b>
```

The content is e.g. ALICE

${CURBASE.C} The current database cluster

Example:

Show current database cluster

```
<b>${CURBASE.C}</b>
```

The content is e.g. ALL_DBS

${CURPART} Current part record number

Read-only in TRIPhtml document.

Example:

Show current part record number.

```
<b>${CURPART}</b>
```

The content is e.g. 13

${CURRID} (or ${CURRRID}) Current record number in search set

Read-only in TRIPhtml document.

Example:

Show current record in search

```
<b>${CURRID}</b>
```

The content is e.g. 12

${CURRIS} (or ${CURRRIS}) Current ordinal record number in search set

Read-only in TRIPhtml document.

Example:

Show current ordinal record in search

```
<b>${CURRIS}</b>
```

The content is e.g. 1

${DBERROR} TRIP error message

Returns the TRIP error message from the previous search or CCL command.

Read-only in TRIPhtml document.

Example:

Show error message from TRIP

```
<p>${DBERROR}</p>
```

The content is e.g. "Unbalanced parenthesis"

${DEBUG} Debug flag

Debug flag, returns TRIPhtml as plain text to the WWW-client. The debug flag should be the first argument in href.

N.B. This flag will not work if disabled via the NODEBUG setting in the TRIPhighway configuration file thw.conf.

${DEBUG} has two values:

0 = debugging off

non zero = debugging on

Must be set before query.

Example:

Set debug on

```
<a href="${THW}?${DEBUG}=1&${BASE}=alice&${HTML}=liform&${SURLSLI}&
  ${MAXPAGE}=11&${FREETEXT}=${DPYTERM.+}">${DPYTERM}</a>
```

${DIHTML} Display TRIPhtml

The next TRIPhtml document in case of a display query. Needed only when the user can select between FIND and DISPLAY order. If the ${DIHTML} is not defined TRIPhighway uses the ${HTML} document. See also ${DPYTERM}.

Must be set before query.

Example:

Define four different TRIPhtml files.

- ${HTML} for standard search query.
- ${DIHTML} for display query.
- ${OOHTML} for search set with only one record
- ${SNHTML} for syntax error or no hits.

```
<input type="hidden" name="${HTML}" value="liform">
<input type="hidden" name="${DIHTML}" value="display">
<input type="hidden" name="${OOHTML}" value="docu">
<input type="hidden" name="${SNHTML}" value="nosyn">
```

${DPYTERM} Display term

Current display term in TRIP search set. Valid only with display queries.

Read-only in TRIPhtml file.

Example:

Display term

```
<p>${DPYTERM}</p>
```

The content is e.g. ALICE

${FINDSTR} Find/display search order

Last FIND/DISPLAY order (as parsed and executed in TRIP).

Read-only.

Example:

Show last search order when there are no hits

```
<h3>Sorry no hits: ${FINDSTR}</h3>
```

${FINDSTR} is the last find/display order parsed by TRIP, e.g. (person=alice)

${FORD} Search type

Find, Fuzz, Display or Dummy search (default is find). Dummy search is useful within the TRIPhighway navigation mechanism where there is no actual database search.

Must be set before query.

Example:

Set display type query

```
<input type="hidden" name="${FORD}" value="display">
```

${FORWARD} URL to the next page in current search set

${FORWARD} goes to the next page in the current search set. ${FORWARD} is used for navigation within record lists. See also ${BACKWARD}.

Read-only.

Example:

URL to the next page in the current search set

```
<TRIPIF MORERIS>
    <a href="${FORWARD}">continue</a>
<TRIPIF END>
```

${FREETEXT} Free text search order

Freetext search, e.g. any valid CCL find/display command. With ${FREETEXT} it is possible to do more complicated searches, such as

searching in all fields of type TEXT and PHRASE.

complete CCL search orders.

Must be set before query.

Example:

User input for any CCL command

```
<input name=”${FREETEXT}” size="40">
```

${FROMURL} Back to previous URL

Address to previous URL, which can be any of

TRIPhtml document saved by ${SURLCUR}

TRIPhtml document saved by ${SAVEHTML}

TRIPhtml document saved by ${SURLSLI}

Passed by ${PASSURL}.

Modifier: ${FROMURL.X} where X is a positive or negative number.

if X > 0 then back to top of urlstack.

if X = 0 same as ${FROMURL} without modifier

if X < 0 then back to bottom of urlstack.

Read-only.

Example:

Back to up

```
<a href="${FROMURL}"><img src= "/thwex/pict/up.gif" alt="up"></a>
```

${FUZZ} Use fuzzy search in query

${FUZZ} has two values:

= Do NOT use fuzzy search.

other values = Use fuzzy search.

Must be set before query.

Example:

Set fuzzy search to be used in search orders.

