编写简单脚本——执行系列动作

【操作步骤】

1. 在前端编写脚本，脚本的5种动作：调用服务、延时、等待、条件判断、触发事件
2. 演示1：编写与执行一个脚本，调用服务：persistent\_notification.create
3. 演示2：编写与执行一个脚本，系列动作，包括多次调用服务、延时。
4. 查看对应的文本编辑文件scripts.yaml
5. 手工编写脚本配置，演示3：增加条件判断，在条件满足下继续执行脚本

【参考】

* 脚本文档

<https://www.hachina.io/docs/434.html>

* 服务文档

<https://www.hachina.io/docs/471.html>

* 视频中演示的scripts.yaml完整样例

'1539694616968':

alias: 演示脚本

sequence:

- condition: state

entity\_id: light.gateway\_light\_7c49eb18e3a7

state: "off"

- data:

message: 运行演示脚本……

notification\_id: 123321

title: 演示

service: persistent\_notification.create

- data:

entity\_id: light.gateway\_light\_7c49eb18e3a7

service: homeassistant.turn\_on

- delay: 00:00:05

- data:

entity\_id: light.gateway\_light\_7c49eb18e3a7

service: light.turn\_off

- data:

notification\_id: 123321

service: persistent\_notification.dismiss

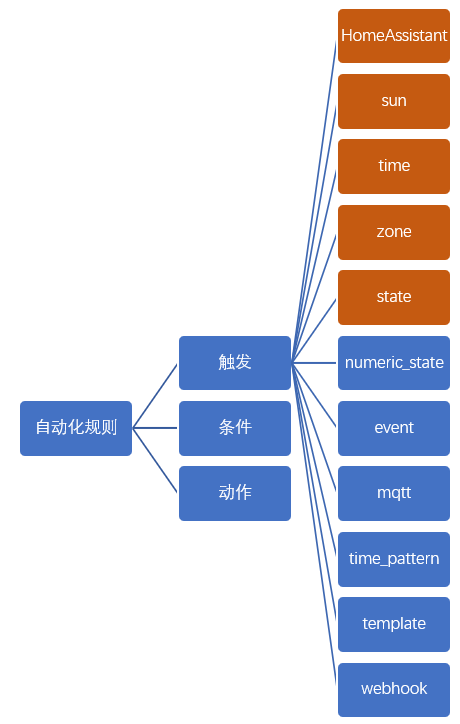
编写简单自动化规则

【操作步骤】

1. 自动化规则的三要素：触发、条件、动作
2. 前端样例编写1：当打开灯时，语音播报开灯信息
3. 手工样例编写2：当关闭灯时，语音播报关灯信息
4. 手工样例编写3：每5秒执行开关灯操作
5. 介绍触发中的：homeassistant、sun、时间、地点

【参考】

* 自动化规则的触发



* 自动化文档

<https://www.hachina.io/docs/445.html>

模板——嵌入配置文件中的程序

【操作步骤】

1. 演示模板开发者界面
2. 包含模板的脚本样例一
3. 包含模板的脚本样例二
4. 包含模板的自动化样例三

【参考】

* 模板参考文档

<https://www.hachina.io/docs/802.html>

* Jinjia2模板语法

<http://jinja.pocoo.org/docs/dev/templates/>

* 模板样例1，播报比特币行情

example1:

alias: 播报比特币行情

sequence:

- service: tts.google\_say

entity\_id: "all"

data\_template:

message: 当前比特币行情{{ states.sensor.exchange\_rate\_1\_btc.state }}美元

* 模板样例2，如果主人在家播报比特币行情，如果主人不在家，播报主人离家距离

example2:

alias: 播报比特币行情或离家距离

sequence:

- service: tts.google\_say

entity\_id: "all"

data\_template:

message: >

{% if is\_state('device\_tracker.zhujishengiphone', 'home') %}

当前比特币行情{{ states.sensor.exchange\_rate\_1\_btc.state }}美元

{% else %}

主人离家{{ distance(states.device\_tracker.zhujishengiphone)|round(2) }}公里

{% endif %}

* 模板样例3，每隔10秒，如果小米网关灯亮度大于200，则运行样例2

- alias: auto\_broadcast

trigger:

- platform: time\_pattern

seconds: /10

condition:

condition: template

value\_template: "{{ is\_state('light.gateway\_light\_7c49eb18e3a7', 'on') and (states.light.gateway\_light\_7c49eb18e3a7.attributes.brightness > 200) }}"

action:

- service: script.turn\_on

entity\_id: script.example2

事件与事件消息接收

【操作步骤】

1. 构建基于事件触发的自动化
2. 演示前端开发者界面中的事件触发
3. 传递事件中的数据
4. 编写脚本触发事件

【参考】

* 事件参考文档

<https://www.hachina.io/docs/470.html>

* 视频中的自动化规则

- id: '1540371927161'

alias: abc Automation

trigger:

- event\_data: {}

event\_type: abc

platform: event

condition: []

action:

