Workshop scenario „Internet of Things for everyone”

http://fundacjaantares.pl/iot.html

Co-funded by the Erasmus+ Programme of the European Union

Workshop scenario „Internet of Things for everyone”

Class 1: Introduction to the Internet of Things. Basics of electronics and programming.

Aims:
- familiarising with the idea of the Internet of Things
- increasing curiosity of the participants and their creative thinking
- raising a positive attitude towards new technologies
- familiarising with electronic systems and basics of programming

Materials:
- sticky notes, pens
- flipchart or a large piece of paper
- educational film: https://www.youtube.com/watch?v=LlhmzVL5bm8
- pictures of sensors (annex 1)
- components – 1 set per group of 2-3 people: NODEMCU ESP8266 board, female-female cables, USB cable, diode. Some boards may be prepared before for visually impaired people by adding colourful dots next to pins that should be connected.
- computer per group (with Internet access and arduino1.8.1. preinstalled)
- projector.

Arduino1.8.1 installation:
- installation file can be downloaded from:
https://www.arduino.cc/en/main/OldSoftwareReleases
- the program should be installed with the shortest possible path: on Desktop or C:/ disk
- do not update
- select the USB port where the board is connected: Tools -> Port
- select the NODEMCU board: Tools -> Board-> NodeMCU 1.0

Class schedule:

1. **Introduction:**

Introductory presentation: the educator and the project, aims of the workshops, topics of each class.

Expectations of the participants. Give sticky notes and pens to the participants. Ask them to write 1-2 reasons why they came to class and what are their expectations towards the workshops. Stick the notes to a flipchart, group by expectations and motivations. Leave the paper for the last, evaluation class.

Discussion, sharing experience. Helpful questions:
- How often do you use Internet? Why?
- Did you hear about the Internet of Things?

- What is the main idea of a „smart house/smart fridge”?
- What advantages and disadvantages stand behind such solutions?

2. What is the Internet of Things:

Introductory film is available here: https://www.youtube.com/watch?v=LlhmzVL5bm8
You can choose your national language and make the subtitles large.

Discussion about the film. Ask the participants about their impression. What do they think about the film and the Internet of Things idea?

Using sensors. Work in groups. Divide the participants into groups of 2-3 people. Each group gets pictures with few sensors (cut out from the Annex 1). Ask the participants to discuss in groups possible usage of those sensors in a daily life/industry/communication. At the end of exercise each group presents its ideas.

3. Basics of electronics and programming:

Basics of electronics. Work in groups.
Give a NODEMCU ESP8266 board to each group and ask if the participants know what it is for. Explain that it is an electronic system that can send and receive information to/from different components. It is equipped with a WI-FI module so it can be connected not only to a computer but also to a local network or the Internet.
Distribute USB cables, female-female cables and diodes to every group. Explain that we will be wiring components together. The USB cable should be connected to a computer and a NODEMCU ESP8266 board (Fig. 1). Make sure there are no other devices plugged to the other USB ports (e.g. mobile phones).

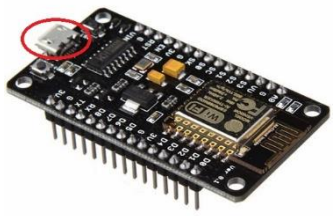


Fig. 1. Port for mini-USB cable on the NODEMCU ESP8266 board.

LED diode can be connected to the board by female-female cables. The shorter leg of a diode should be connected to the ground (G pin), the longer – to a digital pin (e.g. D1) as it is shown on Fig. 2. For visually impaired people you can mark the pins with colourful dots made from a sticky paper because the letters on the board are very small.

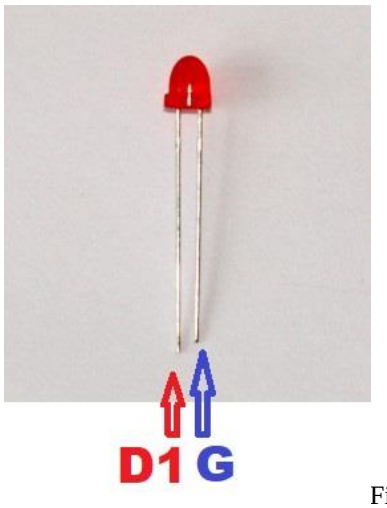


Fig. 2. Wiring a LED diode to the NODEMCU ESP8266 board.

Basics of programming. Group work.
Presentation of the TUNIOT tool. The tool is available on *easycoding.tn*. Tell the participants about the inventor of the tool and the 'IoT for everyone' project. Put on projector the *easycoding.tn* webpage so everyone can follow changes in the code. Select the desired language version of the TUNIOT FOR ESP8266 on the left side toolbar (Fig. 3). Explain what a block programming is: from the menu on the left side you can choose pieces of a code which you can put in the middle panel. The main program is divided into two parts: 'Setup' which is executed only once and 'Main loop' – this part of the code will be repeated.

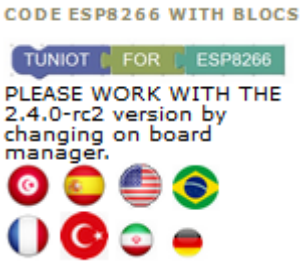


Fig. 3. Hyperlink to the TUNIOT tool on easycoding.tn

Let's try to write our first program. It will turn on a LED diode. From **IN/OUT** menu choose **Digital** option because our diode is connected to a digital pin. Explain what the difference between digital and analog pin is. Digital pins receive and send zero-one signals like turn on – turn off. Only two states are possible: diode is ON or it is OFF. Some sensors can be connected to analog pins which give a broader range of information, e.g. we can measure temperature, humidity or luminosity in a range of values.
Then, choose **DigitalWrite PIN# D0 STAT HIGH** block and drag and drop it in the main program under the **Main loop** block (Fig. 4). The block should stick to the code with a characteristic sound. Check if the participants managed to do it. Inside that new piece of code we can change two things: pin number and its state. In the first case we choose number of the pin where the diode is connected (e.g. D1). The second part of the block should be set as 'HIGH'. 'HIGH' means that the diode is either ON, 'LOW' – it is OFF. The code for turning the diode on is ready now. Explain to the participants that the blocks are translated to a language that computers understand, in this case – C language. You can see how it looks like by clicking the 'CODE' tab (Fig. 5). The first icon of the upper menu allows you to copy the code (Fig. 5, blue circle).

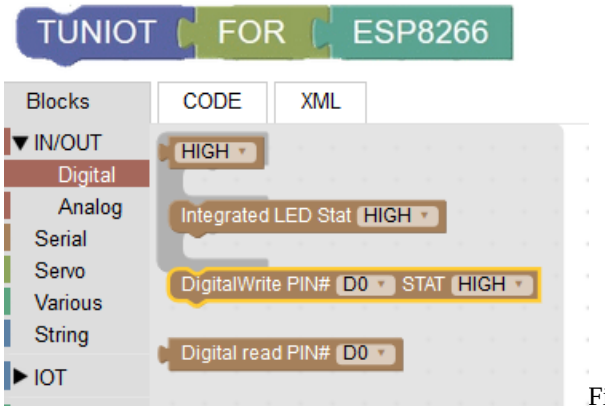
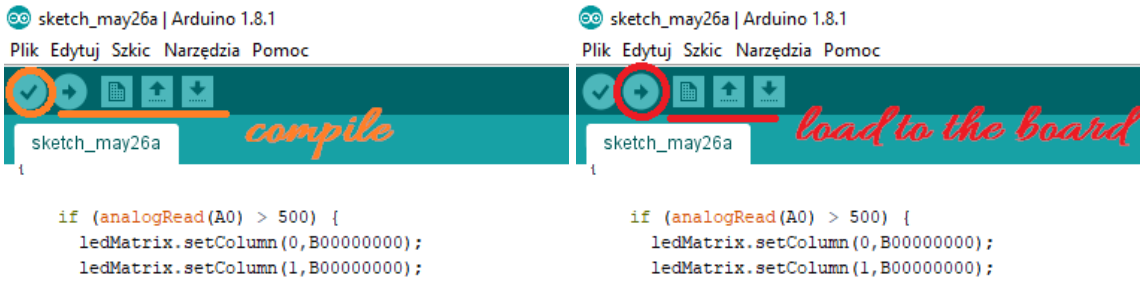


Fig. 4 Block allowing to send a signal to a digital pin with TUNIOT tool.

Fig. 5. A view of the TUNIOT tool with the C code and the copy option marked.

Paste the code to the arduino1.8.1 program. You can use Ctrl+A and Ctrl+V keybinds. Each program should be compiled and loaded to the NODEMCU ESP8266 board. You can do it with the first two icons of the main menu (Fig. 6). When the loading is over, the diode should turn on. If that is not the case, check the wiring, the code and try to load once more.



Rys. 6. Compiling and loading buttons in the arduino1.8.1 program.

4. Closing:

Ask the participants what their impressions after the first class is. What was the greatest difficulty/challenge? What do they remember about the Internet of Things? Can they explain the idea? Announce the topic of the next class.

Homework for participants: explain to your friends and family the idea of the Internet of Things. Where can they find it useful in the modern world?

Class 2: First embedded systems with TUNIOT

Aims:
- familiarise with basics of programming
- explaining programming constructs and their implementation in daily life

Materials:
- components per participant: NODEMCU ESP8266 board, female-female cables, USB cable, diode set. Some boards may be prepared before for visually impaired people by adding colourful dots next to pins that should be connected.
- computer per participant (with Internet access and arduino1.8.1. preinstalled)
- projector.

Instruction of preparing a set of diodes:

Five diodes in different colours should be connected to the common ground with soldered wires. Add resistors between diodes and the ground pin. There should be pins added next to each diode (Fig 7).

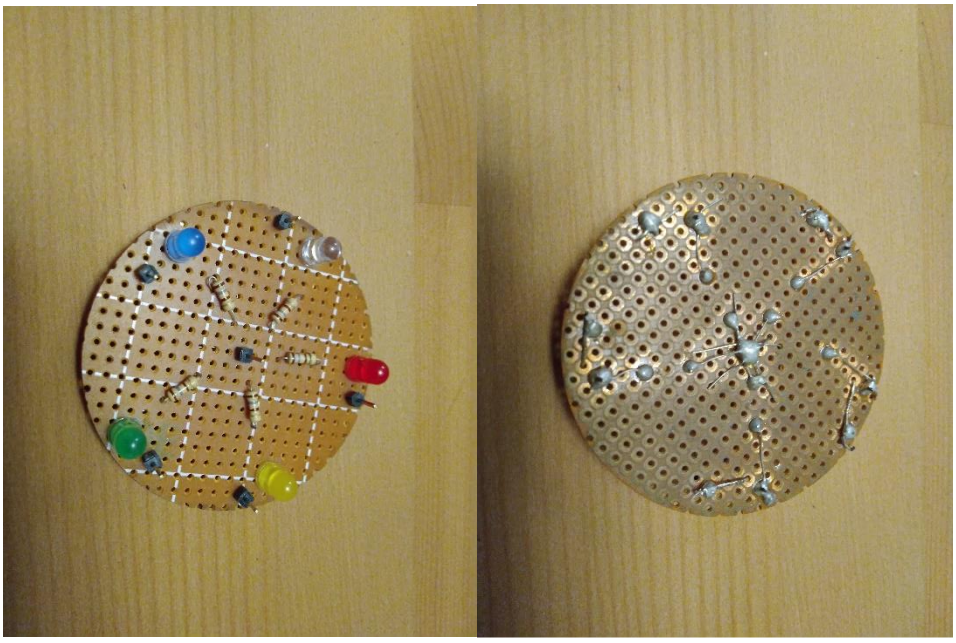


Fig. 7. Set of diodes made for workshops.

