JET Extension Tags

Bradford Matthew Arant Sr.

November 20, 2026

Contents

1 Introduction **5**
- 1.1 Use Cases . . . 5
  - 1.1.1 World Wide Web Site . . . 5
  - 1.1.2 API Handling . . . 5
  - 1.1.3 Asterisk Dial Plan Generator Tool . . . 5

2 Tags and Attributes **7**

3 User Defined Tags and Tag Libraries **9**

4 Variables and Variable Types **11**
- 4.1 Global Variables . . . 11
- 4.2 Local Variables . . . 12
- 4.3 Session Variables . . . 12
- 4.4 Keyword Variables . . . 12
- 4.5 Environment Variables . . . 12
- 4.6 CGI Variables . . . 12

5 Expressions **15**
- 5.1 Operators . . . 15
  - 5.1.1 Arithmetic Operators . . . 15
  - 5.1.2 Boolean Operators . . . 15
- 5.2 Function Reference . . . 15
- 5.3 Date Functions . . . 15
  - 5.3.1 datediff . . . 15
  - 5.3.2 unixtime . . . 15
- 5.4 Math Functions . . . 15
  - 5.4.1 abs . . . 15
  - 5.4.2 acos . . . 15
  - 5.4.3 asin . . . 16
  - 5.4.4 atan . . . 16
  - 5.4.5 cos . . . 16
  - 5.4.6 floor . . . 16
  - 5.4.7 ceil . . . 16

5.4.8 max . . . 16
5.4.9 min . . . 16
5.4.10 pow . . . 16
5.4.11 random . . . 16
5.4.12 sin . . . 16
5.5 String Functions . . . 16
5.5.1 concat . . . 16
5.5.2 integer . . . 16
5.5.3 left . . . 16
5.5.4 ltrim . . . 17
5.5.5 reverse . . . 17
5.5.6 right . . . 17
5.5.7 round . . . 17
5.5.8 rtrim . . . 17
5.5.9 substring . . . 17
5.5.10 tolower . . . 17
5.5.11 toupper . . . 17
5.5.12 trim . . . 17
5.5.13 expr . . . 17

6 Common Gateway Interface Features 19
6.1 Session Control . . . 19

7 Tag Reference 21
7.1 bind . . . 21
7.2 call . . . 22
7.3 comment . . . 23
7.4 cookie . . . 23
7.5 dump . . . 23
7.6 exclude . . . 23
7.7 expr . . . 23
7.8 for . . . 23
7.9 header . . . 24
7.10 if/else . . . 24
7.11 ifrow/else . . . 24
7.12 include . . . 24
7.13 jet . . . 24
7.14 mysql . . . 25
7.15 read . . . 25
7.16 set . . . 25
7.17 sql . . . 26
7.18 stream . . . 26
7.19 system . . . 26
7.20 tag . . . 26
7.21 until . . . 27
7.22 while . . . 27

7.23 whiledir 27
7.24 whilerow 27
7.25 write 27

Chapter 1

Introduction

JET will pass through the untagged areas to the output without any modifications. Data contained within the tags may modify their containers before placing any output. The space taken by the tag itself is not passed to the output and will not appear in the output.

Skip Blank Lines options on containers will skip passing any blank lines or line containing only whitespace to the output.

1.1 Use Cases

1.1.1 World Wide Web Site

1.1.2 API Handling

1.1.3 Asterisk Dial Plan Generator Tool

Chapter 2

Tags and Attributes

XML style tagging is used to insert functions and other behaviors into the script. As the document contents are output a tag may be inserted to perform the desired function and affect the output.

Tag attributes are used to pass parameters into the operations and functionality for each tag.

Outputting a database from mysql may be performed using the <mysql tag in combination with the <ifrow , <whilerow and <sql tags.

Chapter 3

User Defined Tags and Tag Libraries

You can define your own tag definitions and use them within your JET scripts.

Keywords are not defined as part of the tag definition and any attribute added to the implementation of the tag is passed into the tag process as a local variable for that container.

Tag definitions can also be defined as scoped or unscoped, as follows:

scoped - local variables using the #[n] format can be scoped to just your tag contents. Access to local variables where your tag is being used must use the 'parent' scope option to retrieve variables from the parent.

unscoped - local variables are shared with the parent tag environment and are not unique to your tag definition.

A <container /> tag can be inserted into the container and will take the contents of the container at runtime and insert it into the output when processing the user defined tag definition. The container tag of the the <container /> is processed before the container containing the <container /> so that local variabls can be set and processed that will affect the tags overall output.

