# WarpScript Cheat Sheet

March 2023

Documentation and tutorials on warp10.io

## Types

More about Types

| | |
|---|---|
| `100 0x64 0b1100100` | Long numbers |
| `3.1415 1e3 NaN Infinity -Infinity` | Double precision floating point numbers |
| `true false` | Booleans |
| `'hundred %25' "WarpScript"` | Text strings, URL encoded using UTF-8 |
| `<'`<br>`multiline`<br>`string`<br>`'>` | <' and '> must appear alone on a line of their own |
| `[ 1 2.0 '3' ]` | List of objects |
| `{ 'key1' 42 'key2' 'value2' }` | Map of key-value pairs |
| `('unique' 'objects' 42 )` | Set of objects |
| `36 'myvariable' STORE` | Store 36 in a variable |
| `$myvariable !$myvariable` | Use myvariable |
| `<%`<br>`  SAVE 'context' STORE`<br>`  [ 'arg1' 'arg2' ] STORE`<br>`  $arg1 $arg2 /`<br>`  $context RESTORE`<br>`%> 'mymacro' STORE` | Macro definition |
| `44 22 @mymacro` | Call 'mymacro' with parameters 44 and 22 |
| `NEWGTS 'mygts' RENAME` | Create and name a Geo Time Series™ object |
| `NOW 24.4 -5.4 NaN 42 ADDVALUE` | Add a datapoint to a Geo Time Series™ with value 42 at the current timestamp at location 24.4 N and 5.4 W |

## Comments

More about Comments

| | |
|---|---|
| `// WarpScript comment` | Single line comment |
| `# WarpScript comment` | Single line comment |
| `/* WarpScript comment */` | Multi line comment |

## Function calls

More about Calls

| | |
|---|---|
| `arg1 arg2 arg3 ... FUNCTION` | Call 'FUNCTION' with arguments arg1 arg2 arg3 ... |
| `PI COS` | Returns cos π |
| `1 1 +` | Returns 2 |

## String Manipulation

More about Strings

| | |
|---|---|
| `+, JOIN` | Concatenation |
| `SUBSTRING, SPLIT, TRIM` | Substring generation |
| `MATCH, REPLACE, REPLACEALL` | Regular expression or simple search |

## Operators

More about Operators

| | |
|---|---|
| `+ - * / % **` | Add, substract, multiply, divide, modulo, power |
| `== != < <= => >` | Comparison operators |
| `&& \|\| ! AND OR NOT` | Boolean operators |
| `& \| ^ >> << >>> <<<` | Binary operators |

## Frameworks

More about Frameworks

| | |
|---|---|
| `BUCKETIZE` | Apply a function to points in a bucket. Typically used for downsampling |
| `MAP` | Apply a function to a sliding window. Typically used for smoothing |
| `REDUCE` | For each timestamp, apply a function to each datapoint of input series, *e.g.* adding values one to one |
| `FILTER` | Used to select Geo Time Series™ from a larger set |
| `FILL` | Fill missing values in two Geo Time Series™ |

## Composite Structures Manipulation

More about Lists & Maps

| | |
|---|---|
| `+! APPEND` | Add elements to a list. +! will reuse the list |
| `[ 2 3 4 ] <% DROP 2 ** %> LMAP //returns [ 4 9 16 ]` | Map the square macro on every element of the list to build a new list |
| `{} 'value' 'key' PUT` | Put a key value pair in an empty map |
| `index GET, 'key' GET` | Extract a value from a list or a map |
| `CONTAINS, CONTAINSKEY, CONTAINSVALUE` | Test presence in a list or a map |
| `LSORT, SORTBY, MSORT, REVERSE` | Sorting function for list and maps |

## Control / Flow

More about Control

| | |
|---|---|
| `<% if-macro %> <% then-macro %> IFT` | If then |
| `<% if-macro %> <% then-macro %> <% else-macro %> IFTE` | If then else |
| `0 10 <% 'i' STORE %> FOR` | For i = 0 to 10 |
| `[ 2 3 4 ] <% 'arg' STORE $arg 2 ** %> FOREACH` | Iterate on every element of the list |
| `<% try-macro %> <% catch-macro %> <% finally-macro %> TRY` | Try catch. You can use ERROR function in the catch macro |
| `BREAK` | Break out of the current loop |
| `CONTINUE` | Go to the next iteration of the loop |
| `RETURN` | Return immediately from the currently executing macro |