Class schedule:

1. **Introduction:**

Introductory discussion. Reminding the last class. Discussion about the Internet of Things and the homework. What examples of the Internet of Things were found by the participants? Reminding the steps of working with the TUNIOT tool.

2. Simple embedded systems:

Exercise 1. Reminding the exercise from the previous class about turning a diode on. Distribute the NODEMCU ESP8266 boards, USB cables, female-female cables and sets of diodes among participants. Remind the wiring of the components and the steps taken during the code preparation in the TUNIOT tool. Can the participants work unassisted with this task? Try to give some tips without revealing the solution.

Exercise 2. Make a diode blink (turning on and off by itself). Tell the participants to do the task unassisted. They will probably add the piece of code about turning off only. There is nothing wrong with the solution but you should explain that the time interval between turning on and off is so small (in the order of milliseconds) that a human eye cannot catch it. The diode seems to not be blinking at all. You need to add a delay after the turning on and off pieces of code. You can find it in **Various -> Delay Ms**. The delay is set in milliseconds. The participants can try different values of the delay and check how the diode blinks. An exemplary code is presented in Fig. 8.

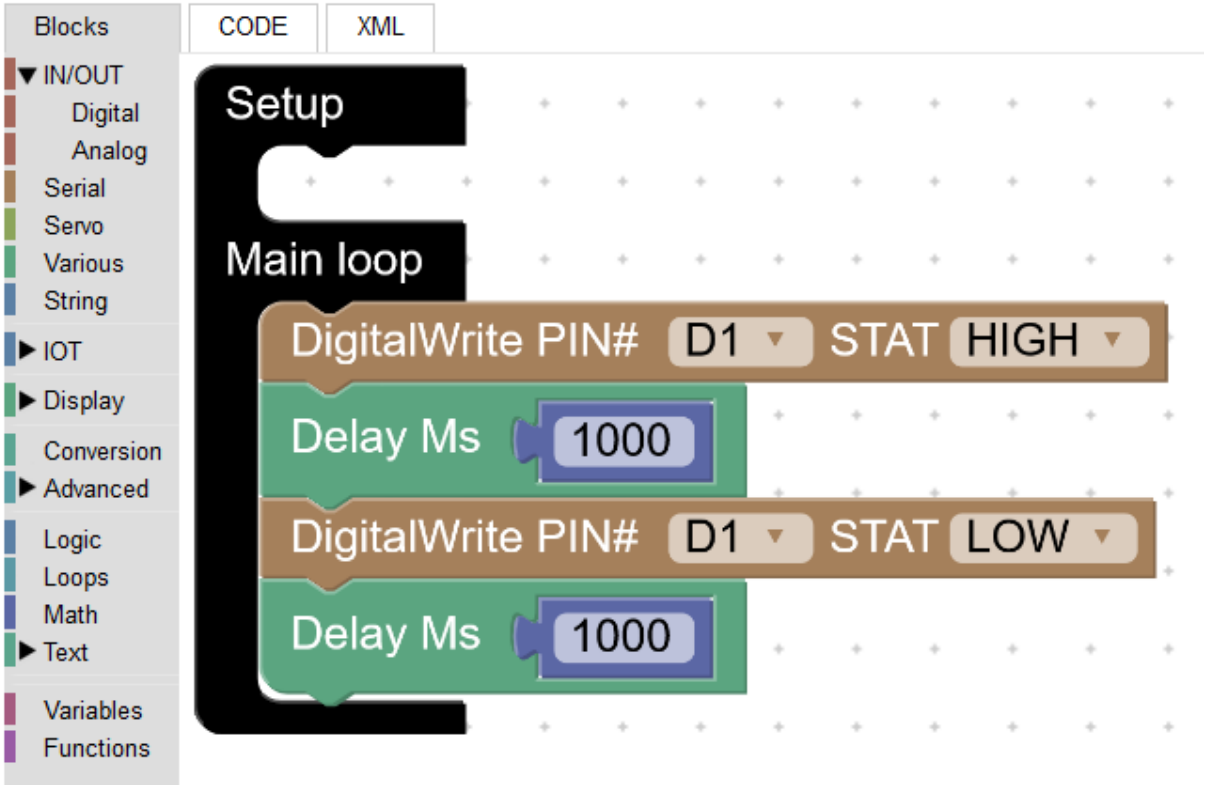


Fig. 8. Block code to the exercise 2.

Exercise 3. Repeat the exercise 2 but for 2 diodes (e.g. red and green) blinking synchroniously (Fig. 9). Remember to correctly connect the next diode to a digital pin.

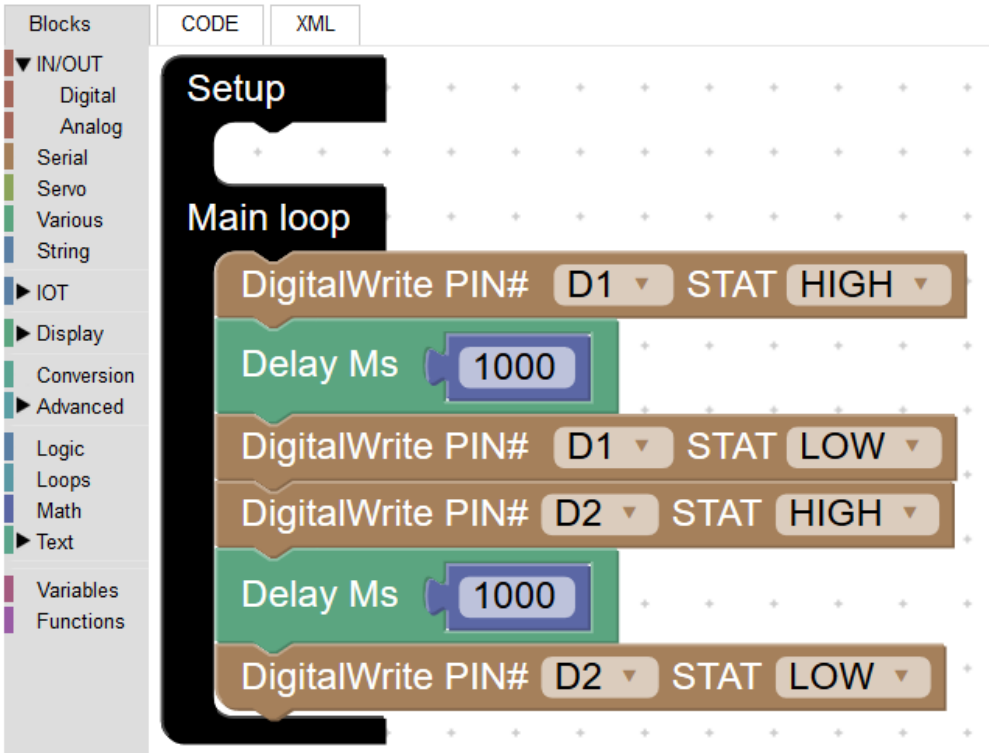


Fig. 9. Block code to the exercise 3.

Exercise 4. Modification of the exercise 3 – system of 2 diodes where one is blinking with a different speed than the other (Fig. 10).

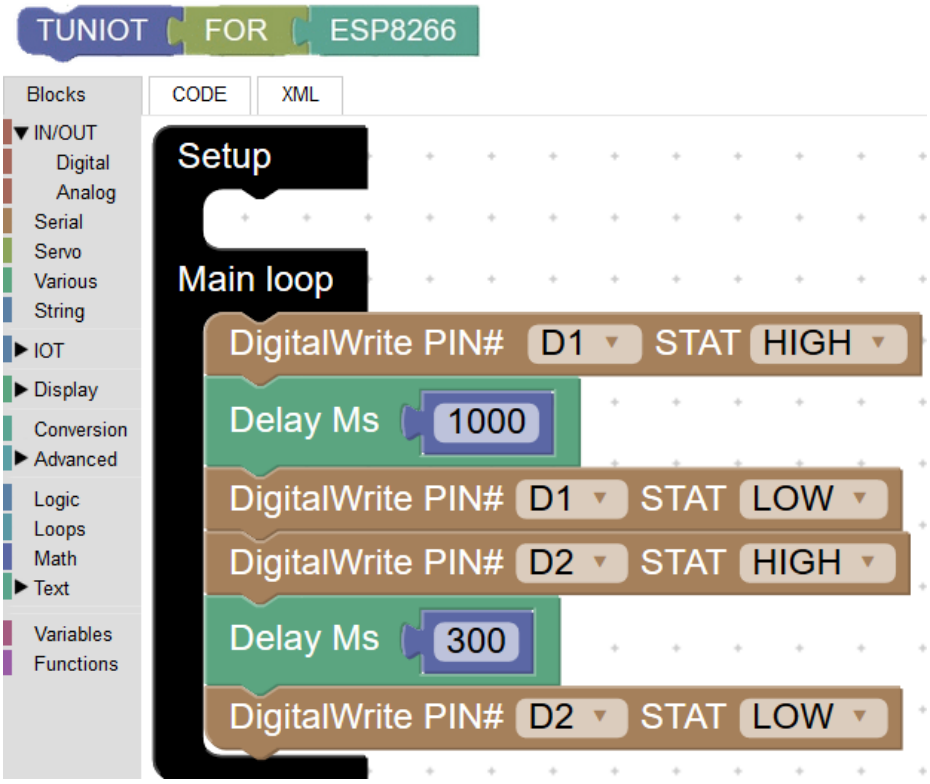


Fig. 10. Block code to the exercise 4.

Exercise 5. Imitation of Christmas tree lamps. Add the third (e.g. yellow) diode. Each diode blinks with the same speed but after the previous diode turns off (Fig. 11).

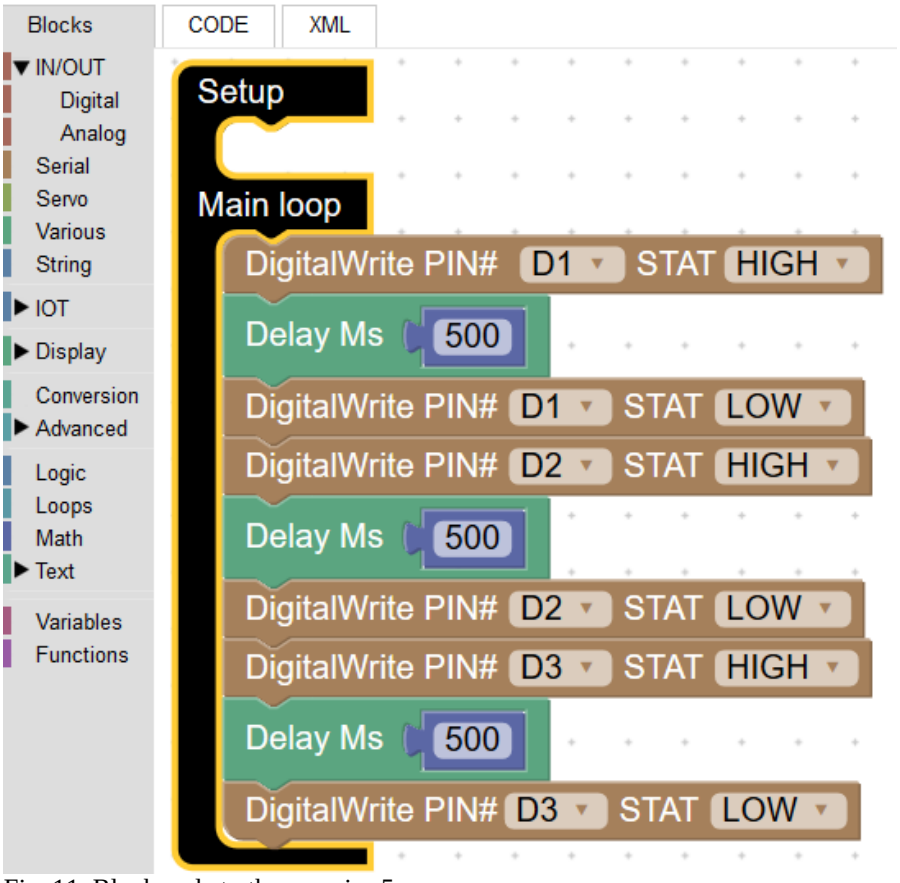


Fig. 11. Block code to the exercise 5.

