Управление образования города Пензы

МБОУ лицей №73 г. Пензы

«Лицей информационных систем и технологий»

Секция «Информатика»

**Smart Aeroponics System**

**«Программное обеспечение роботизированной системы выращивания растений с использованием принципов аэропоники »**

Работу выполнили:

Дьячков Даниил Александрович, ученик 10А класса,

Колосовский Николай Сергеевич, ученик 11 А класса

Научный руководитель:

Учитель технологии

Пеганов Станислав Юрьевич

Пенза 2019

Оглавление

[Введение 3](#_Toc533493417)

[Основная часть 4](#_Toc533493418)

[Состав устройства 4](#_Toc533493419)

[Внешний вид: 5](#_Toc533493420)

[Функциональная схема 5](#_Toc533493421)

[Состав Программного обеспечения 6](#_Toc533493422)

[Управление SAS 6](#_Toc533493423)

[Программное обеспечение работы микроконтроллера Arduino Mega 2560 7](#_Toc533493424)

[Программное обеспечение WEB-интерфейса управления и отображения параметров функционирования устройства 7](#_Toc533493425)

[Программное обеспечение Android-устройств интерфейса управления и отображения параметров функционирования устройства. 8](#_Toc533493426)

[Программное обеспечение нейронной сети, определения уровня развития растений 9](#_Toc533493427)

[Нейронная сеть 9](#_Toc533493428)

[Заключение: 10](#_Toc533493429)

[Библиографический список 10](#_Toc533493430)

[Приложение 1](#_Toc533493431)

[Листинг WEB-интерфейса 11](#_Toc533493432)

[Приложение 2](#_Toc533493433)

[Листинг Android-приложения 19](#_Toc533493434)

## Введение

В современном мире очень остро встаёт проблема питания человечества. Дело не в том, что продуктов не хватает, а в том, что существует переизбыток продуктов низкого качества. Употребление таких продуктов в пищу может привести к такой проблеме, как «скрытое голодание».

**Проблема:**

В основном в продуктах тепличного сектора аграрной промышленности не хватает витаминов и микроэлементов. Современные знания в области агротехники позволяют исправить этот недостаток и выращивать продукты с заранее заданным минеральным составом. Но при современных способах выращивания продуктов ЭТО НЕ ВЫГОДНО!!! Это главная проблема, на решение которой направлен наш проект.

Для решения выбранной проблемы была создана роботизированная система выращивания растений, которая работает по принципу аэропоники - Smart Aeroponic System (SAS).

**Цель:**

**Создать устройство, которое возможно внедрить в тепличный сектор аграрной промышленности, тем самым решив проблему продуктов низкого качества.**

**Разработать программное обеспечение для автоматизированного управления, задания параметров функционирования, отображения текущих показателей микроклимата, уровень роста растений.**

Создаваемое устройство должно обеспечить выращивание высоко качественного продукта, с такой же (или более высокой) рентабельностью что и существующие промышленные теплицы.

**Задачи:**

1. Создать программное обеспечение для максимальной автономности функционирования теплицы.
2. Обеспечить глобальное управление устройством SAS через WEB-сайт в сети Интернет.
3. Обеспечить локальное управление SAS с помощью Android-устройств.
4. Внедрить основы искусственного интеллекта для определения уровня развития выращиваемых растений.

Общий принцип работы теплицы заключается в чередовании идеальных условий для ускоренного роста растений и стрессового воздействия для провоцирования синтеза витаминов. Выработка витаминов в растениях подробно описана в научной статье«Содержание витаминов в зависимости от условий» А.А. Нечипоровича, [члена-корреспондента АН СССР](https://ru.wikipedia.org/wiki/%D0%A7%D0%BB%D0%B5%D0%BD%D1%8B-%D0%BA%D0%BE%D1%80%D1%80%D0%B5%D1%81%D0%BF%D0%BE%D0%BD%D0%B4%D0%B5%D0%BD%D1%82%D1%8B_%D0%A0%D0%90%D0%9D_%D0%B7%D0%B0_%D0%B2%D1%81%D1%8E_%D0%B8%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8E_%D1%81%D1%83%D1%89%D0%B5%D1%81%D1%82%D0%B2%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)<http://lsdinfo.org/soderzhanie-vitaminov-v-zavisimosti-ot-uslovij/>

