Advanced Internet and IoT Technologies Assignment 03

HAW Hamburg WS 2026

Prof. Dr. Thomas Schmidt, Michel Rottleuthner, M.Sc. Deadline: 9. December 2026

1. **Flash an IoT Device**

   With this task we will start programming our own IoT devices.

   For that we first look at the source code of two example applications that cover the basics.

   Tools: `git, make`

   (a) Create a GitHub account if you don't have one already.
   (b) Fork the upstream RIOT-OS repository (`https://github.com/RIOT-OS/RIOT`) with your account.
   (c) Add a new git remote for your own repository in the existing RIOT folder.
   (d) Run the following command form the RIOT root direcotry to compile, flash and run the gcoap test application on your IoT device:

   ```
   CFLAGS+="<your lab configs here>" BOARD=adafruit-feather-nrf52840-sense \
   make -C examples/networking/coap/gcoap all flash term
   ```

   (e) Explore the RIOT shell on the IoT device and get familar with available commands.
   (f) Look at the `*.c` files of the example application and discuss how it works.
   (g) Flash and explore the *examples/basic/saul* firmware.

2. **Manage your Code**

   To build your own IoT firmware you will need a clean development environment. Follow the given structure and always commit all your changes to your repositories to be able to track any changes during development. Later assignments will build ontop of the code so it pays off to keep everything organized.

   Tools: `git`

   (a) Create a separate repository for your own application that references your RIOT fork as a git submodule.
      i. Your new application will reside in a separate folder *outside* of the RIOT code base.
      ii. Changes that affect the RIOT code base should be comitted to your fork — however, this should not be necessary for now.

3. **Program Custom IoT Firmware**

   Create your own firmware that provides CoAP access to all peripherals (sensors/LEDs) of the IoT node.

   Tools: `git, make, C`

   (a) Program shell command handlers that allow to manually issue CoAP requests via the serial terminal of the IoT node (see the gcoap example).
   (b) Program additional handlers for accessing all sensor values via the SAUL interface.
   (c) Use the CoAP commands to read out sensors of other teams.
   (d) Use the CoAP commands to change the LED color of other teams.