

# Zephyr – new shell and logger

Jakub Rzeszutko, Krzysztof Chruściński

*19.09.2018*

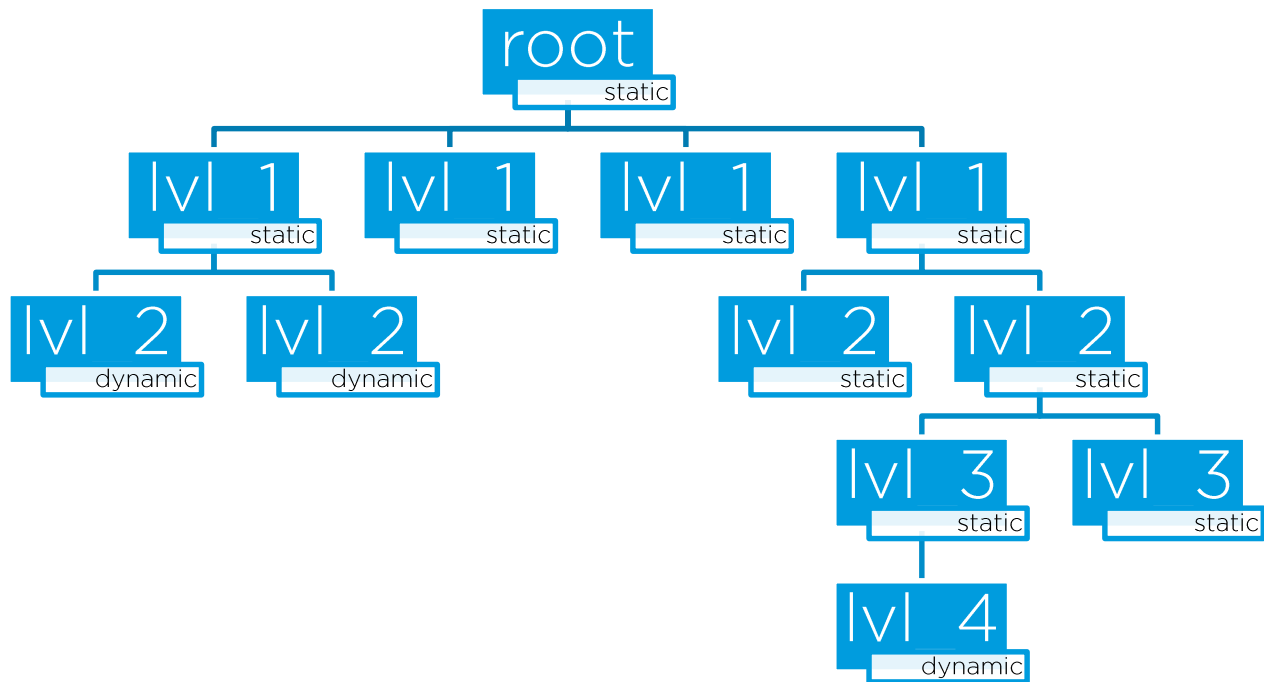
Shell

# Features

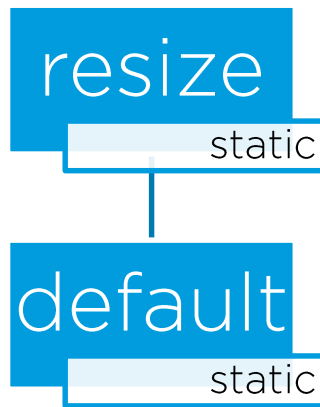
- Multi-instance
- New commands concept: commands tree, static and dynamic commands
- Integration with Zephyr Logger
- Smart commands completion with the Tab key
- Commands history
- Meta keys: ctrl+a, ctrl+c, ctrl+e, ctrl+l, ctrl+u, ctrl+w
- Wildcards support: \* and ?
- Build-in commands
- Kconfig configuration to optimize resources usage

# Commands

- Root commands can be defined in any file
- Root commands must have unique syntax
- On one level only one command type is allowed
- Deepest found handler will be executed



# Commands - examples



```
SHELL_CREATE_STATIC_SUBCMD_SET(m_sub_resize)
{
    SHELL_CMD(default, NULL, SHELL_HELP_RESIZE_DEFAULT, cmd_resize_default),
    SHELL_SUBCMD_SET_END
};

SHELL_CMD_REGISTER(resize, &m_sub_resize, SHELL_HELP_RESIZE, cmd_resize);
```

