bluetoothctl Bluetooth from the shell

START HERE

`bluetoothctl`
Open the prompt

`pair` *`<dev>`*
Pair with a device

`power on`
Switch the adapter on

`trust` *`<dev>`*
Allow auto-reconnect

`agent on`
Handle pairing prompts

`connect` *`<dev>`*
Connect to it

`scan on`
Look for devices

`devices`
List known devices

Scan find it → `pair` **Pair** exchange keys → `trust` **Trust** allow reconnect → `connect` **Connect** use profiles

CONCEPTS

<dev> is the device MAC address (AA:BB:CC:DD:EE:FF). Press Tab to complete it from the scan results.

Pairing needs an agent. Inside the prompt it is registered for you; in one-shot mode it is not, so pair interactively.

This tool talks to the bluetoothd daemon; if the prompt shows no controller, the service is not running.

SESSION

`bluetoothctl`
Open the interactive prompt

`bluetoothctl` *`<command>`*
Run one command and exit
Example: bluetoothctl devices Paired

`bluetoothctl --timeout` *`<seconds>`* **`scan on`**
Scan for a fixed time, then exit

`bluetoothctl --agent` *`<capability>`*
Start with an agent of that capability

`bluetoothctl --monitor`
Print low-level monitor output

`bluetoothctl --version`
Show the BlueZ version

`help`
List the commands of this menu

`quit`
Leave the prompt

`menu` *`<name>`*
Enter a submenu

`back`
Return to the main menu

CONTROLLER

`power on`
Switch the adapter on
Many setups need this after boot or an rfkill unblock.

`power off`
Switch the adapter off

`show`
Show controller state

`list`
List controllers

`select` *`<ctrl>`*
Pick the default controller

`system-alias` *`<name>`*
Set the name others see

`discoverable on`
Let other devices find this one

`discoverable-timeout` *`<seconds>`*
Show or set discoverable time
0 means no timeout.

`pairable on`
Accept incoming pairing requests

`agent on`
Register the pairing agent

`default-agent`
Make this agent the default

`agent` *`<capability>`*
Choose the agent's input/output type
DisplayOnly, DisplayYesNo, KeyboardDisplay, KeyboardOnly, NoInputNoOutput.

`advertise on`
Start LE advertising

DISCOVER

`scan on`
Start scanning for devices

`scan off`
Stop scanning

`scan le`
Scan for Low Energy devices only

DISCOVER (cont.)

`scan bredr`
Scan for Classic devices only

`devices`
List known devices

`devices Paired`
List only paired devices

`devices Connected`
List only connected devices

`devices Trusted`
List only trusted devices

`info` *`<dev>`*
Show a device's state and profiles

PAIR AND CONNECT

`pair` *`<dev>`*
Pair with a device

`trust` *`<dev>`*
Let it reconnect automatically

`connect` *`<dev>`*
Connect all profiles

`disconnect` *`<dev>`*
Drop the connection

`remove` *`<dev>`*
Forget the device and its pairing
Use it to fix a device that refuses to reconnect, then pair again.

`connect` *`<dev>`* *`<uuid>`*
Connect one profile only
Profile names such as a2dp-sink or hfp-hf also work.

`untrust` *`<dev>`*
Stop automatic reconnects

`block` *`<dev>`*
Refuse connections from it

`unblock` *`<dev>`*
Allow it again

`bearer` *`<dev>`* *`<last-seen/bredr/le>`*
Prefer Classic or LE for a dual-mode device

SUBMENUS

`menu gatt`
Browse and read GATT attributes (LE)

`menu scan`
Set scan filters (transport, RSSI, UUID)

`menu advertise`
Configure LE advertising data

`menu player`
Control media players on a connected device

Other submenus: admin, monitor, endpoint, transport, mgmt, assistant, le, bredr.

RECIPES

Pair a new headset

```
bluetoothctl
power on
agent on
default-agent
scan on
# wait for the device, then:
pair AA:BB:CC:DD:EE:FF
trust AA:BB:CC:DD:EE:FF
connect AA:BB:CC:DD:EE:FF
scan off
```

Reconnect a known device

```
bluetoothctl devices Paired
bluetoothctl connect
AA:BB:CC:DD:EE:FF
bluetoothctl info
AA:BB:CC:DD:EE:FF
```

Start over with a stubborn device

```
bluetoothctl remove
AA:BB:CC:DD:EE:FF
bluetoothctl --timeout 15 scan
on
# then pair, trust, connect
interactively
```

Script several commands

```
bluetoothctl <<EOF
power on
devices Connected
show
EOF
```