Exercise 6. Turn the first diode on three times, then the second one three times and the third (Fig. 12). Explain the concept of loops. Code in the loop will be executed three times (from 1 to 3 with a step of 1). The loop block can be found in: **Loops -> count with**…

Exercise 7. Imitation of traffic lights. Tell the participants to try and make the task unassisted. Firstly, turn the green diode on for 5 seconds, then let the yellow diode blink for 3 seconds. In the end turn the red lamp on for 5 seconds (Fig. 13).

Exercise 8. Modification of the exercise 7. Make two traffic lights: the one for cars, the second for pedestrians (Fig. 15). When the green and yellow light for cars is on, the red (or another colour) light for pedestrians should be on. When the red lamp for cars is on, turn on the lamp allowing pedestrians to walk across a street.

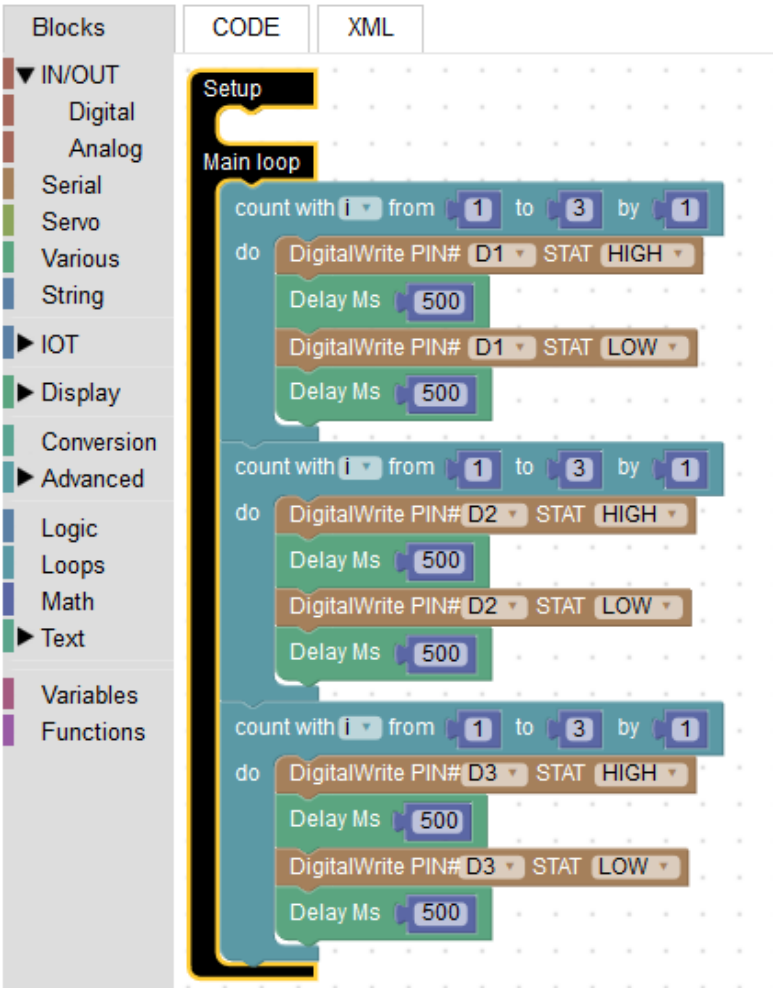


Fig. 12. Block code to the exercise 6.

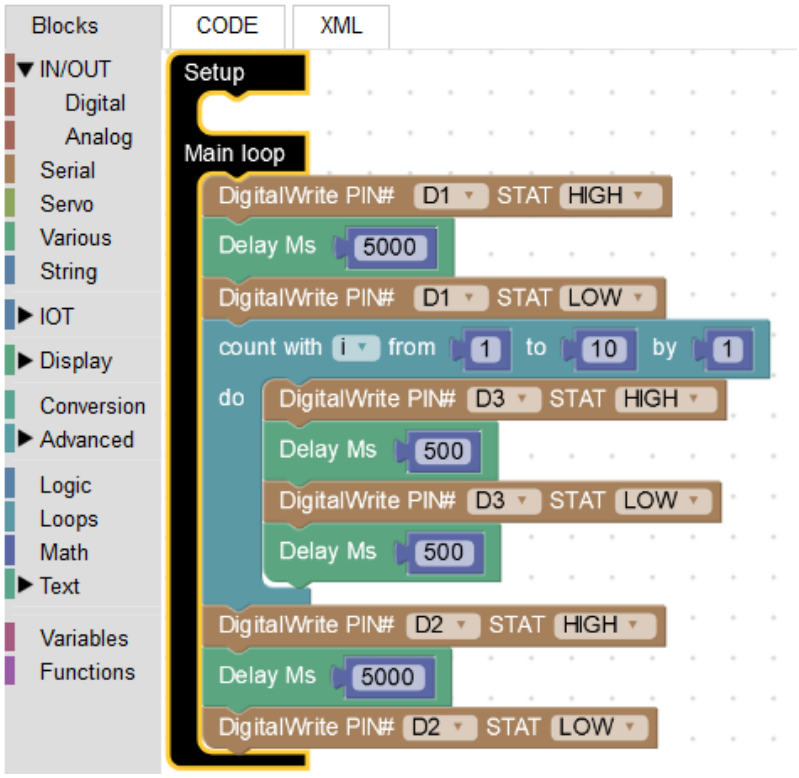


Fig. 13. Block code to the exercise 7.

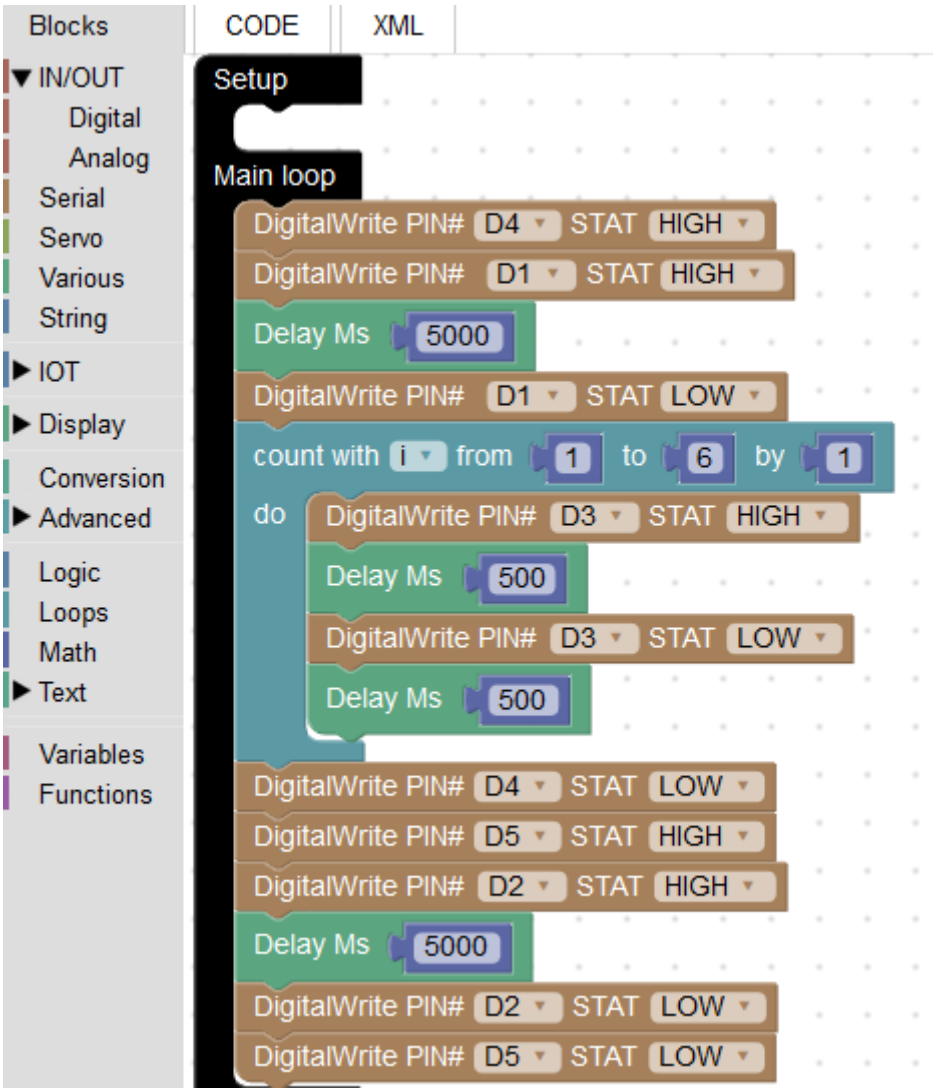


Fig. 14. Block code to the exercise 8.

3. Closing:

Ask the participants which of the exercises was the most challenging and which they liked the most. Do they have an idea for a similar task? How confident do they feel working with the TUNIOT tool and electronic components? Announce the topic of the next class.

Homework for participants: ask your friends and family (or search on the Internet) what kind of programming constructs can be implemented in a code.

Class 3: Work with LED screen and soil moisture sensor

Aims:
- familiarise with LED screens and soil moisture sensors
- learning about conditional statement
- embedded systems preparation for calibration in home environment.

Materials:
- printed objects from annex 3 and colourful pens/pencils,
- components per participant: NODEMCU ESP8266 board, female-female cables, USB cable, LED matrix, soil moisture sensor. Some boards may be prepared before for visually impaired people by adding colourful dots next to pins that should be connected.
- a computer per participant (with Internet access and arduino1.8.1. preinstalled)
- projector
- calibration sheets (annex 2)
- a glass of water per participant.

LED matrix library installation instruction:

Install the MAX7219Led Matrix library before the class. You can find it on *easycoding.tn* webpage (Resources tab). Choose **Sketch -> Add library (MAX7219Led Matrix)** in arduino1.8.1 program. Leave the program open for the participants.

Class schedule:

1. **Introduction:**

Introductory discussion. Reminding the last class. Discussion about the participants' finds. What kind of programming constructs have the participants found?

Explaining conditional constructs on examples.
Exemplary conditional constructs:
- „If it is raining, take an umbrella with you" – a simple conditional construct (if)
- „If it is raining, take an umbrella with you; if the sun is shining, walk the dog" – conditional construct with two conditions (else if)
- „If it is raining, take an umbrella with you; if not (it is not raining), walk the dog" – conditional construct negation of the first condition (else)

Ask the participants what the difference between the second and the third sentence is. When exactly the dog will be walked? What if it is not raining but cloudy?

Example – a joke:
„Wife says to her husband-programmer: Go to a shop, buy a bread. If they have eggs, buy 10"
Question to the participants: what did the husband-programmer buy?
Answer: 10 loaves of bread.
Explain how computers „think". They do exactly what we tell them to do, not always what we mean.

A task with colouring objects (shapes). Distribute annex 3 paper and colourful pens/pencils to the participants. Ask them to colour the objects according to the following instructions:

- If an object has 3 sides, colour it red
- If an object has 1 right angle, colour it blue
- If an object has a right angle, mark it with green colour, if it has not - mark it with blue colour
- If an object has a right angle and an acute angle, colour it green
- If an object has 4 right angles and a circle, colour it orange, if it has 4 right angles and 2 acute angles, colour it yellow
- If an object has 4 sides or an internal square, colour it brown.