## Основная часть

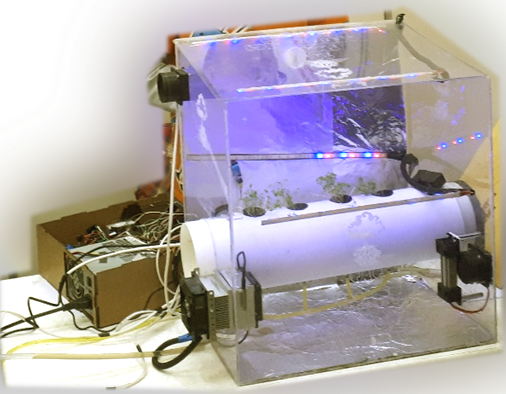
Роботизированная система выращивания растений Smart Aeroponic System (SAS) обеспечивает:

1. Выращивание растений без использования грунта.
2. Автоматический полив растений при уменьшении влажности.
3. Выявление гниения корней растений.
4. Поддержание заданных значений температуры и влажности в теплице.
5. Вентиляция теплицы и корней растений для поддержания комфортных условий.
6. Освещение растений светом оптимальных длин волн.
7. Автоматизированного включения освещения при уменьшении естественного светового потока.
8. Отображение данных датчиков на WeB-сайте.

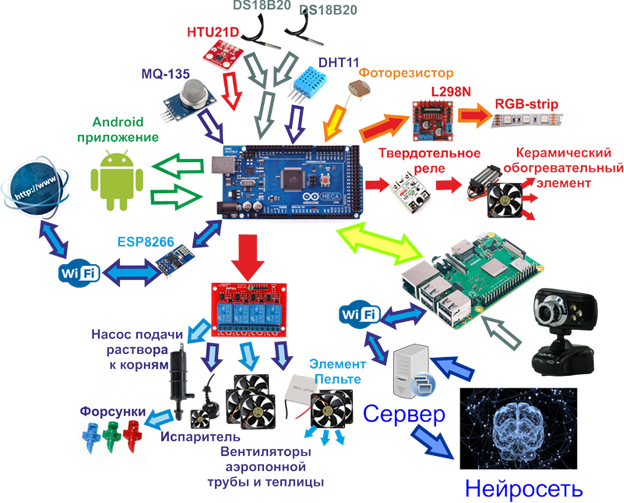
## Состав устройства

1. Плата Arduino Mega 2560.
2. О[дноплатный компьютер](https://ru.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D0%BF%D0%BB%D0%B0%D1%82%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) Raspberry Pi.
3. Модуль Wi-Fi ESP8266.
4. Модуль Bluetooth HC-05.
5. Плата управления электродвигателями L298N.
6. Релейные модули Arduno.
7. Датчик освещенности.
8. Датчик метана MQ-4.
9. Датчик температуры и влажности DHT-11.
10. Датчик температуры и влажности HTU21D.
11. Датчики температуры DS18b20.
12. Твердотельное реле.
13. Блок питания ~220В – 12В.
14. RGBсветодиодная лента (SMD 5050, 30 шт/м).
15. 4 вентилятора.
16. Ультразвуковой испаритель.
17. Насос подачи питательного раствора к корням растений.
18. Элемент Пельтье.
19. Керамический обогревательный элемент 220В.

## Внешний вид:

****

## Функциональная схема

****

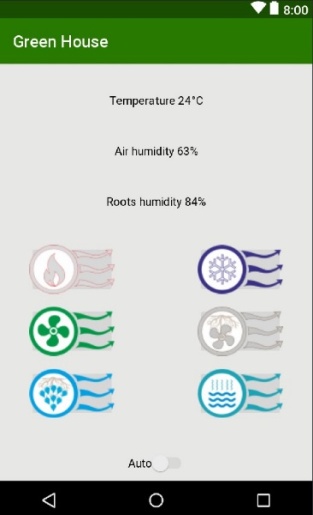
## Состав Программного обеспечения

1. Программное обеспечение работы микроконтроллера Arduino Mega 2560.
2. Программное обеспечение WEB-интерфейса управления и отображения параметров функционирования устройства.
3. Программное обеспечение Android-устройств интерфейса управления и отображения параметров функционирования устройства.
4. Программное обеспечение нейронной сети, определения уровня развития растений.