# Commands - handlers

```
static void cmd_resize(const struct shell *shell, size_t argc, char **argv)
{
    int err;

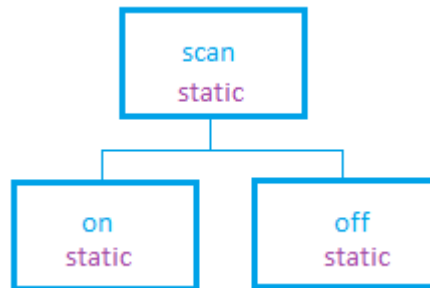
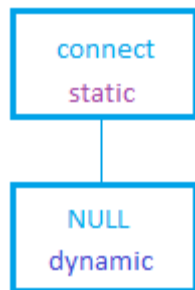
    if (!shell_cmd_precheck(shell, (argc <= 2), NULL, 0)) {
        return;
    }

    if (argc == 2) {
        shell_fprintf(shell, SHELL_ERROR, "%s:%s%s\r\n", argv[0],
                      SHELL_MSG_UNKNOWN_PARAMETER, argv[1]);
        return;
    }

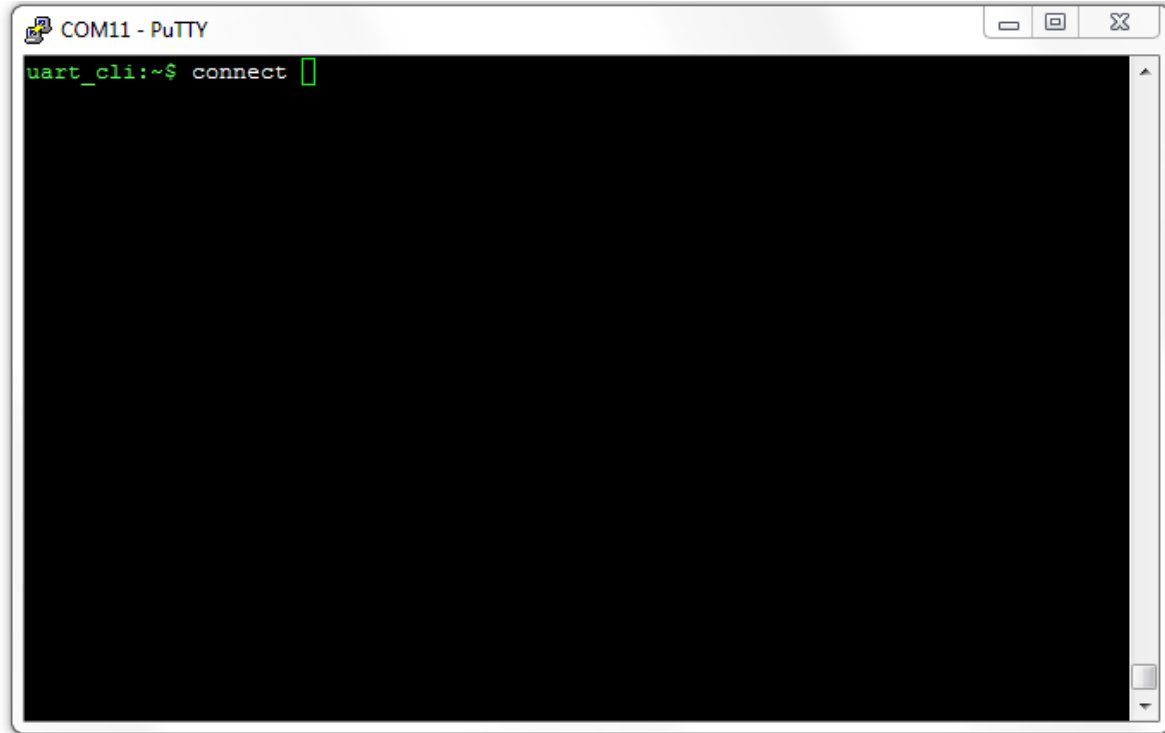
    err = terminal_size_get(shell);
    if (err != 0) {
        shell_fprintf(shell, SHELL_WARNING,
                      "No response from the terminal, assumed 80x24 "
                      "screen size\r\n");
    }
}
```