Explain how the instructions above are understood by a computer. The answers can be found on Fig. 15.

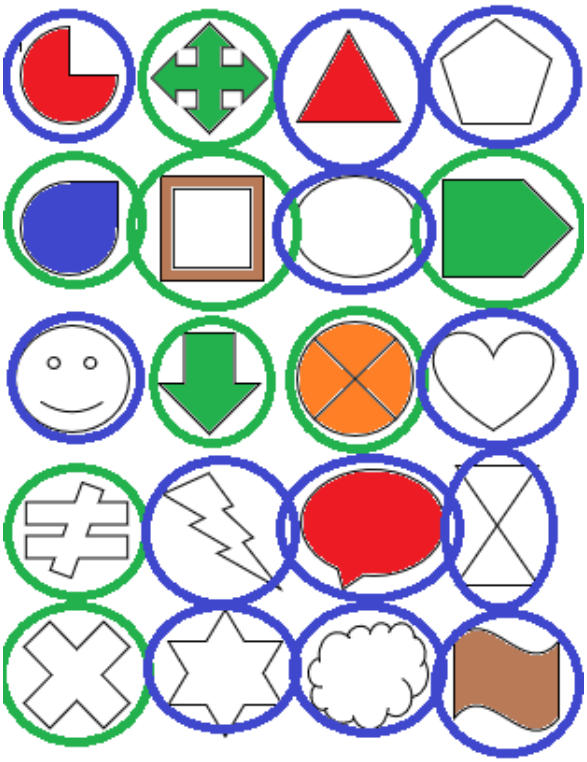


Fig. 15. Answers to a task with conditional constructs.

2. Work with LED screens:

Exercise 1. Distribute the NODEMCU ESP8266 boards, USB cables, female-female cables and LED matrix among participants. Each participant should have their own set. The wiring of a LED matrix as follows:

VCC – 3V
GRD – G
DIN – D7
CS – any digital pin
CLK – D5

You can prepare the sets for visually impaired people by adding colourful dots next to pins that should be connected.

The task consists of printing a name or a short word on the LED screen. In the TUNIOT menu choose **Display** and then **MAX7219** and the **MAX7219 INIT Number of devices: 1 PIN [the digital pin where the CS pin is connected]** block. Drag and drop to the **Setup** part. In the **Main Loop** set **Display -> MAX7219 -> MAX7219 Display dot matrix:** blocks one

under the other. Each block consists of one letter of the name/word. You should unchecked the fields that you do not want to print. Remember about a delay between pieces of the code (Fig. 16).

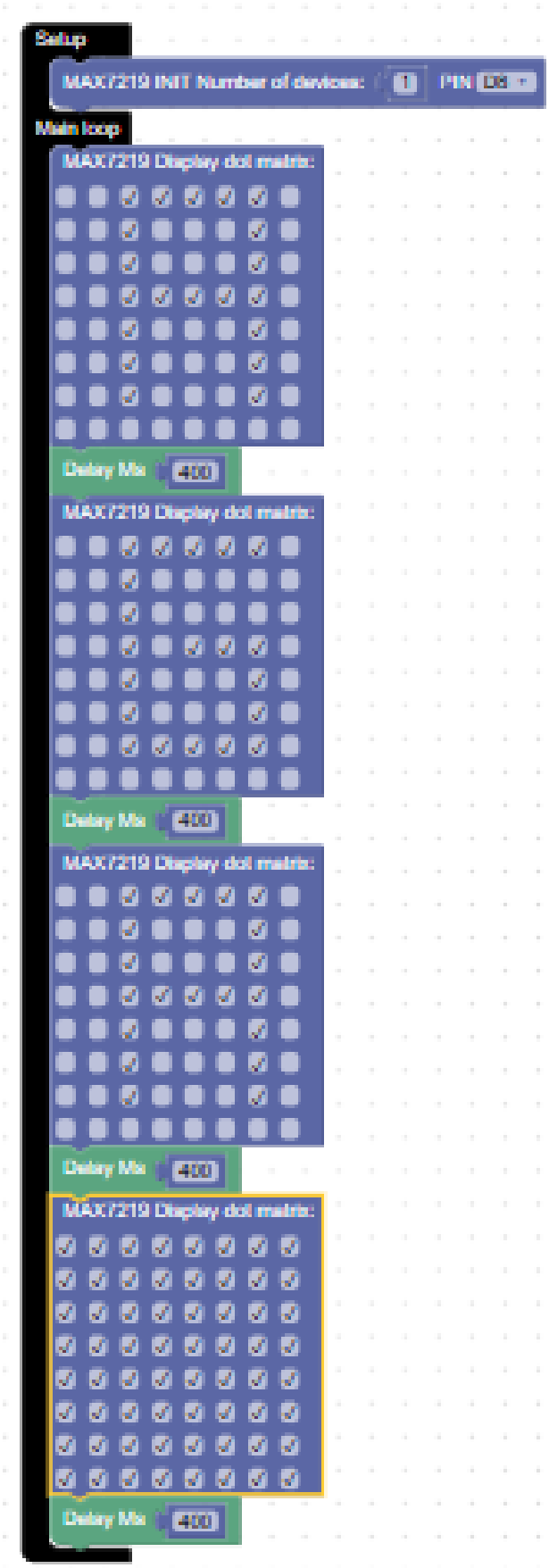


Fig. 19. Block code to the exercise 1.

Exercise 2. It is a modification of the previous task. This time we will print a short sentence

(e.g. „Hello world”). Add the **Set text** (from **MAX7219** menu) block with the sentence in the **Setup** part. The text block can be found in the **Text** menu. Clear the **Main Loop** and place the **Scroll Text** block there (leave the Left option) and the **Draw Text** block. Then set a short delay and add two more blocks: **Commit** and **Clear** (Fig. 17).

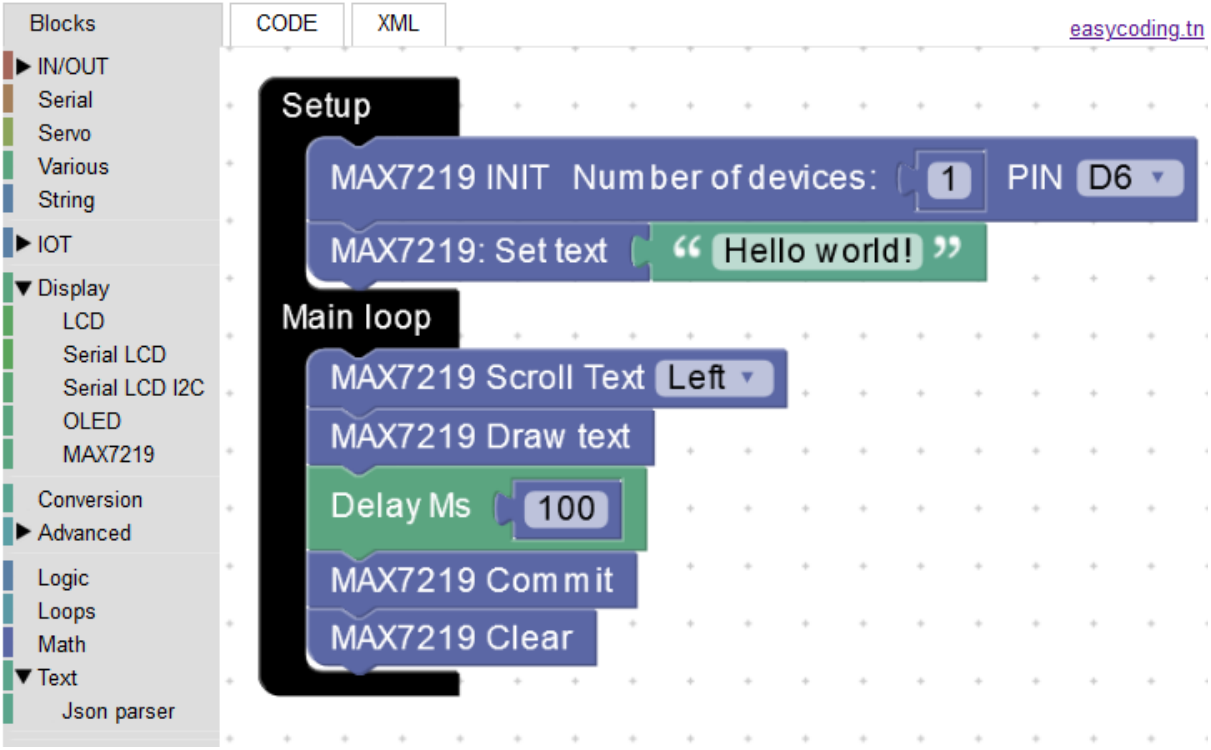


Fig. 17. Block code to the exercise 2.

3. Work with soil moisture sensors:

Exercise 3. Printing the soil moisture sensor reading on the computer. Distribute soil moisture sensors and glasses of water to the participants. They should not disconnect the LED matrixes. The wiring of the soil moisture sensor is as follows:
A0 – A0
D0 – any digital pin
GND – G
VCC – 3V
Remind what the difference between digital and analog sensors is. Digital sensors give zero-one signal (or something is ON or OFF). Analog sensors return value in a certain range. In the case of a soil moisture sensor, it will return a value from 0 to the upper limit depending on measured humidity. The upper level can differ between sensors. The first task consists in checking what the limiting values are. Choose **Print on new line block** from **Serial** menu and drop it in the Main Loop. Uncheck the text part of the block and place Analog read PIN A0 (from IN/OUT -> Analog) instead. Then set a short delay, e.g. 3 seconds (Fig. 18). Copy-paste the code to the arduino1.8.1 program and execute in a standard way. Ask the participants to read out the value of their sensors. It is printed in the **Serial Monitor** which can be opened by clicking on the magnifier icon in the upper right corner (Fig. 19). Ask them to put the sensors into glasses of water and check the value again. What is an intermediate

value between wet and dry conditions?

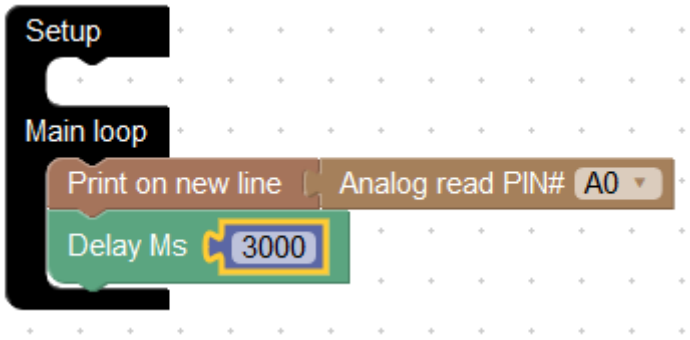


Fig. 18. Block code to the exercise 3.

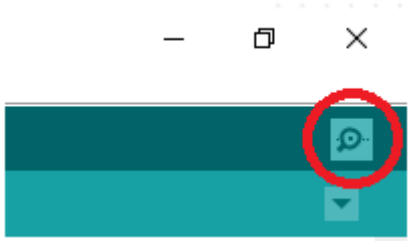


Fig. 19. The Serial Monitor icon in arduino1.8.1 program.

Exercise 4: Lets print a sad or a happy face on the LED matrix depending on the soil moisture sensor value. In the TUNIOT tool set the **MAX7219 INIT Number of devices:** block in the **Setup** part. In the **Main Loop** design a condition of displaying sad/happy face depending on exceeding the intermediate value of the sensor which was found in the previous task (Fig. 20). Remember about a delay.

Exercise 5: Modify the exercise 4 but display the exact value on the LED screen this time. In the **Setup** set **Declare i as String Value** block from the **String** menu and add here the **Analog read** block. Then, set the printed text as „i" variable (also from the **String** menu). In the **Main Loop** set a **repeat 30 times** with the content similar to Exercise 2. This is a proper value to print a 3 digit reading once. Under the loop set **set String i to** with the **Analog read** block and set the printed text as „i" variable once more (Fig. 21). You can set a bit longer delay (e.g. 10 seconds).