```
<input type="radio" name="${FUZZ}" value="1">
```

${HILITE} Highlight option

This tag is used to tell TRIPhighway what highlighting options should be used. It must be set before the search is made, i.e. in the URL specifying the call. The possible values are:

1. No Highlight
2. Highlight all hits using the specified highlight attribute or the default attribute bold.
3. Highlight all hits using a built-in href construct that makes navigation between hits possible.

All records found by a search will by default have all term hits highlighted using the bold attribute. To avoid highlighting, the tag ${HILITE} can be used to switch it off.

${HILITEATTR} Highlight attribute

Specifies the highlight attribute. Default is bold, <b>.

Example:

```
<font color="orangered" face="helvetica">
```

${HTML} TRIPhtml file (default)

The next TRIPhtml document. The TRIPhtml document may also be given with ${TRIPSHOW}. If both ${HTML} and ${TRIPSHOW} are given, the TRIPhtml document in ${TRIPSHOW} is used.

Must be set before query.

Example:

Define default TRIPhtml to file.

```
<input type="hidden" name="${HTML}" value="liform">
<input type="hidden" name="${DIHTML}" value="display">
<input type="hidden" name="${OOHTML}" value="docu">
<input type="hidden" name="${SNHTML}" value="nosyn">
```

${LANGUAGE} TRIP command language

Specifies the language for TRIP if another language than that specified in the thw.conf file or the TRIPsystem default is to be used.

${MATCHES} Number of hits in search set

Returns the number of hits in records in the last search or display query.

Read-only.

Example:

Show number of hits

```
<td align="center">${MATCHES}</td>
```

The returned value might be, for example, 100

${MAXHITS} Maximum number of records to be shown

Overrides default value from configuration file.

Must be set before query.

Example:

Override default and configuration value of maximum records to be shown.

```
<input type="hidden" name="${MAXHITS}" value="200">
```

${MAXPAGE} Maximum number of records to be shown in a page

Note that ${MAXPAGE} must be one greater than the number of records in a page.

Must be set before query.

Example:

Set a page to contain maximum 10 records.

```
<input type="hidden" name="${MAXPAGE}" value="11">
```

${NEXTREC} URL to the next record in search set

Link to the next record in the search set. ${NEXTREC} uses the current TRIPhtml document.

Read-only.

Example:

URL to the next record in the search set.

```
<TRIPIF NLAST>
    <a href="${NEXTREC}"><img src="/thwex/pict/next.gif" alt="next"></a>
<TRIPIF END>
```

${OOHTML} TRIPhtml file in case of just one record in search set

The next TRIPhtml document to use when only ONE record is found. If there is only one record hit, it might be better to show the whole record than a list of documents of the search set.

Must be set before query.

Example:

Define TRIPhtml form for search set with only one hit record.

```
<input type="hidden" name="${OOHTML}" value="docu">
```

${PASSURL} Passes URL

Passes the URL stack for later use. The stack contents is previously saved by ${SURLCUR}, ${SURLSLI} or ${SAVEHTML}.

Read-only.

Example:

Pass URL

```
<a href="${THW}${BASE}=alice&${HTML}= format&${PASSURL}">
```

${PREVREC} URL to the previous record in search set

Previous record in the search set. The current TRIPhtml document is used.

Read-only.

Example:

URL to previous record in the current search set.