## Debugging WarpScript

More about Debug

| | |
|---|---|
| `STOP` | Stop the WarpScript execution |
| `FAIL, 'my error' MSGFAIL` | Stop the WarpScript execution with an error |
| `SECTION` | Name a portion of code |
| `LINEON` | Enable automatic section naming based on line number |

## Stack Manipulation

More about Stack

| | |
|---|---|
| `SWAP` | Swap the top 2 elements of the stack |
| `DUP` | Duplicate the top element of the stack |
| `DROP` | Drop the top element of the stack |
| `CLEAR` | Empty the stack |
| `ROT` | Cycle the top 3 elements of the stack |
| `N ROLL` | Cycle up one level the top N elements of the stack |
| `N ROLLD` | Cycle down one level the top N elements of the stack |
| `SNAPSHOT` | Generate WarpScript code which will regenerate the content of the stack |

## Conversions

More about Conversion

| | |
|---|---|
| `->JSON, JSON->` | Serialize / deserialize JSON structures |
| `->B64, B64->` | Encode / decode string in base64 |

## Date and Time functions

More about Date & Time

| | |
|---|---|
| `NOW` | Push onto the stack the current timestamp in platform time units |
| `ps, ns, us, ms, s, m, h, d, w` | Push onto the stack the number of platform time units for the specified duration, e.g. 4 d will push the number of time units in 4 days |
| `NOW 'optional_timezone' TSELEMENTS` | Convert a timestamp into a list of timestamp elements |
| `NOW 5 ADDDAYS` | Add days to a timestamp or a list of timestamp elements |
| `NOW 4 ADDMONTHS` | Add months to a timestamp or a list of timestamp elements |
| `[ 2009 02 14 6 01 31 ] 'optional_timezone' TSELEMENTS->` | Convert a list of timestamp elements into a timestamp |
| `'2016-01-19T16:07:37Z' TOTIMESTAMP` | Convert a date in ISO8601 format into a timestamp in the platform time units |
| `$ts 'optional_timezone' ISO8601` | Convert a timestamp into its ISO8601 representation using UTC or the specified time zone |

## Geo Time Series (GTS) functions

More about GTS

| | |
|---|---|
| `NEWGTS` | Pushe onto the stack an empty GTS instance |
| `$gts 'foo' RENAME` | Change the name of a GTS |
| `$gts { 'foo' 'bar' 'id' '42' } RELABEL` | Change the labels of a GTS |
| `$gts WRAP` | Pack a GTS (or a list thereof) or a GTS Encoder into a STRING |
| `$wrapped_gts UNWRAP` | Unpack a GTS (or a list thereof) or a GTS Encoder from a STRING |
| `$gts [ [ $t0 $t1 ] [ $t2 $t3 ] [ ... ] ] CLIP` | Clip a GTS according to a series of tick intervals |
| `$gts $end_tick $duration TIMECLIP` | Clip GTS by restricting their ticks to those within a time interval |
| `$bucketized_gts FILLNEXT` | Fill gaps in a bucketized GTS by re-using the value/location/elevation of the next non empty bucket to fill each empty bucket |
| `$bucketized_gts FILLPREVIOUS` | Fill gaps in a bucketized GTS by re-using the value/location/elevation of the previous non empty bucket to fill each empty bucket |
| `$bucketized_gts [ $lat $long $elevation $value ] FILLVALUE` | Fill gaps in a bucketized GTS with a fixed value/location/elevation |
| `$gts DEDUP` | Remove duplicate ticks |
| `$gts_list MERGE` | Merge multiple GTS |
| `$gts $size SHRINK` | Shrink the number of values of a GTS to size |
| `$gts $n LTTB` | Select n meaningful values in a GTS |
| `$gts COMPACT` | Remove measurements which have the same value, location and elevation as the previous one |
| `$gts RANGECOMPACT` | Retain only the first and last mesurement of ranges with identical value location and elevation |
| `COMMONTICK` | Keep only the common timestamps of all input GTS |
| `[ $token 'classname' { 'label1' 'value1' } $end $duration ] FETCH` | Retrieve data according to fetch criteria |
| `[ $token 'classname' { 'label1' 'value1' } ] FIND` | Identify GTS according to criteria on classes and labels / attributes |
| `[ $token 'classname' { 'label1' 'value1' } ] FINDSETS` | Return sets of values for the classes, labels and attributes |
| `[ $token 'classname' { 'label1' 'value1' } ] FINDSTATS` | Compute statistics on matching GTS |

Collect all the WarpScript badges and ask for your diploma!

Hello World | FETCH/UPDATE | BUCKETIZE/MAP | Processing | WarpScript on Spark