4. **Closing:**

Distribute the power banks among the participants. Ask them to connect it to the NODEMCU board. Now the system should work independently of a computer. Check if it works for everyone. Distribute the calibration sheets (annex 2) and explain the calibration task. The participants will measure the soil moisture of a chosen plant for the next few days in a few hours intervals. The task aims to find the limiting value which indicate the time when the plant needs water. Calculate the mean or an intermediate value from all the measurements. Announce the topic of the next class.

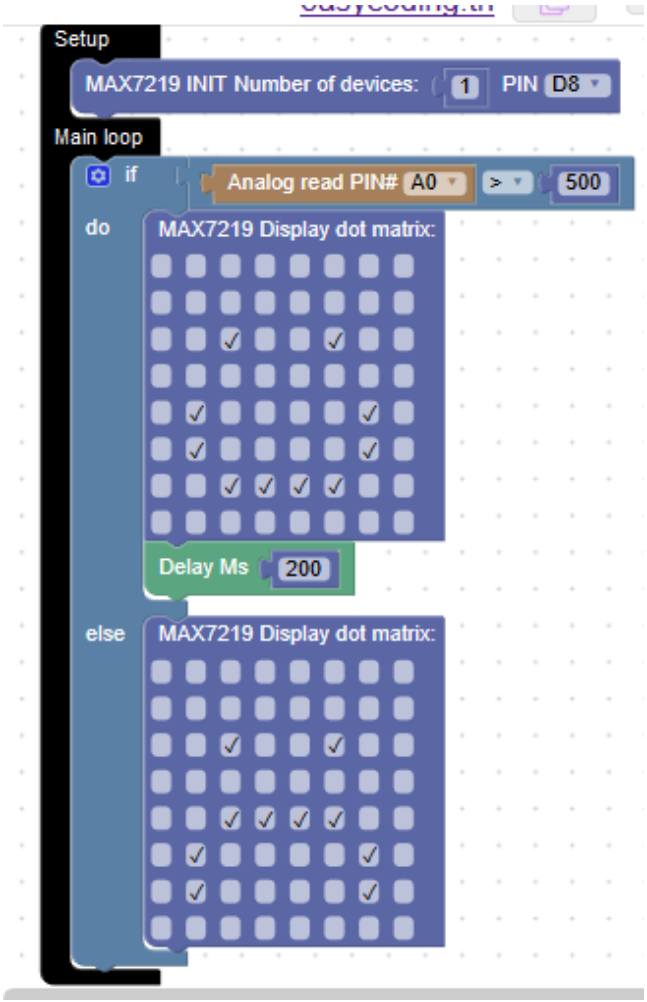


Fig. 20. Block code to the exercise 4.

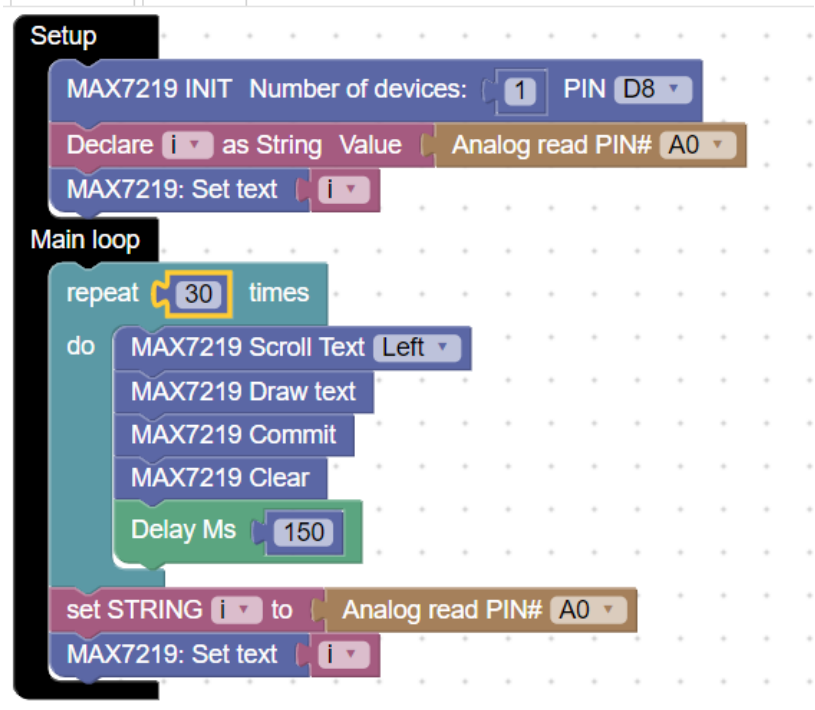


Fig. 21. Block code to the exercise 5.

Class 4: Client-server communication

Aims:
- familiarising the participants with client-server architecture and examples of using this architecture
- familiarising the participants with advantages of working in a network
- learning a block code to communication in a local network
- preparation to calibration soil moisture sensors in home conditions.

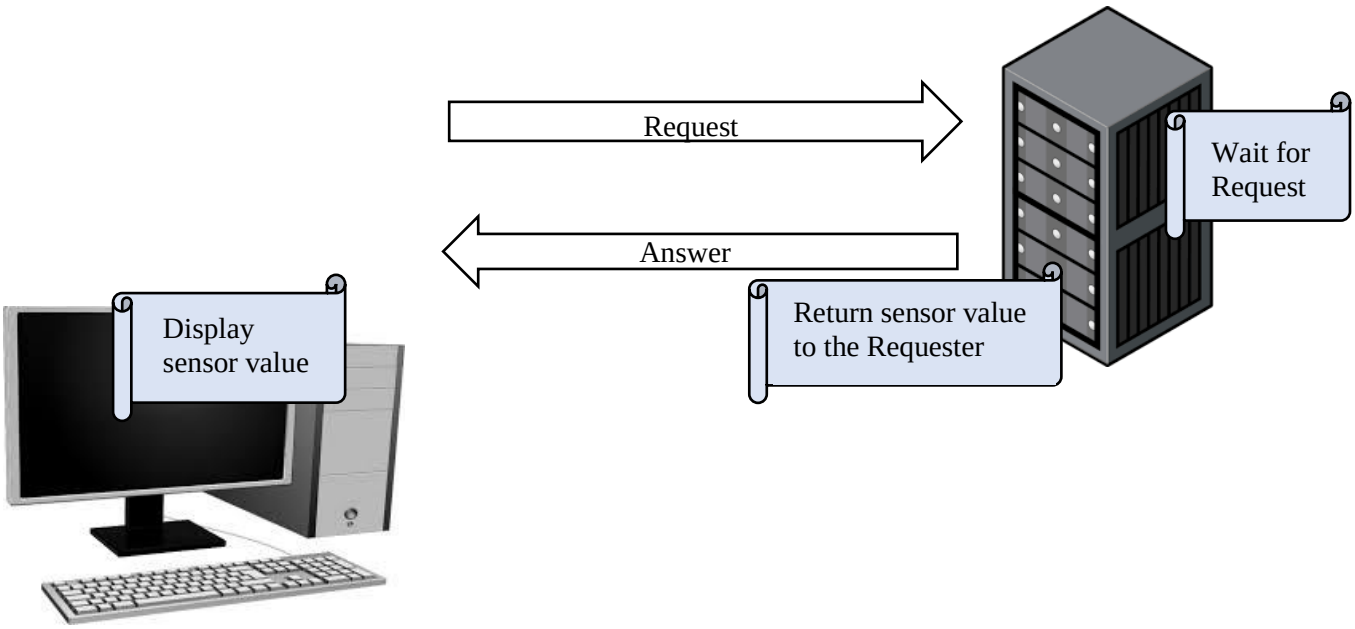
Materials:
- components per participant: NODEMCU ESP8266 board, female-female cable, USB cable, soil moisture sensor, power bank, set of diodes.
- a computer per participant (with Internet access and arduino1.8.1. preinstalled)
- projector
- a glass of water per participant.

Class schedule:

1. **Introduction:**

Introductory discussion. Reminding the last class. Ask the participants how the calibration was. Did they have any problems with measurement/equipment? What values did they determine? If someone did not manage to calibrate the sensor, he/she can use the limiting value determined by another participant.

Explain what the client-server communication is. You can use an analogy of a shop assistant and a buyer. The scheme below can be useful:



What is the client/ the server in our case? A computer or a NODEMCU board?
Give some examples of client-server architecture:
- e-mail: a program allowing to open / send messages is a client of an e-mail server. Mails are stored on the server
- WWW websites: a client sends a request to the WWW server from a browser. The server replies with a WWW webpage file (a HTML file) that can be decoded by the client.

2. Connecting to a local network:

Check components brought by the participants before the new exercises start. Do they work correctly? Change components if it is needed. Are they correctly connected?

Exercise 1. Set a hotspot from your computer or mobile phone. You can also use a local WI-FI if there is not a two-stage login process (e.g. with a browser). Drag the **Disconnect** block to the **Setup** in TUNIOT. It can be found here: **IOT -> IOT Station.** This command disconnects the board from any other networks that it could be connected before. Set a short delay, ca. 3-5 seconds. This time is needed to open the Serial Monitor. Let's print „START" to check if the program works. Then, choose the **Connect Network SSID** block with or without a password from **IOT -> IOT Station**. You should give the network name in the first field and the password in the second (if the network is secured). The next step is to create a while loop where we will check if the connection succeeded. Explain how the while loop works: until the board is not connected with the local network, the program will print „ …". When the loop condition is satisfied, print a proper message (Fig. 22). Check in the Serial Monitor if the participants succeed to connect with the local network..

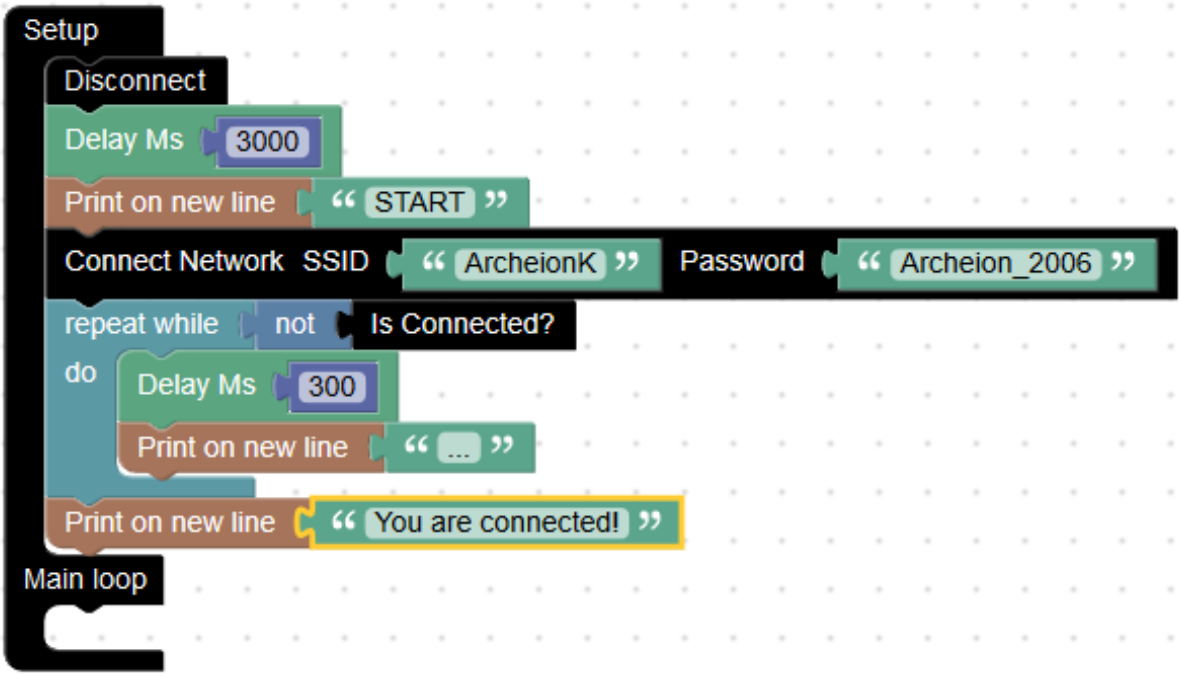


Fig. 22. Block code to the exercise 1.