## Управление SAS

Задание поддерживаемых параметров микроклимата в системе, отображение текущих значений, а также аварийное ручное отключение нагревателей, охладителя и вентиляторов может производится двумя способами:

1. **Глобально**, благодаря использованию Wi-Fi модуля ESP8266-01. SAS имеет собственный WEB-сайт в глобальной сети Интернет с возможностью полного управления устройствами системы и отображения ее параметров.



1. **Локально**, благодаря использованию модуля Bluetooth HC-05. Используя любое Android-устройство и самостоятельно разработанное ПО, также имеется возможность полного управления устройствами системы и отображения ее параметров. При этом оба режима управления полностью синхронизированы между собой.

## Программное обеспечение работы микроконтроллера Arduino Mega 2560

Данное программное обеспечение управляет работой различных исполнительных устройств (нагреватель, насос, вентиляторы, испарители, освещение) через микроконтроллера по показаниям, полученным с датчиков в соответствии заданными параметрами.

Параметры работы устройства задаются как с использованием WEB-интерфейса так и Android-приложения.

Для создания программы использовалась оболочка Arduino IDE и Си - подобный язык программирования.

## Программное обеспечение WEB-интерфейса управления и отображения параметров функционирования устройства

Для обеспечения глобального управления устройством и информирования о текущих параметрах его функционирования и нами был разработан динамический Web-сайт.

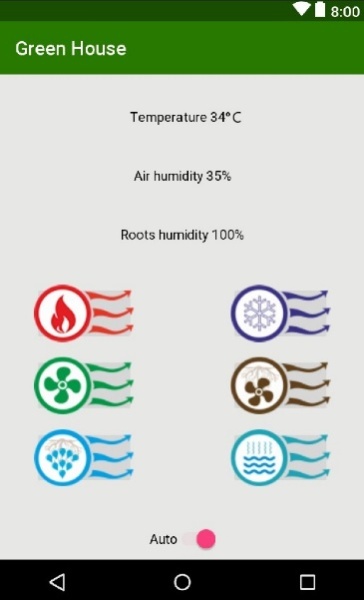
Мы постарались сделать его максимально дружественно-ориентированным, с приятным интерфейсов и максимально информативным. Web-сайт писался самостоятельно с использованием технологий: HTML, CSS, JavaScript, JQuery, AJAX (для обновления данных без обновления самой страницы).

* HTML – отвечает за каркас сайта (создание всех кнопок, надписей)
* CSS – Красивое оформление каркаса сайта,
* JavaScript – отвечает за изменяющиеся элементы на сайте, в том числе и появление/скрытие кнопок, изменения показаний приборов
* AJAX (асинхронный [JavaScript](https://ru.wikipedia.org/wiki/JavaScript) и [XML](https://ru.wikipedia.org/wiki/XML))- суть данной технологии заключается в «фоновом» [обмене данными](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BC%D0%B5%D0%BD_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%BC%D0%B8) между [браузер](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80)ом и [WEB-сервером](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80).В результате, при обновлении данных, сервер возвращает только ту часть документа, на которую пришёл запрос и [веб-страница](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0) не перезагружается полностью, за счет скрипта, который вносит изменения с учётом полученной информации. Это позволяет сделать Web-сайт более удобным для пользователя

На странице Web-сайта, также как и в мобильном приложении, существует переключатель из режима Авто в ручной режим (и обратно). Включив ручной режим пользователь получает возможность полить растения, включить/выключить свет. Перезапуск теплицы доступен в любом режиме

Web-сайт реализован на модуле ESP 8266-01, с самостоятельно написанной прошивкой (в программной среде разработке Arduino IDE). Для передачи данных с ArduinoMega на ESP 8266 используется протокол UART.

## Программное обеспечение Android-устройств интерфейса управления и отображения параметров функционирования устройства.

Для того, чтобы контролировать теплицу находясь в непосредственной близости от неё, было разработано Android приложение на объектно-ориентированном языке Java для локального управления данной системой.

По нашему мнению, это удешевляет и упрощает конструкцию блока управления. Нет необходимости во множестве проводов и цепей, соединяющих плату микроконтроллера, дисплей и органы управления, если бы они были. Помимо этого, сейчас смартфон или планшет есть почти у каждого. Кроме того, по себестоимости это удешевление конструкции на 1,5-2 тысячи рублей.