```
<TRIPIF NFIRST>
    <a href="${PREVREC}”><img src="/thwex/pict/prev.gif" alt="prev"></a>
<TRIPIF END>
```

${RECORDS} Number of records in search set

Number of records in the last search or number of terms in the last display query.

Read-only.

Example:

```
<h3>Found ${RECORDS} Chapters </h3>
```

${SAVEHTML} Save html (not TRIPhtml)

Saves current URL (any other than TRIPhtml) to the urlstack for returning back later to this URL. Use ${SURLCUR} or ${SURLSLI} for TRIPhtml files.

Must be set before query.

Example:

Save standard html to the urlstack

```
<input type="hidden" name="${SAVEHTML}" value="/thwex/alice/thw”>
```

${SNHTML} Next html file in case of syntax error, no hits or any defined limit reached

Defines next html form if there is a syntax error in the search order, if there are no hits in the search or if any max limit is reached (e.g. sort max). If ${SNHTML} is not defined, the default html page is used. The default html page displays an error message.

Must be set before query.

Example:

Define html form for syntax error or no hits

```
<input type="hidden" name="${SNHTML}" value="nosyn">
```

${SORT} Sort search set

Sort the search set. The sort order syntax is the same as in the TRIP CCL order SHOW SORT.

Must be set before query.

Example:

Sort by person and chapter

```
<input type="radio" name="${SORT}" value="person,chapter">
```

${SURLCUR} Save current URL to urlstack

Save this URL to the urlstack inside TRIPhighway and refer to the current record. You can come back later to this URL using ${FROMURL}. See ${SURLSLI} for the difference between ${SURLCUR} and ${SURLSLI}

Read-only.

Example:

Save this page to urlstack for later use

```
<a href="${THW}?${TRIPCUR}&${HTML}=format&${SURLCUR}">${chapter}</a>
```

${SURLDUM} Save current URL to urlstack when there is no search set

Save this URL to the urlstack inside TRIPhighway when no search has been set in TRIP.

${SURLSLI} Save current URL to urlstack

Save this URL to the urlstack inside TRIPhighway and refer to the whole search list. You can come back later to this URL using ${FROMURL}. The difference between ${SURLCUR} and ${SURLSLI} is that ${SURLSLI} points to the start of the current search set and ${SURLCUR} points to the current record in the search set.

Read-only.

Example:

Save this page to urlstack for later use

```
<a href="${THW}?${TRIPCUR}&${HTML}=format&${SURLSLI}">${chapter}</a>
```

${THES} The thesaurus database name

Must be set before query.

Example:

Set thesaurus to thesali

```
<input type="hidden" name="${THES}" value="thesali">
```

${THLEVEL} Thesaurus level

Indent the thesaurus term to show the level of the term.

Read-only.

Example:

Show the thesaurus field ctx and its level

```
<p>${THLEVEL} ${ctx}</p>
```

${THW} TRIPhighway

The TRIPhighway symbol. Used instead of a hard link to TRIPhighway.

Read-only.

Example:

Symbol containing the TRIPhighway CGI-script name

```
<a href="${THW}?${TRIPCUR}&${HTML}= format&${SURLSLI}">${chapter}</a>
```

${TRIPCOL} Maximum number of columns in current output format

The maximum number of columns from TRIP formatted output can be specified using ${TRIPCOL}. The default value is 80 columns. The TRIP format page size is specified by using ${TRIPCOL} and ${TRIPROW} together. The product of ${TRIPCOL} and ${TRIPROW} must not exceed 8192.

Read-only.

Example:

Set TRIP format page size to 25 rows and 100 columns

```
<pre>${TRIPROW}=25 ${TRIPCOL}=100</pre>
```

${TRIPCUR} Current record in search set

Passes TRIPhighway information to another TRIPhtml document. ${TRIPCUR} contains the URL to the current record of search set.

Read-only.

Example:

Save this search set and current record information for later use

```
<a href="${THW}?${TRIPCUR}&${HTML}= format&${SURLSLI}">${chapter}</a>
```

${TRIPCURX} Current record in search set appended to the CGI-script name

Same as $THW?${TRIPCUR}.

See also ${TRIPCUR}

Example:

```
<a href="${TRIPCURX}&${HTML}= format&${SURLSLI}">${chapter}</a>
```

${TRIPFORM} TRIP's output formatter

Shows the search result using a TRIP output format. The TRIP output format is defined with the ${TRIPSHOW} variable. If the TRIP output format is omitted, TRIPhighway uses the default output format of the database. ${TRIPFORM} shows the search result record by record while <TRIPFORM> shows the whole search set. When using ${TRIPFORM} it is possible to mix TRIPhtml and a TRIP output format, i.e. you can have html elements in the output format as well.

Read-only.

Example:

Show current record with TRIP's output format