Exercise 2. We will need IP address of the NODEMCU board in order to connect it to the network. Explain that it is a parameter describing a device in a network. You need to add the **Print on new line** block with **Local IP** connected to it (Fig. 23). The IP address will be printed in the **Serial Monitor**.

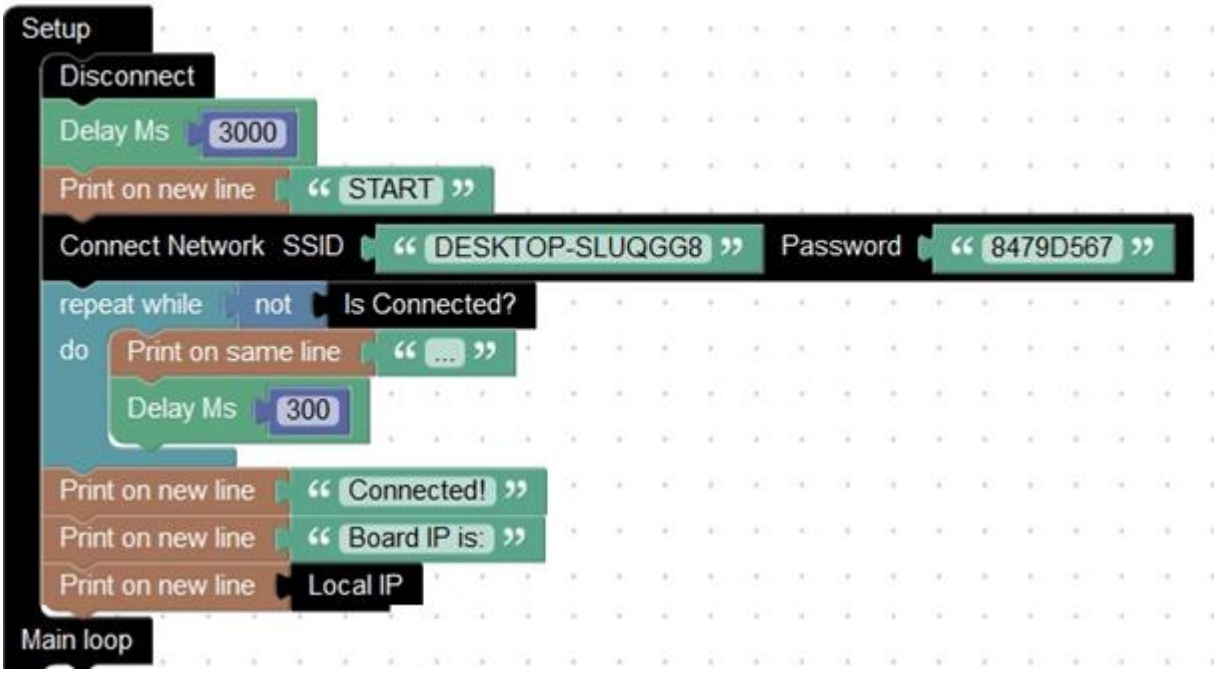


Fig. 23. Block code to the exercise 2.

Exercise 3. Our board-server will wait for a request send by a computer-client. But we need to program the board first. Select the **Start Server Port 80** block from **IOT -> IOT Server** and drop it in the **Setup** part. Do not change the port. It is a standard one. In the **Main Loop** we will check if a client sent a request. The green diode should blink this time. Set **Wait Connection** after the blinking part and a message (e.g. „Welcome"). From the same menu choose **Answer „"** block and type your message. At the end we clear the request with the **client flush** block (Fig. 24). A request can be sent by typing the IP address of the board in a browser with „/" and any sequence of letters. Keep in mind that the delay used for the blinking diode cannot be too long because the server does not „listen" for a request while executing this part of the code. Too many requests may freeze the server. In such case, press the RST button on the NODEMCU board in order to reset the program.

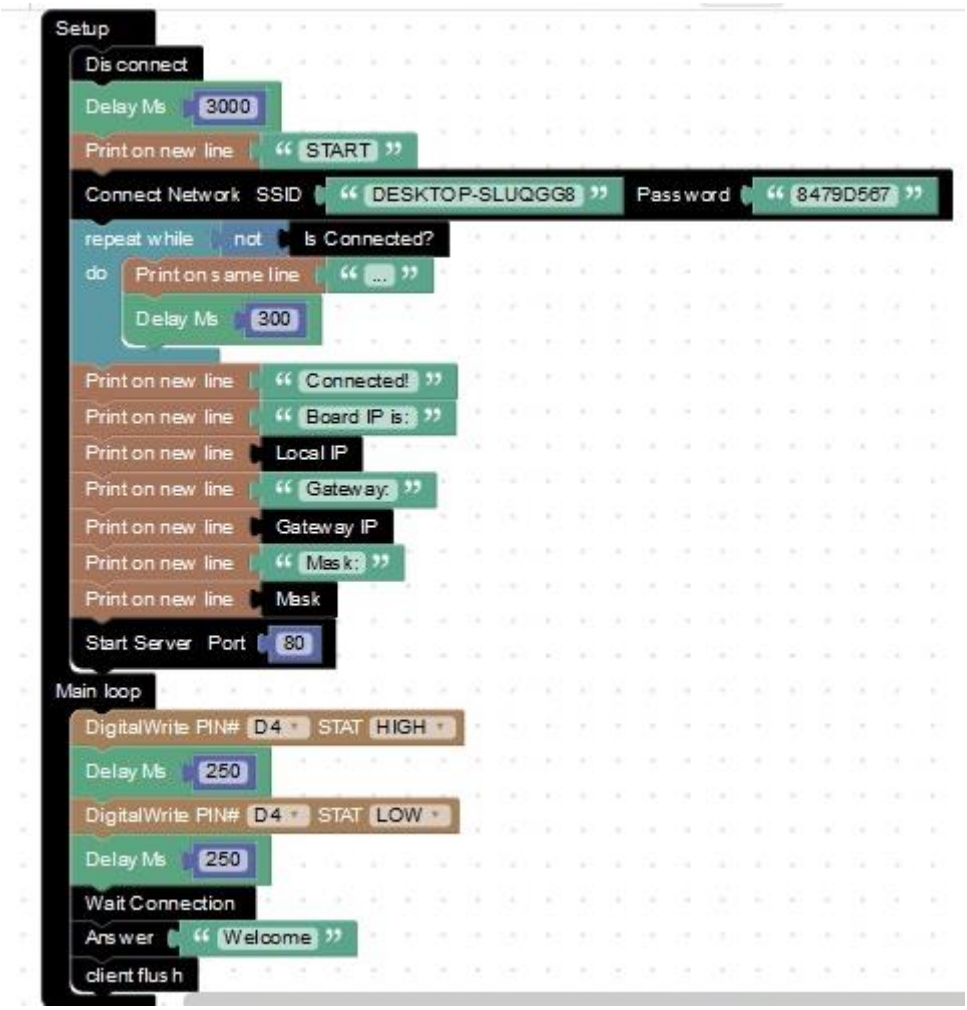


Fig. 24. Block code to the exercise 3.

Exercise 4. Modification of the previous exercise by developing the NODEMCU board message. The message will contain the „Welcome" text and the information about the soil moisture sensor reading. Distribute glasses of water among participants and remind them the wiring for a soil moisture sensor. Set a conditional statement so that the **Analog read** value is greater or equal the calibrated value. In the **do** and **else** part set **Answer Web** page block

which can be found in **IOT -> Web page**. This block allows you to modify the HTLM code of the website where you send a request. In the **Body** part set the "Welcome" message using the **Heading** block. The next block will print a message with the sensor value. You can find it in the same menu as **SPAN**. Ask the participants what message should be printed if the sensor value exceeds the calibrated one. Let the participants personalise the website. They can change a size and colour of the font, the message or add their own. They can also add a button with a hyperlink to a sad/happy plant picture. Ask them to test the code with a dry and wet sensor.

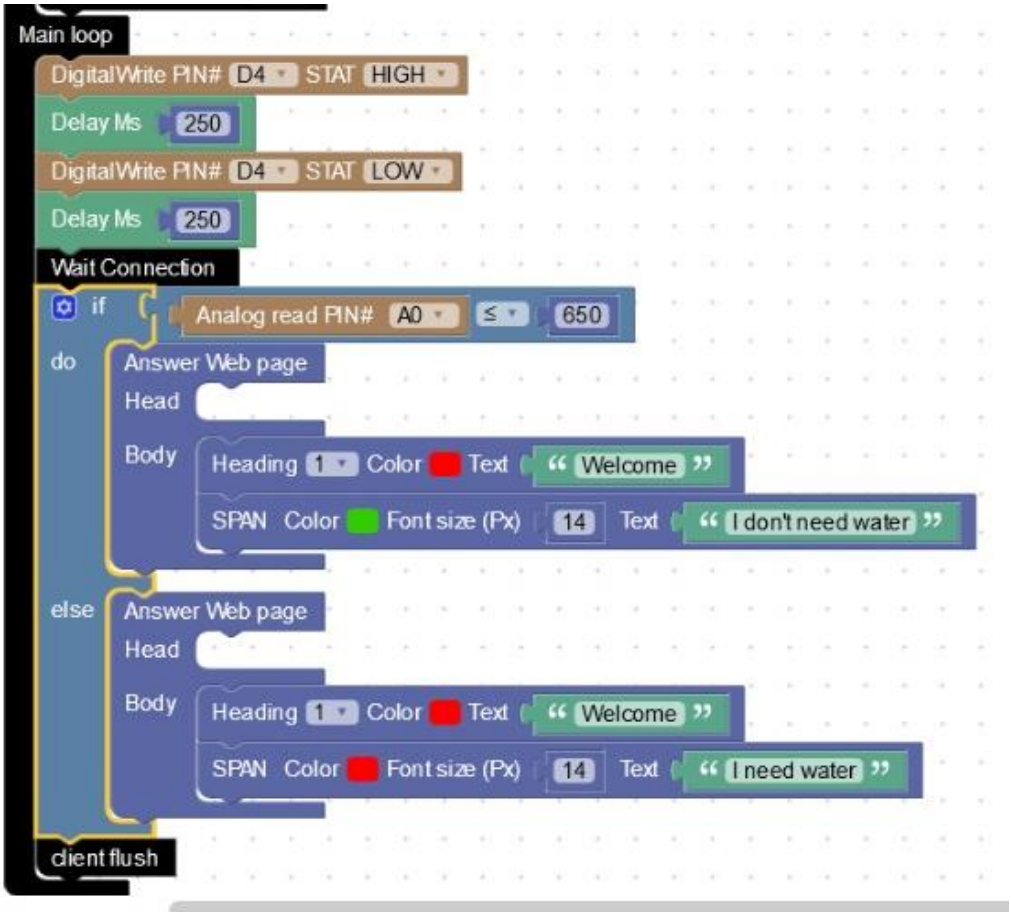


Fig. 25. Block code to the exercise 4.