Tags are many times nested with tags running within the container of a higher tag.

Chapter 4

Variables and Variable Types

Variables are used to contain dynamic content values and can be sourced from several locations and limited in scope depending on use. Some variable types can be set to reflect script state or are read only from other outside sources of data.

Jet tags can be either scoped or unscoped.

The following is a list of variable types and a brief description of their source:

- Global - values can be initialized using the set tag and specifying the scope of global.
- Local -
- Keyword - within a tag's container we may access the keyword values specified on the containing tag.
- Session - values stored in the session file with the set tag and a scope of session. They persist across requests from the same browser.
- Environment -
- CGI Variables -

4.1 Global Variables

Global variables are available to all logic once the values have been set.

Global variables are accessed throughout the procedure with the $[name] syntax.

Global is the default if the scope is not specified for any function that writes to a variable.

4.2 Local Variables

Local variables are established using the set tag and specifying a scope of local. Local variables are typically only available within the container where they are defined. Some containers may not be capable of storing local variables (i.e. set tag) and are inheriting the local scope from their parent tag.

4.3 Session Variables

Session variables are stored in the session file for the current session and remain available to later requests from the same browser. Session control must be enabled (see Session Control).

A session variable is written with the set tag and a scope of session:

```
<set name="_login_id" value="$[login_id]" scope="session">
```

The name can be any variable name. It does not need an @ prefix. Session variables are read with the $[@name] syntax, for example $[@_login_id].

Setting a session variable does not also set a global variable of the same name. When the same request needs to read the value as $[name], set it with both scopes:

```
<set name="_login_id" value="$[login_id]">
<set name="_login_id" value="$[login_id]" scope="session">
```

4.4 Keyword Variables

Keyword variables are set when they are specified on a containing tag. Within the container you can access the variable using the syntax $[%name].

The variable retrieved is not processed for variable resolution.

4.5 Environment Variables

Environment variables are values existing in the exported list of value/pairs as provided by the operating system process. Apache, NGINX as well as other HTML server platforms provide environment variables to describe the requests being handed in.

4.6 CGI Variables

When operating Jet in CGI mode you have access to the form data submitted by the remote client (usually a browser or curl request) using the POST method.

Expressions are used to perform complicated calculations (arithmetic), perform comparisons between values (boolean), manipulate strings (string) or perform operations on dates (date).

Chapter 5

Expressions

5.1 Operators

Operators are used to perform arithmetic or boolean operations on arguments to derive a particular result. There are two types of operators, as follows:

- Arithmetic -
- Boolean -

5.1.1 Arithmetic Operators

5.1.2 Boolean Operators

5.2 Function Reference

5.3 Date Functions

5.3.1 datediff

5.3.2 unixtime

5.4 Math Functions

5.4.1 abs

Returns the absolute value of the given parameter.

5.4.2 acos

Returns to arc cosine of the given parameter.

5.4.3 asin

5.4.4 atan

5.4.5 cos

5.4.6 floor

5.4.7 ceil

5.4.8 max

Returns the greatest value item from a list of comma seperated values in a list.

5.4.9 min

Returns the least value item from a list of comma seperated values in a list.

5.4.10 pow

5.4.11 random

Use the random function to return a random number between 0 and 1.

5.4.12 sin

5.5 String Functions

String functions are provided to accelerate the building and parsing of strings passed into the expression. Numbers can be treated as a string or a number depending on the function called and the data state of the values being manipulated.

5.5.1 concat

The concat function is used to assemble a string by concatenating one or more values to formulate the resulting string.

5.5.2 integer

5.5.3 left

Use the left function to return the number of characters specified in the second parameter from the parameter passed as the first parameter.

If the number of characters to return is greater than the length of the first parameter then the entire parameter is returned.

5.5.4 ltrim

Use the ltrim function to remove the leading blanks (spaces, tabs and newlines) from a value. Trailing blanks are kept.

```
ltrim('  abc  ')    returns 'abc  '
```

5.5.5 reverse

5.5.6 right

5.5.7 round

Use the round function to round a numeric value to the specified number of digits after the decimal point.

5.5.8 rtrim

Use the rtrim function to remove the trailing blanks (spaces, tabs and newlines) from a value. Leading blanks are kept.

```
rtrim('  abc  ')    returns '  abc'
```

5.5.9 substring

Use the substring operation to extract a portion of a string and return the value as a string.