# Dynamic commands – use case

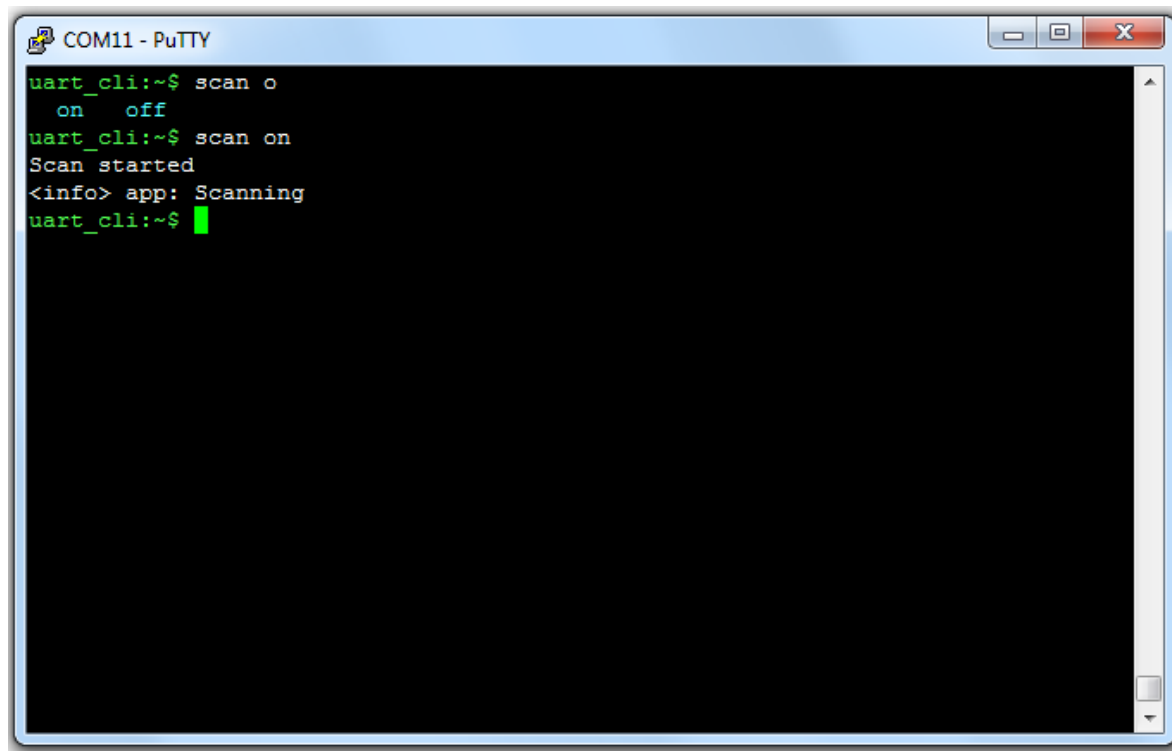


# Dynamic commands – use case



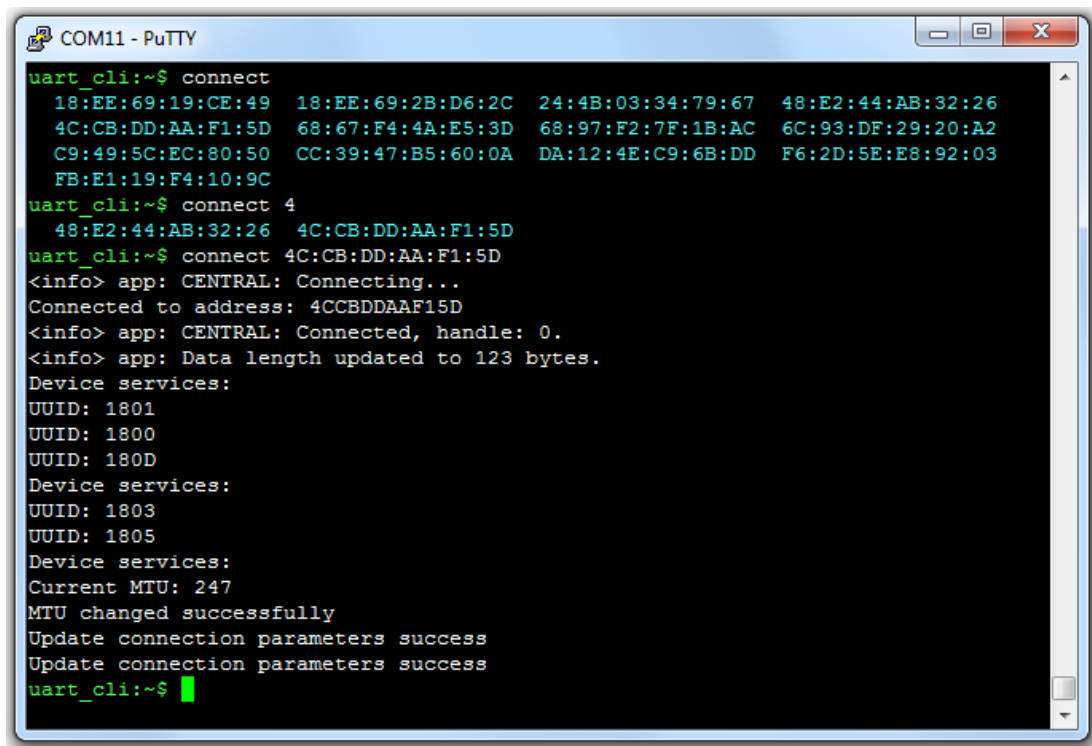


# Dynamic commands – use case



```
COM11 - PuTTY
uart_cli:~$ scan o
on off
uart_cli:~$ scan on
Scan started
<info> app: Scanning
uart_cli:~$
```

# Dynamic commands – use case

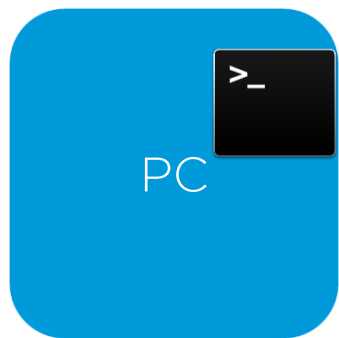


```
COM11 - PuTTY
uart_cli:~$ connect
18:EE:69:19:CE:49 18:EE:69:2B:D6:2C 24:4B:03:34:79:67 48:E2:44:AB:32:26
4C:CB:DD:AA:F1:5D 68:67:F4:4A:E5:3D 68:97:F2:7F:1B:AC 6C:93:DF:29:20:A2
C9:49:5C:EC:80:50 CC:39:47:B5:60:0A DA:12:4E:C9:6B:DD F6:2D:5E:E8:92:03
FB:E1:19:F4:10:9C
uart_cli:~$ connect 4
48:E2:44:AB:32:26 4C:CB:DD:AA:F1:5D
uart_cli:~$ connect 4C:CB:DD:AA:F1:5D
<info> app: CENTRAL: Connecting...
Connected to address: 4CCBDDAAF15D
<info> app: CENTRAL: Connected, handle: 0.
<info> app: Data length updated to 123 bytes.
Device services:
UUID: 1801
UUID: 1800
UUID: 180D
Device services:
UUID: 1803
UUID: 1805
Device services:
Current MTU: 247
MTU changed successfully
Update connection parameters success
Update connection parameters success
uart_cli:~$
```

# Future

- Changing shell Rx to be interrupt driven
- Adding ring buffer for Rx
- Migrating examples using legacy shell
- Add more backends: RTT, BT, USB ...

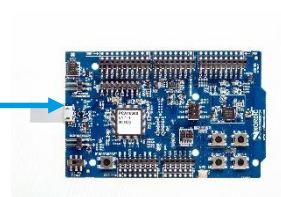
# Zephyr demo



UART

1 log backend:

- UART shell as log backend



Logger

# Logger features

- Deferred – time consuming operations deferred to known context
  - 4us (cortex-m4 @64MHz) to log a message
- Support for **multiple backends** (up to 9) – 1 in master, 3 in PRs
- Compile time filtering
- Independent **runtime filtering**:
  - per backend
  - per module
  - per instance (optional)
- Panic support - flushing logs synchronously
- Designed with **multi-domain** system in mind

# Future

- More backends like:
  - Shell (PR), RTT(PR), native posix (PR)
  - Crashlog (filesystem, flash)
  - Wire/wireless (network?, BLE?)
- Mutli-domain support
- Changing shell Rx to be interrupt driven
- Adding ring buffer for Rx

# nRF5 SDK demo