В автоматическом режиме пользователь может отслеживать температуру воздуха в теплице и температуру и влажность корней.

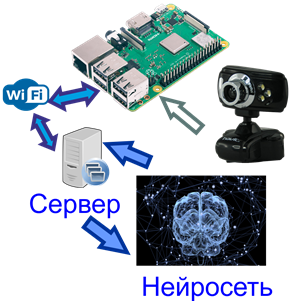
Также пользователь может отслеживать состояние основных систем на дополнительных индикаторах, например, включен ли подогрев или происходит ли вентиляция корней.

При переключении в ручной режим индикаторы начинают выполнять роль кнопок и у пользователя появляется возможность вручную контролировать нагрев, вентиляцию, полив, и другие системы теплицы.

Обмен данными приложения с устройством происходит по сети Bluetooth. Для этого в устройстве имеется модуль Bluetooth HC-05/

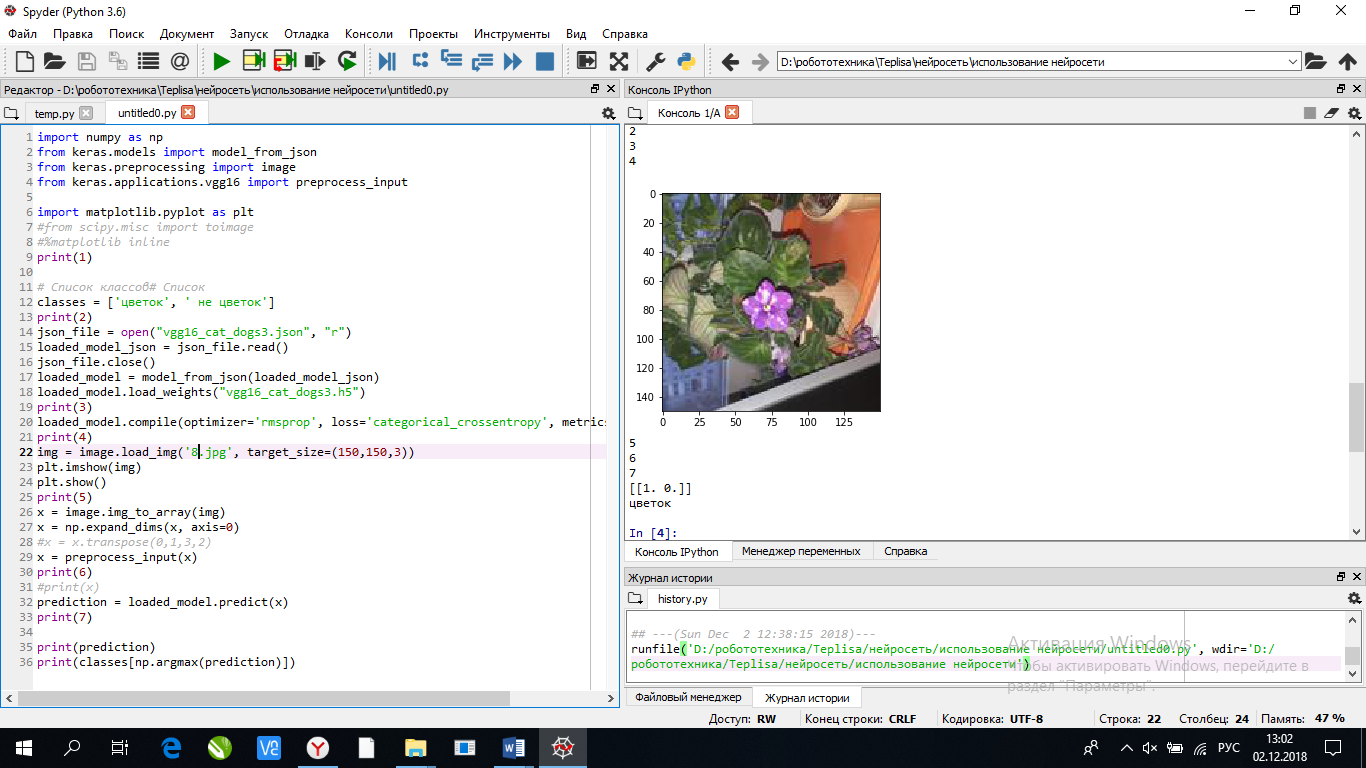
Глобальное (через WEB-интерфейс) и локальное управление (по Bluetooth) полностью синхронизированы между собой, так если, например на WEB-сайте перевести устройство в ручной режим работы на подключенном Android-устройстве высветится индикация ручного режима работы и наоборот.