5.5.10 tolower

5.5.11 toupper

5.5.12 trim

Use the trim function to remove the blanks (spaces, tabs and newlines) from both ends of a value. See also ltrim and rtrim.

```
trim('  abc  ')     returns 'abc'
```

5.5.13 expr

Use the expr function to parse the string contents contained in the expr string.

Chapter 6

Common Gateway Interface Features

JET extension tags provides options for enabling the processing of data using the Common Gateway Interface (CGI) specification used by Apache and NGINX as well as other HTTP server applications. This makes JET a powerful tool for creating web interfaces and API handler environments for various javascript frameworks as well as just native HTML style interaction.

To enable the CGI features within a requested document you can specify the cgi=”true” attribute on a <jet> tag containing the document or <jet /> tag contained within the document.

When you enable the CGI interface the script will look for specific environment variables used to retrieve the input data from the browser interface and automatically provide these values to the JET scripting using the variables interface. There are currently three supported data formats or content types through this interface. These content types are:

formdata/urlencoded mutipart/formdata text/json

6.1 Session Control

Common use of the CGI is requiring the transaction based interaction of a web browser to maintain a state between interactions. JET 2.0 provides a built in mechanism to assist in managing a session concept. This approach uses a single session cookie to identify the session to the back end services.

By specifying a seesiondir keyword on the <jet> tag you enable the session control functionality. Upon the delivery of a request that does not contain a session ’token’ a session token will be generated for the request and a cookie will be returned in the reponse containing the session token. Additionally, a session file is placed into the directory named after the token. Captured session data will become available to future requests coming into the server and made

available through the variable formats for the session.

Under APIServer, session control is enabled for a site by its sessionDir setting in the server configuration instead of the <jet> tag.

Use the <set> tag with scope="session" to store a value in the session, and $[@name] to read it back (see Session Variables).

The session tags in SESSIONCTL.tag are deprecated for session management. Its setenv tag has been removed, and the startsession and endsession tags are no longer provided. Use scope="session" instead.

Chapter 7

Tag Reference

This chapter will cover all the tags and parameters required to perform the operations that JET provides.

A special character sequence is used to exclude a section from tag and variable parsing. This sequence is |> this is not parsed or resolved <|. This takes affect through all levels and depths of tag processing and is different from the <exclude /> tag as exclude may be further parsed if output is again evaluated.

7.1 bind

Use the bind tag inside an sql tag to supply the value for a ? placeholder in the statement. Each bind fills the next ? in order. The value is escaped or checked before it reaches the server, so whatever it contains can never change the meaning of the statement. Use bind for every value that comes from a request, a cookie or a session instead of writing $[name] into the statement text.

The bind tag has no closing tag and produces no output. It must be inside an sql tag, although it may be inside other tags (if, for and so on) within that sql tag.

```
<sql>select id, name from users where email=? and status=?
   <bind value="$[email]">
   <bind value="ACTIVE">
</sql>
```

The attributes are:

value - the value to bind. Required. Any expression or variable reference may be used here.

type - how the value is checked and sent. The values are:

- string - the default. Quoted and escaped.

- int - a whole number of up to 19 digits, optionally negative. Sent without quotes.
- decimal - a number with an optional fraction, for example -12.50. Sent without quotes.
- bool - 1, true, yes, 0, false or no (any case). Sent as 1 or 0.
- date - YYYY-MM-DD.
- datetime - YYYY-MM-DD HH:MM or YYYY-MM-DD HH:MM:SS.
- ident - a column or table name, or a sort direction, for the places SQL cannot take a value (ORDER BY and so on). Requires the allow attribute. Names are sent in backticks; ASC and DESC are sent as they are.

null - set to empty to send SQL NULL when the value is empty. Without it an empty value is sent as an empty string (and is an error for int, decimal, bool, date and datetime).

allow - for type="ident" only. A comma separated list of the values that are permitted. Any other value is an error.

```
<sql>select * from products order by ? ?
   <bind value="$[sort]" type="ident" allow="name,price,created">
   <bind value="$[dir]"  type="ident" allow="ASC,DESC">
</sql>
```

A ? inside a quoted string, a backtick name or a comment in the statement is not a placeholder. An sql tag that contains no bind tags is run exactly as written, so a ? in it is left alone.

These are errors, and can be caught with try/catch: a value that does not fit its type, an ident that is not in the allow list, a bind outside an sql tag, a bind with no value, an unknown type, and a statement whose number of ? placeholders differs from its number of bind tags. The error text never contains the bound value itself.