```
<pre>${TRIPFORM}</pre>
```

It is also possible to specify an output format with TRIPFORM:

Example:

Show current record with the specified output format FORM1

```
<pre>${TRIPFORM.FORM1}</pre>
```

<TRIPFORM> TRIP's output formatter

Uses TRIP's output formatter for displaying search results. See also ${TRIPFORM} to find the difference between ${TRIPFORM} and <TRIPFORM>.

Example:

Show the whole search set using TRIP's output format

```
<pre><TRIPFORM></pre>
```

<TRIPIF> If test structure

There are a several types of <TRIPIF> elements. A brief summary (IFTYPE and explanation) is shown in the table, while there is a better explanation in the examples following.

Syntax of <TRIPIF>:

<TRIPIF [IFTYPE=] END | FPART | ISEMPTY | MORERIS | MORETONE | NEMPTY | NFIRST | NLAST | NOHITS | ONLYONE | PREVRIS | SORTMAX | SYNERR | SYNTAX>

<TRIPIF> cannot be nested e.g. only one <TRIPIF> is in action at one time.

END	Ends current <TRIPIF> element
FPART	True when first record part
LPART	True when last record part
MORERIS	True when not last record in current search set
MORETONE	True when more than one record in current search set
NEMPTY	True when field is not empty
NFPART	True when not first record part
NLPART	True when not last record part
NOHITS	True when no hits in last search order
ONLYONE	True when only one record in current search set

PREVRIS	True when not first record in current search set
SORTMAX	True when sort max reached in last search order
SYNERR	True when syntax errors or max limits in last search order
SYNTAX	True when syntax errors in last search order

<TRIPIF END>

Ends current <TRIPIF> element. See examples below.

<TRIPIF FPART>

True when current record part is the first one (used in <TRIPLOOP PART>).

Example:

Show header only at first record part

```
<TRIPIF FPART>
    <hr><h3>Arguments:</h3>
<TRIPIF END>
```

<TRIPIF ISEMPTY>

True when field is empty.

Example:

Show something when field rtx is empty

```
<TRIPIF ISEMPTY>
    ${rtx.#0}${rtx}<br>
<TRIPIF END>
```

<TRIPIF MORETONE>

True when more than one record in current search set.

Example:

Return back to list

```
<TRIPIF MORETONE>
    <a href="${FROMURL}"><img src="/thwex/pict/up.gif" alt="up"></a>
<TRIPIF END>
```

<TRIPIF NEMPTY>

True when the field is NOT empty.

Example:

Show something when field rtx is non-empty

```
<TRIPIF NEMPTY>
    ${rtx.#0}${rtx}<br>
<TRIPIF END>
```

<TRIPIF NFIRST>

True, when the current record is not the first one in the search set.

Example:

Test if the record is not the first one in the search set

```
<TRIPIF NFIRST>
    <a href="${PREVREC}"><img src="/thwex/pict/prev.gif" alt="prev"></a>
<TRIPIF END>
```

<TRIPIF NLAST>

True when the current record is not the last one in the search set.

Example:

Test if the record is not last in search set

```
<TRIPIF NLAST>
    <a href="${NEXTREC}"><img src="/thwex/pict/next.gif" alt="next"></a>
<TRIPIF END>
```

<TRIPIF NOHITS>

True when there are no hits in the current search.

Example:

Show the last search order if there were no hits.

```
<TRIPIF NOHITS>
    <h3>Sorry no hits: ${FINDSTR}</h3>
<TRIPIF END>
```

<TRIPIF ONLYONE>

True when there is only one record in the current search set.

Example:

```
<TRIPIF ONEONLY>
    <a href="${FROMURL}"><img src="/pict/home.gif  alt=home"></a>
<TRIPIF END>
```

<TRIPIF SORTMAX>

True when the defined sort max is reached in the last search order. If this element is used to select sort limit errors, syntax errors should be selected using the SYNTAX element. If SYNERR is used, sort max error messages might come out doubled as both SORTMAX and SYNERR will be activated.

Example:

Show last search order and error message if sort max is reached in the order.

```
<TRIPIF SORTMAX>
    <h3>Sort max reached: ${FINDSTR}<br>${DBERROR}</h3>
<TRIPIF END>
```

<TRIPIF SYNERR>

True when there were syntax errors or max limits reached in the last search order. Use this element only if SORTMAX is not used and not together with the SYNTAX element.

Example:

Show last search order and error message if there were syntax/limit error in the order.

