Module ngx_http_map_module

The `ngx_http_map_module` module creates variables whose values depend on values of other variables.

Example Configuration

```
map $http_host $name {
    hostnames;

    default       0;

    example.com   1;
    *.example.com 1;
    example.org   2;
    *.example.org 2;
    .example.net  3;
    wap.*         4;
}

map $http_user_agent $mobile {
    default       0;
    "~Opera Mini" 1;
}
```

Directives

Syntax: **map** *string $variable* { ... }
Default: —
Context: `http`

Creates a new variable whose value depends on values of one or more of the source variables specified in the first parameter.

Before version 0.9.0 only a single variable could be specified in the first parameter.

Since variables are evaluated only when they are used, the mere declaration even of a large number of “ `map` ” variables does not add any extra costs to request processing.

Parameters inside the `map` block specify a mapping between source and resulting values.

Source values are specified as strings or regular expressions (0.9.6).

Strings are matched ignoring the case.

A regular expression should either start from the “ `~` ” symbol for a case-sensitive matching, or from the “ `~*` ” symbols (1.0.4) for case-insensitive matching. A regular expression can contain named and positional captures that can later be used in other directives along with the resulting variable.

If a source value matches one of the names of special parameters described below, it should be prefixed with the “ `\` ” symbol.

The resulting value can contain text, variable (0.9.0), and their combination (1.11.0).

The following special parameters are also supported:

`default` *`value`*

sets the resulting value if the source value matches none of the specified variants. When `default` is not specified, the default resulting value will be an empty string.

`hostnames`

indicates that source values can be hostnames with a prefix or suffix mask:

```
*.example.com 1;
example.*     1;
```

The following two records

```
example.com   1;
*.example.com 1;
```

can be combined:

```
.example.com  1;
```

This parameter should be specified before the list of values.

`include` *`file`*

includes a file with values. There can be several inclusions.

`volatile`

indicates that the variable is not cacheable (1.11.7).

The search is performed in the following order of priority and terminates on the first matching variant:

1. the string value without a mask
2. the longest string value with a prefix mask, e.g. “ `*.example.com` ”
3. the longest string value with a suffix mask, e.g. “ `mail.*` ”
4. the first matching regular expression (in order of appearance in the configuration file)
5. the default value

Syntax: **`map_hash_bucket_size`** *`size`*;
Default: `map_hash_bucket_size 32|64|128;`
Context: `http`

Sets the bucket size for the map variables hash tables. Default value depends on the processor’s cache line size. The details of setting up hash tables are provided in a separate document.

Syntax: **`map_hash_max_size`** *`size`*;
Default: `map_hash_max_size 2048;`
Context: `http`

Sets the maximum *`size`* of the map variables hash tables. The details of setting up hash tables are provided in a separate document.