7.2 call

Use the call tag to evoke an executable file from a jet script.

The atrributes are:
pgm
argn
name
input
error
The call tag is unscoped.

```
<call pgm="ls" arg1="-al" name="listing" />
```

will call the ls command with parameter -al and place the output into a variable named 'listing'.

7.3 comment

Use the comment tag to create a section in the jet script that can be used for making comments and is ignored by the jet reader.

The comments tag has no attributes.

7.4 cookie

Use the cookie tag to specify a cookie when using the CGI features of JET.

7.5 dump

Use the dump tag to display the contents of all the global, local and cgi cariables to a specified file.

The attributes are:

file

7.6 exclude

Use the exclude tag to exclude a conainer contents from normal parsing and variable resolution.

7.7 expr

7.8 for

Use the for tag to iterate a tag container for a logical number of times.

The attributes are:

start

end

step

name - defines a name for the iterator that will be available within the loop.

scope - defines the scope of the iterator for the loop.

7.9 header

Use the header tag to output a header prior to outputting the process buffer to the requester.

The attributes are:
name
expr
value
container

7.10 if/else

Use the if tag to perform a conditional output on the tag container. An optional else container provides alternate output in the event the condition is not met.

The attributes are:
value1
value2
type
expr

7.11 ifrow/else

Use the ifrow tag to output the tag container if a row exists in the mysql tag. An optional else container provides alternate output in the event that there is no row for the sql result.

The attributes are:
sessionid

7.12 include

Use the include tag to include another content file at the location of the include tag.

The attributes are:
file

7.13 jet

Use the jet tag to specify parameters for the jet parsing session and to control the output options.

7.14 mysql

Use the mysql tag to specify parameters for connecting to a mysql server. The container of the tag is where you can specify the sql statement and method of processing the result output of the executed sql statement. The mysql session created is valid only withon the container of this tag.

The attributes are:
host
database
user
password
sessionid

7.15 read

Use the read tag to read the contents of a file contained on the local file system into a variable for further output or processing.

The attributes are:
file
name

7.16 set

Use the set tag to store initialize a variable to a value contained in a value attribute, the result of an expression (expr) attribute or the contents of the set tag container.

The attributes are:
name
expr
value
container
scope - where the variable is stored. The values are:

- global - the default. Available to all logic for the rest of the request.
- local - available only within the container where it is set.
- parent - stored in the scope of the enclosing tag.
- session - stored in the session file and kept across requests. Requires session control. See Session Variables.

7.17 sql

Use the sql tag to specify an sql statement to run on the sql server specified in the mysql tag.

As long as sql statements are executed within the same mysql containing tag then the same mysql session is used for each sql statement.

Values that come from a request, a cookie or a session should be passed with bind tags and ? placeholders rather than written into the statement text. See the bind tag.

```
<sql>delete from cart_item where session_uuid=? and sku=?
   <bind value="$[_session_id]">
   <bind value="$[sku]">
</sql>
```

A statement that changes the connection character set (SET NAMES, SET CHARACTER SET and similar) is refused. The connection always uses utf8mb4.

The attributes are:

sessionid

container

7.18 stream

Use stream tag to output data from the server without waiting for the standard output buffering and processing of the JET script. This is useful for outputting images and streams for audio and video without the server having to load the whole thing into RAM first.

The attributes are:

file

7.19 system

Use the system tag to execute a bash shell command within the JET script.

7.20 tag

Use the tag tag to define a new tag definition for use within the script.

Tags are normally defined globally and can be used anywhere within the script.

Tags can also be defined within other tags and the scope of those tags is only within the defining tag's container. The enclosed tag definition will not be available globally.

7.21 until

Use the until tag to process a container at least once and continue to process the container until the specified condition is met.

Note that this tag always processes the container at least once.

Be careful not to create a never ending loop by not modifying any of the condition parameters specified in the condition clause.

7.22 while

Use the while tag to process a container if the specified condition is true.

Note that the container may not be processed if the condition is never met.

Be careful not to create a never ending loop by not modifying any of the condition parameters specified in the condition clause.

7.23 whiledir

Use the whiledir tag to loop a directory path to the container for processing.

7.24 whilerow

Use the whilerow tag in combination with mysql and sql tags to provide a container to process for each row retrieved by the sql statement.

7.25 write

Use the write tag to write data to the local file system.