```
<TRIPIF SYNERR>
    <h3>Syntax/limit error: ${FINDSTR}<br>${DBERROR}</h3>
<TRIPIF END>
```

<TRIPIF SYNTAX>

True when there were syntax errors in the last search order. Use this element only if SORTMAX is used and not together with the SYNERR element.

Example:

Show last search order and error message if there were syntax error in the order.

```
<TRIPIF SYNTAX>
    <h3>Syntax error: ${FINDSTR}<br>${DBERROR}</h3>
<TRIPIF END>
```

<TRIPLOOP> Loop structure

There are a several types of <TRIPLOOP> elements. A brief summary (LOOPTYPE and explanation) is shown in the table, while there is a better explanation in the examples following.

TRIPLOOPs can be nested. Loops must occur in the correct order (ris, part, paragraph, item). There is no error checking if <TRIPLOOP> doesn't end correctly. All <TRIPLOOP> elements end at the end of the TRIPhtml file.

Syntax of <TRIPLOOP>:

<TRIPLOOP [LOOPTYPE=] RIS | PART | HITPART | PARA | ITEM [SINGLE | MULTI] | END>

RIS	Starts record in search loop
PART	Starts part loop
HITPART	Starts hit part loop
PARA	Starts paragraph loop
ITEM	Starts item loop
END	Ends current <TRIPLOOP> element

<TRIPLOOP RIS>

Starts record in search loop.

Example:

List the field ”person” from every record in the search set

```
<TRIPLOOP RIS>
    ${person}<br>
<TRIPLOOP END>
```

<TRIPLOOP PART>

Starts record part loop.

Example:

```
<dl>
<TRIPLOOP PART>
   <dt><b>Name:</b> ${arg_name.2}</dt>
    <dt><b>Type:</b> ${arg_type.2}</dt>
<TRIPLOOP END>
</dl>
```

<TRIPLOOP HITPART>

Starts record hit part loop, i.e. load data from record parts that have been hit by a previous search.

Example:

```
<dl>
<TRIPLOOP HITPART>
    <dt><b>Name:</b> ${arg_name.2}</dt>
    <dt><b>Type:</b> ${arg_type.2}</dt>
<TRIPLOOP END>
</dl>
```

<TRIPLOOP PARA>

Starts paragraph loop. The field should be a text-field otherwise the loop is ignored.

Example:

```
<TRIPLOOP PARA>
    <TRIPLOOP ITEM>
        <dd>
            <p>${c_example}</p>
        </dd>
    <TRIPLOOP END>
<TRIPLOOP END>
```

<TRIPLOOP ITEM [SINGLE | MULTI]>

Starts subfield (item) loop. The difference between MULTI and SINGLE is that SINGLE loops through all fields and MULTI only through fields that have modifier .C. The default value is SINGLE.

Example:

Show all person's in different lines

```
<p>
<TRIPLOOP ITEM>
    ${person}<br>
<TRIPLOOP END>
</p>
```

<TRIPLOOP END>

Ends current <TRIPLOOP> element. See examples above.

<TRIPRIS> Opens current record from search set

Opens current record from search set.

Example:

```
<TRIPRIS>
```

${TRIPROW} Maximum number of rows in current output format

Number of rows in TRIP formatted output. The default value is 50 rows.

For description and examples, please see ${TRIPCOL}.

${TRIPSHOW} TRIP output format

Show command arguments for output. Used together with ${TRIPFORM} or <TRIPFORM>.

Must be set before query.

Example:

Output format

```
<select name="${TRIPSHOW}">
  <option selected="selected" value="html=list oohtml=docu">as html</option>
  <option value="html=trip oohtml=record">default output format</option>
  <option value="format=special html=oform">special output format</option>
</select>
```

${TRIPSLI} URL to current list of records in search set

${TRIPSLI} is the URL to the first record in search set.

Read-only.

See ${TRIPCUR} for example.

${TRIPSTRING} Output a TRIP string field

Returns binary data from the string field. The value of ${TRIPSTRING} is the name of the string field and the extension of the field content. The syntax is

${TRIPSTRING}=” [part.]fieldname[.filetype][.mimetype]”

The default value for filetype is ”GIF”. If part is included, the string field of this record part is picked up. Please note that ${TRIPSTRING} must be called with ${THW} (see ${THW}) due to the current httpd implementation.

Must be set before query.

Example:

Get GIF-picture from TRIP database field blob