## Программное обеспечение нейронной сети, определения уровня развития растений

Для диагностики уровня развития растения используется камера, подключенная к RaspberryPi. Для работы с изображениями используется библиотека OpenCV. Чтобы оперативно реагировать на сигналы, в программе, запущенной на Raspberry используется многопоточность – способность программы выполнять разные операции одновременно. В разные потоки разнесены: Отправка данный серверу (или приём данных с сервера) и общение с Arduino.

Самостоятельно написанная программа на языке Python, которая использует данную библиотеку, анализирует количество зелёного, его оттенки на фотографии, и в зависимости от ситуации отправляет те или иные команды на Arduino (по протоколу UART). К примеру, если листья стали желтеть, то отправляется сигнал тревоги, если увеличился объём зелени, то проходит команда на увеличение интервала полива. Микроконтроллер Arduino принимает данные, пришедшие с RaspberryPi, и корректирует программу выращивания растений в зависимости от пришедших данных.

## Нейронная сеть

В теплице могут выращиваться разные растения, и чтобы увеличить независимость теплицы от человека, необходимо диагностировать цветение растений. Благодаря технологии нейронных сетей это стало возможным. Каждый час RaspberryPi отправляет фотографию по Wi-Fi на сервер (для этого используется библиотека socet в Python). На сервере происходит анализ данной фотографии, при помощи написанной и самостоятельно обученной нейросети, на предмет наличия цветов (нейросеть основана на сети VGG16, которую написали работники Оксфордского университета). Для работы с нейросетью используется язык Python и библиотека Keras.

Ответ, после анализа, передаётся обратно на Raspberry. Если ответ пришёл положительный, то Raspberryпередаетсигнал Arduino, откуда информация о цветении выводится на сайт и на мобильное приложение.

## Заключение:

Таким образом, мы создали модель роботизированной системы выращивания растений с использованием принципов аэропоники, которая способна не только быстро выращивать качественные продукты, но и задавать продуктам нужные вкусовые характеристике и витаминный состав.

Для обеспечения работы данного устройства мы использовали самое различное программное обеспечение, которое каждое в отдельности наилучшим и образом способно обеспечить выполнение поставленных задач.

## Библиографический список

<http://lsdinfo.org/soderzhanie-vitaminov-v-zavisimosti-ot-uslovij/>

<https://ithobo.wordpress.com/2017/09/18/raspberry-pi-3-nastrojka/>

<https://docs.python.org/3/library/socket.html>

<https://python-scripts.com/threading>

<http://robocraft.ru/blog/computervision/264.html>

<https://neurohive.io/ru/machine-learning-cases/nejronnaya-set-keras-python/>

## Приложение 1

## Листинг WEB-интерфейса

<!DOCTYPE html>

<html>

<head>

<style type="text/css">

@charset "utf-8";

body {

background-color: rgba(255,157,0,1);

text-align: left;

float: left;

background-repeat: repeat;

background-image: url("https://thumb.cloud.mail.ru/weblink/thumb/xw1/DaAT/GEzwgndBv/fon5.jpg?x-email=rkbcu%40mail.ru");

}

.qw {

width: 327px;

height: 117px;

font-size: 35pt;

text-align: center;

background-color: rgba(27,145,0,0.87);

-webkit-box-shadow: 0 0px;

box-shadow: 0 0px;

border: 19px groove rgba(153,66,0,1.00);

border-radius: 45px;

transition: background .1s linear 1s;

}

.qr {

width: 595px;

height: 83px;

/\* [disabled]margin-top: 12px; \*/

text-align: center;

background-color: rgba(247,207,0,0.82);

border-color: rgba(151,0,102,0.83);

position: relative;

z-index: 86;

padding-top: 19px;

}

.qe {

border: 19px groove rgba(255,150,0,1.00);

background-color: rgba(255,0,7,0.80);