3. Closing:

Check if the last program works for all participants. Ask the participants to write down the IP address of their NODEMCU board. They should change the network name and password for their home network. If someone does not remember these data, he/she can change the code at home. A TUNIOT code can be saved as a file with the second icon from the right and uploaded with the first icon from the right (Fig. 26). The board should work independently of a computer when you plug the USB cable to a power bank. Ask the participants to connect the embedded system to the their home network and test the calibration few times a day. Are the messages about watering adequate to the reality? Ask the participants to take a picture of the calibrated plant.

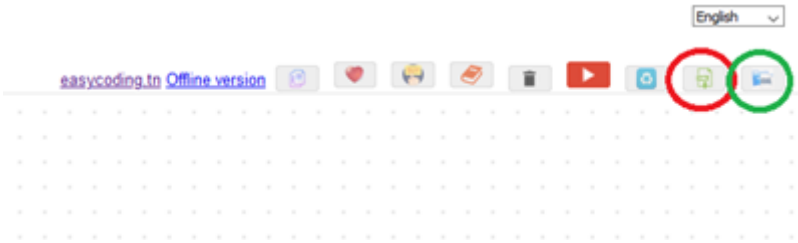


Fig. 26. Save and load a file icons in TUNIOT.

Class 5: Work with mobile applications

Aims:
- familiarising the participants with mobile applications and their usage in the Internet of Things
- preparing an application for checking the soil moisture of a chosen plant
- workshops evaluation.

Materials:
- components per participant: NODEMCU ESP8266 board, female-female cable, USB cable, soil moisture sensor, power bank
- a computer per participant (with Internet access and arduino1.8.1. preinstalled)
- projector
- a glass of water per participant
- optional: mobile phones to test the application.

Class schedule:

1. **Introduction:**

Introductory discussion. Reminding the last class. Ask the participants, if the last tests match with the calibration? Does the code work properly? Did they have any problems with testing? Try to solve problems. Ask the participants to share the photos of their plants. What kind of plants were used to the calibration? How often do they need to be watered? How the kinds of plants are related to the values of calibration?
Discussion about smartphones. Do the participants use Internet in mobile phones? What applications do they use most often? Do they have any concern about smartphones? What are the reasons? What are advantages and disadvantages of using smartphones?

2. **Making mobile applications:**

We will use a free on-line tool from the *https://appinventor.mit.edu/* website. A Google account is needed to log in. You can prepare a few accounts before. Click **Create Apps!**, log in and create a new project (**Start new project**).
The view is divided into a few sections (Fig. 27): **User Interface**, **Viewer**, **Components** and

Properties. All above sections can be found in **Designer** tab. From the **User Interface** you can choose elements of the application and move them to the **Viewer** section. In this section you can see how the application looks like. Some components are visible (e.g. buttons), some are hidden (e.g. a voice message after clicking on a button). In the **Properties** section you can change options of the components.
In the **Blocks** tab you can create a block program in a similar way to the TUNIOT.

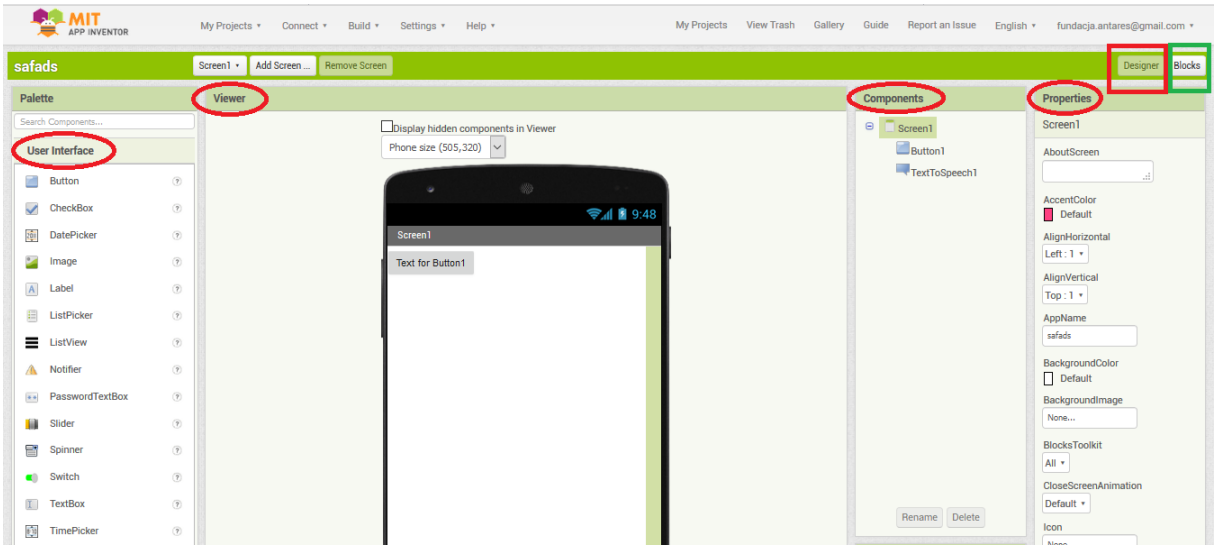


Fig. 27. Interface of the MITapp program.

Exercise 1: Our first application consist of a button which plays a voice message. From the User Interface choose **Button** component and drop it in the **Viewer**. The next component **TextToSpeech** can be found in the **Media**. It is a hidden component so it is not displayed in the Viewer, but under the picture of a smartphone. In the Blocks tab choose **when Button1.Click do** from the **Button1** menu. Stick **call TextToSpeech1.Speak message** block from the **TextToSpeech1** menu to it. The message should be written in a text block (first block in the Text menu) and stuck to the rest of the code (Fig. 28).

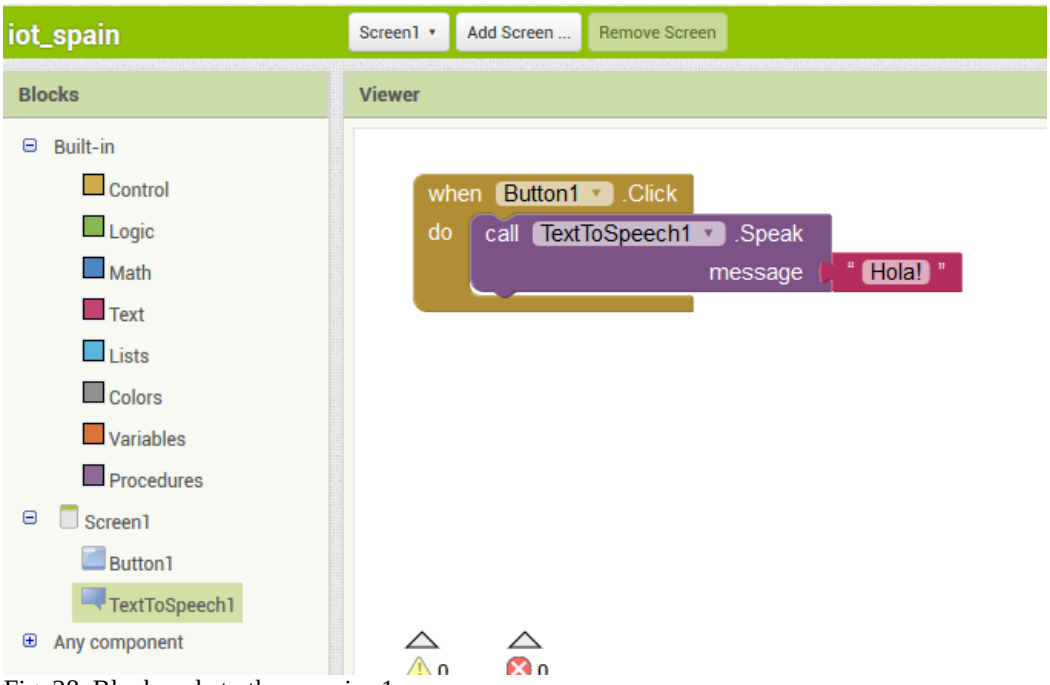


Fig. 28. Block code to the exercise 1.

Ask the participants to install MIT App Inventor 2 application on their smartphones (with Android). Return to the website and select Connect -> Al Companion from the main menu. A OR code should show on the screen. You can scan it with the application or type a 6-digit code. Test your application. Remember that the mobile phone and the computer should be in the same local network.
Exercise 2: The next tasks consist in reading and printing the soil moisture sensor value in the mobile application. In the **Design** part of MITapp select the Button element and change its name to "Download data". You can do it in **Properties -> Text** section. Then, add two more elements: **Label** where the sensor value will be printed and **Web** (from **Connectivity** menu). The last element is hidden. In the **Blocks** part leave the **when Button1.Click do** block only. Add to it set **Web1.Url to** and **call Web1.Get** blocks, both from **Web1** menu. Stick a text block to the first of those blocks and type there the IP address of the NODEMCU board. The next block can be found in **Web1 -> when Web1.GotText do**. Stick the **set Label1.Text to** block to it (from **Label1** menu) and then **get** from **Variables** menu. From **get** block the **responseContent** option can be chosen or drag-and-dropped from the field above, as it is shown on Fig. 29.

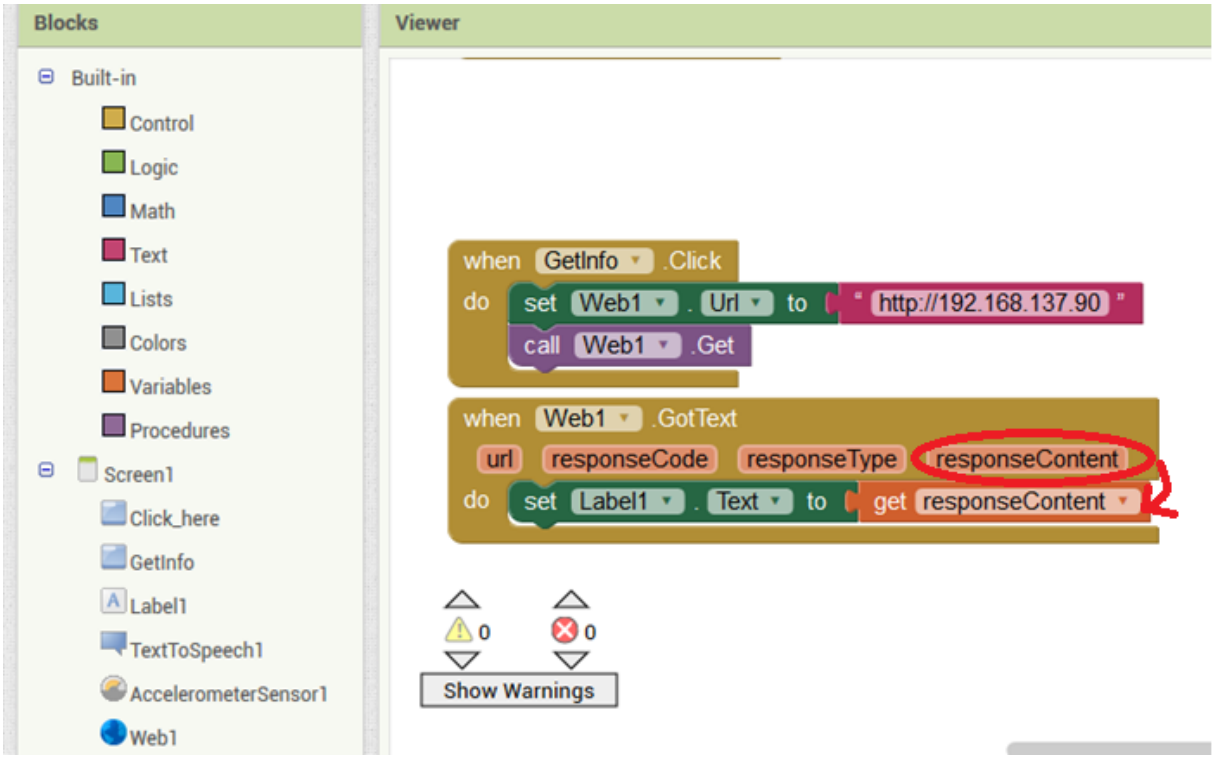


Fig. 29. Block code to the exercise 2.

