2 用户宏

概述

除了支持开箱即用的宏之外，Zabbix 还支持更灵活的用户宏。

用户宏可以在全局、模板和主机级别进行定义。这些宏具有一个特殊的语法：

```
{$MACRO}
```

用户宏可被用于：

- 监控项名称；
- 监控项键值参数；
- 触发器名称和描述；
- 触发器表达式参数和常量(详细查阅下文的 示例)
- 许多其他位置（详细查阅 位置支持的宏)

宏名称中允许使用以下字符：**A-Z** , **0-9** , **_** , **.** 。

Zabbix 根据以下优先级解析宏：

1. 主机级别的宏（首先检查)；
2. 为主机的第一级别模板定义的宏（即，直接链接到主机的模板），按照模板 ID 来排序；
3. 为主机的第二级别模板定义的宏，按照模板 ID 来排序；
4. 为主机的第三级别模板定义的宏，按照模板ID来排序，等；
5. 全局宏（最后检查)。

换言之，如果一个主机不存在一个宏，Zabbix将会尝试在级别递增的主机模板中找到它，如果仍然找不到，那么将会使用全局宏（如果全局宏存在的话）。

如果 Zabbix不能找到宏， 那么宏将不能被解析。

如果要定义用户宏，请转到Zabbix的前端页面的如下位置：

- 对于全局宏，请访问 *管理 → 常规 → 右上角下拉菜单选择 “宏”* 。
- 对于主机和模板级别的宏，请打开主机或模板属性并查看 *宏* 标签页面。

如果在模板的监控项或触发器使用用户宏，建议将该宏添加到模板，即使它被定义在全局级别上。这样的话，将模板导出至XML文件中，之后在其他系统中导入，那么在其他系统中使用也将会达到预期的使用效果。

全局和主机宏的常用案例

- 利用具有主机特定属性的模板：密码、端口号、文件名称、正则表达式等；
- 运用全局宏进行全局的一键配置更改或微调。

示例

示例 1

在 “Status of SSH daemon” 监控项键值中使用主机级别的宏：

```
net.tcp.service[ssh,,{$SSH_PORT}]
```

该监控项可以分配给多个主机，前提是在这些主机上定义了 **{$SSH_PORT}** 的值。

示例 2

在 “CPU load is too high” 触发器上使用主机级别的宏：

```
{ca_001:system.cpu.load[,avg1].last()}>{$MAX_CPULOAD}
```

这样的触发器将会在模板上创建，而不会在单个主机中编辑。

如果要使用数值作为函数参数（例如，**max(#3)**），则在宏定义中要包含井号（hash mark）例如：SOME_PERIOD ⇒ #3

示例 3

在“CPU load is too high”触发器中使用了两个宏：

```
{ca_001:system.cpu.load[,avg1].min({$CPULOAD_PERIOD})}>{$MAX_CPULOAD}
```

请注意，宏可以用作触发器函数的参数，在这个示例中为 **min()**。

在触发器表达式中，如果引用参数或者常量，则用户宏将会解析。如果引用主机、监控项键值、函数、操作或其他触发器表达式的话，他们将不会解析。

用户宏上下文

An optional context can be used in user macros, allowing to override the default value with context-specific one. 可以在用户宏中使用可选上下文，允许使用特定的上下文来重写默认的值。

User macros with context have a similar syntax: 具有上下文的用户宏具有类似的语法：

```
{$MACRO:context}
```

Macro context is a simple text value. The common use case for macro contexts would be using a low-level discovery macro value as a user macro context. For example, a trigger prototype could be defined for mounted file system discovery to use a different low space limit depending on the mount points or file system types. 宏上下文是一个简单的文本值。宏上下文的常见使用案例是使用自动发现宏值作为用户宏上下文。例如，根据文件系统的挂载点或文件系统类型，可以为挂载的文件系统自动发现定义自动发现触发器原型以使用不同的可使用空间阈值。

Only low-level discovery macros are supported in macro contexts. Any other macros are ignored and treated as plain text. 在宏上下文中只支持自动发现的宏。任何其他宏都将被忽略，并视为纯文本。

Technically, macro context is specified using rules similar to item key parameters, except macro

context is not parsed as several parameters if there is a , character: 从技术上讲，宏上下文是使用类似于监控项键值参数的规则来指定的，除非有一个字符“,”，否则宏上下文是不被解析为几个参数：

- Macro context must be quoted with " if the context contains a } character or starts with a " character. Quotes inside quoted macros must be escaped with the \ character. The \ character itself is not escaped, which means it's impossible to have a quoted macro ending with the \ character - the macro {$MACRO:"a:\b\c\"} is invalid.
- 宏的内容必须引用’“’, 如果宏内容包含有’}’的字符或者从一个’”’字符. 那在引号中的引号必须使用’\’字符进行转义从而不改变宏的内容, 这也说明宏被引用后的字符’“’没有被转义则不能使用, 宏{$MACRO:"a:\b\c\"}是无效的.
- The leading spaces in context are ignored, the trailing spaces are not. For example {$MACRO:A} is the same as {$MACRO: A}, but not {$MACRO:A }.
- 宏内容中前面的空格会被忽略, 后面的空格不会忽略. 例如:{$MACRO:A} 和 {$MACRO: A} 相同, 但不同于 {$MACRO:A }.
- All spaces before leading quotes and after trailing quotes are ignored, but all spaces inside quotes are not. Macros {$MACRO:"A"}, {$MACRO: "A"}, {$MACRO:"A" } and {$MACRO: "A" } are the same, but macros {$MACRO:"A"} and {$MACRO:" A "} are not.
- 宏变量中引号外面的空格可以被忽略, 但是引号里面的空格不会忽略. 宏{$MACRO:"A"}, {$MACRO:"A"}, {$MACRO:"A" } 和 {$MACRO: "A" } 相同, 但是跟宏{$MACRO:"A"} 和 {$MACRO:" A "} 不同.

The following macros are all equivalent, because they have the same context: {$MACRO:A}, {$MACRO: A} and {$MACRO:"A"}. This is in contrast with item keys, where key[a], key[a] and key["a"] are the same semantically, but different for uniqueness purposes.

以下的宏是一样的, 因为它们表示的内容一样：{$MACRO:A}, {$MACRO: A} 和 {$MACRO:"A"}. 这与监控项相反, key[a], key[a] 和 key["a"] 在语法上相同, 但唯一性不同.

When context macros are processed, Zabbix looks up the macro with its context. If a macro with this context is not defined by host or linked templates, and it is not a defined as a global macro with context, then the macro without context is searched for.

当宏在使用时,zabbix会查看宏的背景, 如果宏没有在主机或者模板上关联, 或者有没有在全局宏中定义, 那么宏不会被使用.

See usage example of macro context in a disk space trigger prototype and take limitation clause into consideration.

参考usage example章节中磁盘触发器原型关于宏变量的使用示例.