}

.qt{

transition: background .1s linear 1s;

}

.qt:hover{

color: rgba(255,255,7,0.80);

background-color: rgba(19,1,255,0.89);

background: blue;

transition: background .1s linear;

}

#a5 {

}

.d2 {/\*position: relative;\*/}

.f1 {

}

#f2 {

text-align: right;

display: none;

}

#f3 {

text-align: left;

}

.f4{

}

.osn {

text-align: center;

width: 930px;

height: 1525px;}

#d4 {}

#sa1 {

font-size: 62pt;

position: relative;

text-align: center;

display: inline;

font-weight: bolder;

width: 232px;

height: 484px;

color: #FFFFFF;

}

#m2 {

}

#m1 {

}

#m3 {

font-size: 76pt;

font-weight: 700;

display: inline;

position: relative;

text-align: center;

transition: 1s;

z-index: 51;

}

#m4 {

font-size: 77pt;

font-weight: 900;

display: inline;

position: relative;

text-align: center;

transition: 1s;

}

#m5 {

font-size: 45pt;

font-weight: bold;

display: inline;

position: relative;

text-align: center;

transition: 1s;

}

#d4{

}

@-webkit-keyframes de1 {

0% { color: rgba(255,0,0,1.00); }

50% { color: hsla(0,0%,0%,0.00); }

100% { color:rgba(255,0,0,1.00); }

}

@keyframes de1 {

0% { color: rgba(255,0,0,1.00); }

50% { color: hsla(0,0%,0%,0.00); }

100% { color:rgba(255,0,0,1.00); }

}

.de1 {

-webkit-animation: de1 3s linear infinite;

animation: de1 2s linear infinite;

font-weight: bold;

color: rgba(255,246,17,1.00);

}

.de2 {

}

.de {

width: 32px;

background-color: rgba(255,0,4,1.00);

height: 383px;

[disabled]border-collapse: separate;

border: 0px double rgba(255,90,90,1.00);

-webkit-box-shadow: 0 0px 744px 243px rgba(255,0,4,1.00);

box-shadow: 0 0px 744px 243px rgba(255,0,4,1.00);

border-radius: 126px;

z-index: 9;/\*\*/

display: inline-block;

position: relative;

text-align: center;

}

#d1 {

margin-right: 0px;

margin-left: 40px;

display: inline-block;

position: relative;

text-align: center;

}

#d2 {

margin-right: 38px;

display: inline-block;

position: relative;

text-align: center;

}

#de3 {

font-size: 123pt;

z-index: 88;

}

#de4 {

font-size: 92pt;

visibility: visible;

}

#de5 {

font-size: 68pt;

visibility: visible;

}

.m6 {

display: inline-block;

position: relative;

text-align: center;

}

.g1 {

background-color: rgba(255,255,255,0.81);

padding-left: 50px;

padding-right: 50px;

border-radius: 144px;

color: rgba(255,0,0,1.00);

}

#a7 {

height: 112px;

padding-top: 8px;

}

</style>

<meta charset="utf-8">

</head>

<body>

<div class="osn">

<p id="sa1">SMART AEROPONICS SYSTEM</p>

<div>

<div style="white-space: nowrap;">

<div class="de m6" id="d2" ><span id="m1"></span></div>

<div class="d2" style="display: inline-block;"><center>

<h4 class ="qwqr" id="a5" >Температура: <span id="ADCValue">0</span>°C</h4>

<h4 class ="qwqr" id="a6" >Влажность: <span id="ADC1Value">0</span>%</h4>

<h4 class ="qwqr" id="a7">Красный: <span id="ADC4Value">0</span>%<br>Синий: <span id="ADC5Value">0</span>%</h4></center></div>

<div class= "de m6" id="d1" ></div>

</div></div>

<div class="de1 g1" id="m3"><div class="m6">ОПАСНОСТЬ!</div></div>

<divid="m4" class="g1">Риск гниения</div>

<divid="m5" class="g1">Заканчивается раствор</div>

<div class="f1">

<div class="f4" style="white-space: nowrap;">

<div id="f3" style="display: inline-block;">

<div><button type="button" onclick="sendData2(1)" id="a2" class="qw">Авто</button>