```
<img src="${THW}? ${TRIPSTRING}=”blob.gif">
```

${TRIPVIEW} Output a TRIP string field using TRIPview HTML export

The value of ${TRIPVIEW} is the name of the string field. The syntax is

${TRIPVIEW}=” [part.]string_fieldname[.text_fieldname]

If the text_fieldname is given and the TRIPview version in use if of version 2 or later, the html output will contain hit term highlighting.

${TRIP-VARIABLE} Field value from database

${TRIP-VARIABLE} has two meanings and syntaxes:

1. Show field value.
2. Give search order to a field

In search forms, ${TRIP-VARIABLE} is given without modifier, but in TRIPhtml the ${TRIP-VARIABLE} syntax is

${[part.]fieldname}

where part is the record part number and fieldname is any valid field name.

Field variable modifiers:

${fieldname}	Text fields: Inside <TRIPLOOP PARA>, show the current paragraph. Outside <TRIPLOOP PARA>, show the first paragraph. Non-text fields: Inside <TRIPLOOP ITEM>, show the current item. Outside <TRIPLOOP ITEM>, show the whole field.
${fieldname.*}	Text fields: Show the field with all paragraphs. Non-text fields: Same as ${xxxx}.
${fieldname.n}	Text fields: Show the nth paragraph. Non-text fields: Show the nth item.
${fieldname.#n}	Show the first n characters.
${fieldname.*.#n}	Text fields: Show the first n characters in current paragraph Non-text fields: Invalid.
${fieldname.+}	convert white space to +-char (needed when using a TRIP field content in a href statement, as the field content may have white spaces).
${fieldname.c}	Show current item in <TRIPLOOP ITEM MULTI>.
${fieldname.L}	Show last item.
${fieldname.%n}	Show right adjusted within n characters, filled from left.
${fieldname.Bc}	Output break at character c.

${fieldname.H}	Adds “http://” at the beginning of the value if not already present.
${fieldname.S}	Show a summary of the field content. This works for TEXT fields only and requires the field to be non-Boolean indexed.
${fieldname.Tn}	Show a KWIC (Keyword-In-Context) “teaser” of the field. The *n* argument specifies the maximum size of the output.
${part.fieldname}	Show the field within current record part.

It is also possible to combine two or more modifiers.

TRIP-VARIABLE is read-only.

TRIP-VARIABLE Must be set before, or after, a query.

Examples:

Before query:

```
<form action="/cgi-bin/thw" method="get"> <h3>Search:</h3>
    Person:<input name="person" size="40">
        ...
</form>
<a href="/cgi-bin/thw?${BASE}=alice&${HTML}=liform&${MAXPAGE}=11&
    ${SAVEHTML}=/thwex/alice/index.html&person=alice"
```

After query:

1. Show one field (all types)

```
<p>${c_example}</p>
```

2. Show item 2 in a phrase field

```
<dt><b>Access:</b> ${arg_access.2}</dt>
```

3. Show paragraph 2 and all items in a text field

```
<dt><b>Description:</b>${arg_description.2.*}</dt>
```

4. Convert white spaces to +

```
<a href="${THW}?${MAXPAGE}=11&${HTML}=liform&
${SNHTML}=nosyn&${BASE}=alice&
${FREETEXT}=${ctx.+}&${SURLSLI}">
<b>${ctx}</b>
```

5. Show current item in a multi item loop

```
<TRIPLOOP ITEM MULTI>
    ${field_one.c}
    ${field_two.c}
<TRIPLOOP END>
```

6. Show right adjusted within 7 characters.   filled from left

```
${field_one.%7}
```

7. Show maximum 7 characters

```
${field_one.#7}
```

8. Show last item in a phrase field

```
${field_one.L}
```

9. Break output at character R

```
${field_one.BR}
```

10. Convert white spaces to + and show maximum 7 characters

```
${field_example.+.#7}
```

11. Show item 2 in record part 3 in field arg_access

```
<dt><b>Access:</b> ${3.arg_access.2}</dt>
```