- alias: ''

data\_template:

entity\_id: all

message: 系统中发生了事件ABC：{{ trigger.event.data.my\_message }}

service: tts.google\_say

* 视频中的脚本

abc\_script:

sequence:

- event: abc

event\_data:

my\_message: 欢迎观看我的视频

前端输入组件+packages配置

【操作步骤】

1. 使用packages配置完成样例一配置文件的上传
2. 解释样例一的配置内容，并演示效果
3. 各种前端组件的演示

【参考】

* packages配置介绍

<https://www.hachina.io/docs/4156.html>

* 前端输入组件配置介绍

<https://www.home-assistant.io/components/#search/input>

* 样例一中的配置example\_6\_5\_1.yaml

# example\_6\_5\_1.yaml

input\_number:

turnon\_duration:

name: 亮灯时长

initial: 0

min: 0

max: 8

step: 1

automation:

- alias: turn on light n seconds

initial\_state: True

trigger:

- platform: state

entity\_id: input\_number.turnon\_duration

condition:

condition: numeric\_state

entity\_id: input\_number.turnon\_duration

above: 0

action:

- service: light.turn\_on

entity\_id: light.gateway\_light\_7c49eb18e3a7

- delay:

seconds: "{{ states('input\_number.turnon\_duration')|int }}"

- service: light.turn\_off

entity\_id: light.gateway\_light\_7c49eb18e3a7

- service: input\_number.set\_value

data:

entity\_id: input\_number.turnon\_duration

value: 0

group:

example1\_view:

name: 样例一

entities: light.gateway\_light\_7c49eb18e3a7, input\_number.turnon\_duration

view: yes

语音+音乐+灯光闹钟

【操作步骤】

1. 使用samba上传MP3文件
2. 配置文件介绍
3. 上传配置文件，重启HA
4. 演示闹钟功能

【参考】

* 闹钟的配置example\_6\_6\_1.yaml

# example\_6\_6\_1.yaml

# 是否打开闹钟

input\_boolean:

alarm\_clock:

name: 闹钟

initial: off

icon: mdi:alarm

# 闹钟的时间

input\_datetime:

alarm\_time:

name: 时间

has\_time: true

has\_date: false

initial: 07:00

# 播放的音乐选择

input\_select:

alarm\_music:

name: 音乐

icon: mdi:music

options:

- 卡农

- I Can Feel It Coming

- 沧海一声笑

# 播放的文字

input\_text:

alarm\_text:

name: 语音提醒

initial: 该起床了,该起床了！

min: 0

max: 30

# 亮灯时长

input\_number:

alarm\_light\_duration:

name: 亮灯提醒（分钟）

initial: 0

min: 0

max: 60

step: 1

# 自动化每分钟触发

automation:

- alias: Alarm Clock

initial\_state: True

trigger:

- platform: time\_pattern

minutes: '/1'

seconds: 0

# 判断闹钟是否打开，判断当前时间是否与闹钟时间相同

condition:

- condition: state

entity\_id: input\_boolean.alarm\_clock

state: 'on'

- condition: template

value\_template: '{{((as\_timestamp(now())|int)|timestamp\_custom("%H:%M"))==states.input\_datetime.alarm\_time.state|truncate(5,False,"",0) }}'

action:

# 播放文字语音

- service: tts.google\_say

entity\_id: "all"

data\_template:

message: '{{states.input\_text.alarm\_text.state}}'

# 打开灯

- service: light.turn\_on

entity\_id: light.gateway\_light\_7c49eb18e3a7

# 延时5秒钟

- delay:

seconds: 5

# 等待语音播放结束，最长一分钟

- wait\_template: "{{is\_state('media\_player.vlc','idle')}}"

timeout: '00:01:00'

continue\_on\_timeout: 'true'

# 延时5秒钟

- delay:

seconds: 5

# 根据选择项，播放对应的音乐

- service: media\_player.play\_media

data\_template:

entity\_id: media\_player.vlc

media\_content\_type: music

media\_content\_id: >

{% if states.input\_select.alarm\_music.state=='卡农' %}

/home/pi/Music/Canon.mp3

{% elif states.input\_select.alarm\_music.state=='沧海一声笑' %}

/home/pi/Music/沧海一声笑.mp3

{% else %}

/home/pi/Music/ICanFeelitComing.mp3

{% endif %}

# 延时n分钟

- delay:

minutes: "{{ states('input\_number.alarm\_light\_duration')|int }}"

# 关灯

- service: light.turn\_off

entity\_id: light.gateway\_light\_7c49eb18e3a7

# 将闹钟相关的控制项纳入一个组

group:

alarm\_clock:

name: 闹钟

entities:

input\_boolean.alarm\_clock, input\_datetime.alarm\_time, input\_text.alarm\_text, input\_select.alarm\_music, input\_number.alarm\_light\_duration

# 定义一个页面组，包含闹钟以及闹钟过程中相关的实体

alarm\_clock\_view:

name: 闹钟

entities: light.gateway\_light\_7c49eb18e3a7, media\_player.vlc, group.alarm\_clock

view: yes