Exercise 3: Modification of the previous task by adding a timer which allows the program to check the sensor reading automatically. Add **Clock** (from **Sensors** menu) in the **Designer** part. It is a hidden element. In its **Properties** change the **TimerInterval** (in milliseconds) to the desired value. It indicates the time between the sensor readings. In the **Blocks** part **add when Clock1.Timer** (from **Clock1** menu) to the existing code. Then, stick to it the same part of code as in the case of „Download data" button (Fig. 30). Now the application will be automatically downloading the sensor data, as well as by clicking the "Download data" button.

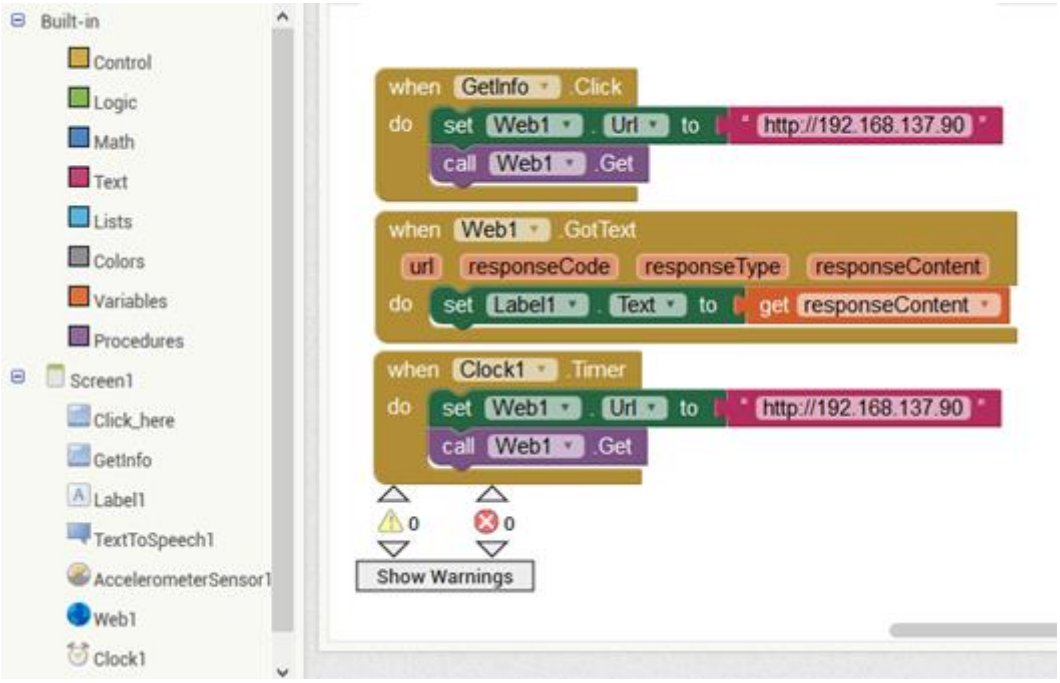


Fig. 30. Block code to the exercise 3.

Exercise 4: Now we will modify the code so that it will print a picture of a sad/happy plant depending on the soil moisture sensor value. The pictures can be downloaded from here:
https://drive.google.com/open?id=1BQt1Tv29RanrdOcfceghKYkNDTDadGy4
https://drive.google.com/open?id=1SLw50cgsDDgkZUj63UhI99PETtSp6OQO
In the **Designer** part tick the **FormatHTML** option in the **Label1**'s **Properties**. Thanks to this, there will be no additional HTML message printed in the application. Add an element **Image** from **User Interface** and change the option **Picture** by selecting downloaded pictures. Once you upload the sad/happy plant pictures, set the **Picture** back to **None**. This way there will be no picture after loading the application (before we download the first data). You can set the **Height** and **Width** of the pictures to 100 pixels. In the **Blocks** part choose **if then else** block from **Control** menu and stick it under the **set Label1.Text to** block. In the first line (after **if**) stick the second block from **Math** menu and change „=" for „>". In the first field set **Label1.Text** block (from **Label1** menu) and the first block from the **Math** menu in the second field. Put your calibration value there, instead of „0". In the line with **then** add the set **Image1.Picture to** block (from **Image1** menu) and a text block. Type there the file name of the sad plant picture (with the extension). Take the same procedure for the line with **else** but for the happy plant picture instead (Fig. 31).

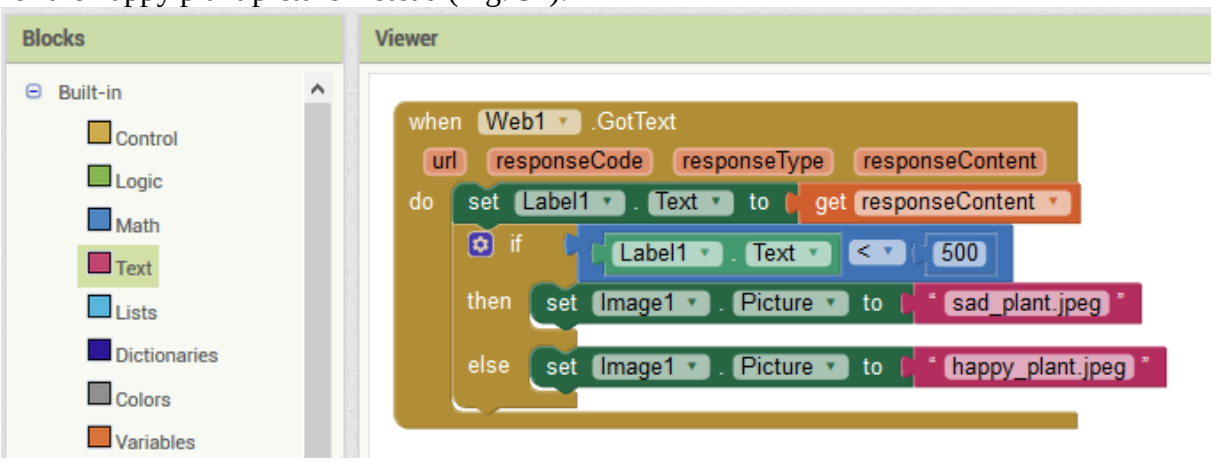


Fig. 31. Block code to the exercise 4.

Exercise 5. If you want to receive an SMS about a need of watering the plant automatically, add another conditional statement. In the **Designer** part add a hidden element **Texting** from **Social** menu and element **Switch** from the **User Interface**. Select option **On** in the **Properties** of the **Switch** element. It means that the switch will be on after starting the application. In the **Blocks** part add **if** block (but without **else** option) under the line with the sad plant picture. Choose the fourth block from the **Logic** menu (with „=”). Stick the **Switch1.On** block (from **Switch1** menu) in the first field, and the **true** block in the second field. In **then** part add the following blocks from **Texting1** menu: **set Texting1.Message to** with a text message about a need of watering, **set Texting1.PhoneNumber** with your telephone number (with a country number) and **call Texting1.SendMessageDirect**. At the end add **set Switch1.On** block from the **Switch1** menu with the **false** block stuck to it (Fig. 32).

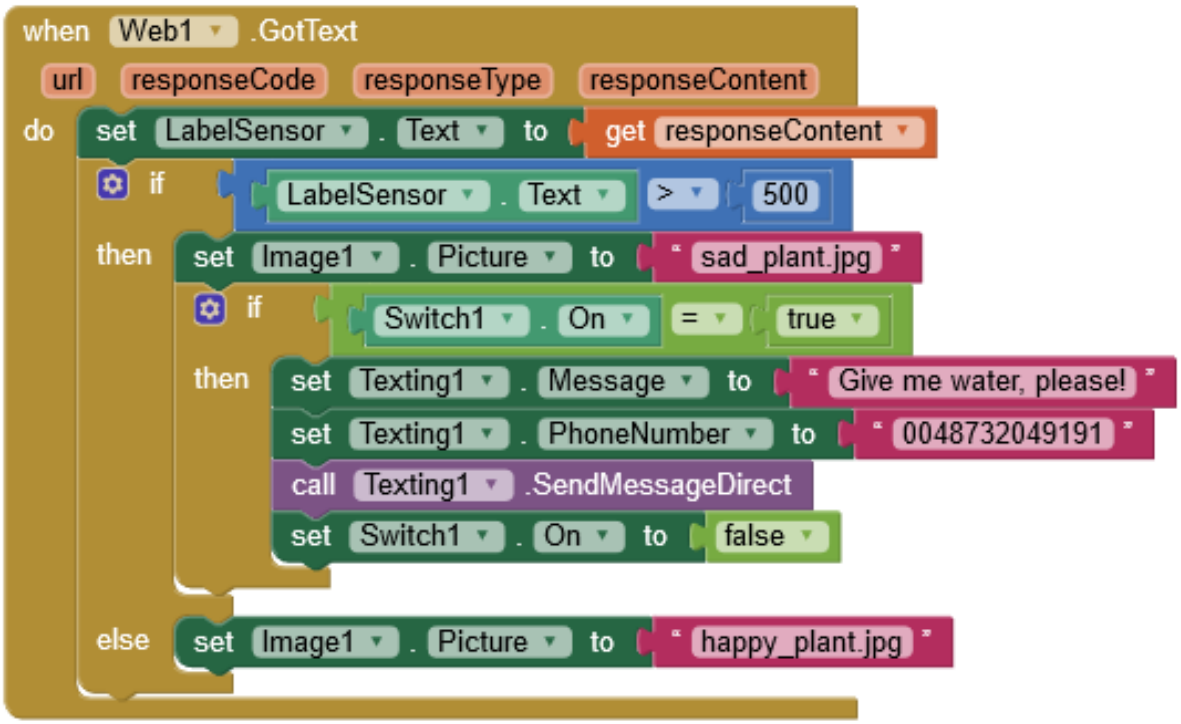


Fig. 32. Block code to the exercise 5.

The conditional statement starts to execute when the sensor value exceeds the given value. In the application you can see the sad plant picture and the program checks if the switch is on. It is defaulted to ON so at the start, the rest of the code will be executed and a SMS will be sent. Then the switch will be set to OFF so you will get no more messages. If you want to get a SMS automatically with the time interval set in the exercise 3, remove the last line from the conditional statement.

3. Closing:

The workshop evaluation. Ask the participants about their impression of the workshop and the idea of the Internet of Things. What did they like the most? What can be improved? Bring out the flipchart which you prepared during the first class. Ask the participants to mark on the flipchart if their expectations are fulfilled with ✓ or x. If they are not sure or partially satisfied, they can mark it with ~. You can prepare some stickers as well. Distribute to the participants the surveys and ask them to fill them up.

Troubleshooting:

- Compilation problem: path to the Arduino folder is too long (move it to the C:)
- Wrong port: in arduino-1.8.1 select Tools -> Port -> COMx (usually the last one)
- Do not plug something else to the other USB port (e.g. a telephone, a mouse)
- Computer should be plugged to the power
- Check components if they are working
- Check USB cable – cheap ones can break quickly
- Check wiring: if the ground PIN is connected
- Check code: did you chose the correct PIN; common mistake – forgetting about delays
- Do not forget to delete all the old blocks in the code
- Serial Monitor should be closed before executing the next program

Annex 1:

barometric pressure sensor

motion sensor

slope sensor

infrared sensor

pressure sensor

humidity sensor

PH sensor

fingerprints reader

fluid flow sensor

sound detector

light and colour detector

temperature sensor

accelerometer

Annex 2:

Day	Hour	Soil moisture

Annex 3:

OBJECT