Simple keyboard udev rule

Sometimes it's useful to trigger an action based on a device event - like "do this when that device connects to my pc". Exactly this was my case when I was connecting my bluetooth keyboard many times a day as long as it automatically goes to sleep after 10 minutes of inactivity. I needed to run a simple command to set up key recurrence speed (rate) and a delay before doing so - `xset r rate 160 40`.

udev rules

udev rules is a nifty tool that triggers a command or a script if system detects an event - `ACTION` in it's terms. The implementation of such rule is "quite" easy as it's a simple text file living under `/etc/udev/rules.d/` directory. The file specifies an `ACTION` which needs to happen to trigger the rule, a `SUBSYSTEM` which the action needs to happen in, may contain `ATTR{}` that narrows down the device list that triggers the action and finally the command or a script that should run - RUN directive.

Get the values

The udev rule params mentioned above need to be stated. There is a set of nifty commands that will help. In my case I can do a little trick that I power off and back on my keyboard so it reconnects and if I'm monitoring current events I can catch first info.

`udevadm monitor -u` and then reconnect the keyboard. From the log I get I'm looking for "add" action which is the third column and then "input" device type which is identified by last column in brackets. Narrowing down the log according these rules gives me one line:

```
UDEV  [371177.724289] add      /devices/pci0000:00/0000:00:14.0/usb1/1-3/1-
 ↪  3:1.0/bluetooth/hci0/hci0:2/0005:05AC:024F.001C/input/input57
 ↪  (input)
```

Now I do have the device path (4th column) that I can work with. I can ask udevadm to give me some device details, like a name:

```
udevadm info -p /devices/pci0000:00/0000:00:14.0/usb1/1-3/1-
 ↪  3:1.0/bluetooth/hci0/hci0:2/0005:05AC:024F.001C/input/input57
```

That gives a lot of useful info and the `NAME` is among them. Also `SUBSYSTEM` can be found there which is also needed for the udev rule.

The script

The command I need to run is `xset r rate 160 40` which is fairly simple, but it needs to have "access" to Xserver. If the command is run by udev it won't have such access unless I tell to udev a few things - `DISPLAY` and `XAUTHORITY`. Those environment variables need to be set prior the command run, so I wrapped it up to this script:

```
#!/usr/bin/env bash

(
    sleep 1

    DISPLAY=":0.0"
    XAUTHORITY="/home/n1/.Xauthority"
    export DISPLAY XAUTHORITY

    xset r rate 160 40
) &
```

The `sleep` part is needed because somehow my keyboard needs a little bit of time to get ready.

The rule

Since all the info is gathered and the script is written I, can construct the udev rule. So

```
ACTION=="add", SUBSYSTEM=="input", ATTR{name}=="Keychron K6",
 ↪  RUN+="/home/n1/scripts/keyboard_xset.sh"
```

saved under `/etc/udev/rules.d/50-keychron_rule.rules` does the job. Now one needs to reload rules set for udev which can be done with `udevadm control --reload-rules` command.

Debugging

udev can produce pretty much detailed log if is told to and one can do that with `udevadm control --log-priority=debug` and then listen on syslog (`tail -f /var/log/syslog`) or journal (`journal -f`)

To put the logging back to where it was the following command can be used `udevadm control --log-priority=info`.