<button type="button" onclick="sendData2(0)" style="display:none" id="a4" class="qeqw" >Ручной</button>

</div>

<div><button type="button" onclick="sendData3(1)" id="b2" class="qwqt">reset</button></div>

</div>

<div id="f2" ><div><button type="button" onclick="sendData(1)" class="qw" id="a1">LED OFF</button>

<button type="button" onclick="sendData(0)" style="display:none" id="a3" class="qeqw">LED ON</button></div>

<div><button type="button" onclick="sendData4(1)" class="qwqt" id="b1">полить</button><br></div></div>

</div>

</div><div><span id="ADC3Value">0</span></div>

</div>

<script>

function sendData(led) {

if (led == 0){

document.getElementById("a1").style.display = '';

document.getElementById("a3").style.display = 'none';

}

else{

document.getElementById("a3").style.display = '';

document.getElementById("a1").style.display = 'none';

}

varxhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 &&this.status == 200) {

document.getElementById("LEDState").innerHTML =

this.responseText;

}

};

xhttp.open("GET", "setLED?LEDstate="+led, true);

xhttp.send();

return false;

}

function sendData2(led) {

if (led == 0){

document.getElementById("a2").style.display = '';

document.getElementById("a4").style.display = 'none';

document.getElementById("f2").style.display = 'none';

}

else{

document.getElementById("a4").style.display = '';

document.getElementById("a2").style.display = 'none';

document.getElementById("f2").style.display = 'inline-block';

}

varxhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 &&this.status == 200) {

document.getElementById("LEDState1").innerHTML =

this.responseText;

}

};

xhttp.open("GET", "setLED1?LEDstate1="+led, true);

xhttp.send();

}

function sendData3(led) {

varxhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() { if (this.readyState == 4 &&this.status == 200) {

document.getElementById("LEDState3").innerHTML =

this.responseText;alert("3");

}

};

xhttp.open("GET", "setLED3?LEDstate3="+led, true);

xhttp.send();

return false;

}

function sendData4(led) {

varxhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 &&this.status == 200) {

document.getElementById("LEDState4").innerHTML =

this.responseText;

alert("4");

}

};

xhttp.open("GET", "setLED4?LEDstate4="+led, true);

xhttp.send();

return false;

}

setInterval(function() {

// Вызовфункциисинтервалом 2 секунды Call a function repetatively with 2 Second interval

getData("ADC3");

getData("ADC");

getData("ADC1");

getData("ADC4");

getData("ADC5");

getData("ADC2");

}, 4000); //2000mSeconds update rate

function getData(cc1) {

varxhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function() {

if (this.readyState == 4 &&this.status == 200) {

var a = this.responseText; //ADC3Value

document.getElementById(cc1+"Value").innerHTML = a;

if (cc1 == "ADC3"){

// alert("--- "+cc1+" "+a);

if (((a == "4")||(a == "5"))||((a == "6")||(a == "7"))){

document.getElementById("m3").style.display = 'inline';

document.getElementById("d2").style.display = 'inline-block';

document.getElementById("d1").style.display = 'inline-block';

if (a == "5"){

document.getElementById("m4").style.display = 'inline';

}

if (a == "6"){

document.getElementById("m5").style.display = 'inline';

}

//document.getElementsByClassName("m6").style.display = 'inline-block';

}

if ((a == "3")||(a == 3)){

document.getElementById("m3").style.display ='none';

//document.getElementById("sa1").style.display = 'none';

document.getElementById("d1").style.display ='none';

document.getElementById("d2").style.display ='none';

document.getElementById("m4").style.display ='none';

document.getElementById("m5").style.display = 'none'; //document.getElementsByClassName("m6").style.display = 'none';

}

}

}

};

xhttp.open("GET", "read"+cc1, true);

xhttp.send();

};

</script>

<br><br>

</body>

</html>

## Приложение 2

## Листинг Android-приложения

package com.example.user.greenhouse;  
  
import android.app.Activity;  
import android.bluetooth.BluetoothAdapter;  
import android.bluetooth.BluetoothDevice;  
import android.bluetooth.BluetoothSocket;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.os.Bundle;  
import android.support.v7.app.AppCompatActivity;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.FrameLayout;  
import android.widget.ImageButton;  
import android.widget.ListView;  
import android.widget.TextView;  
import android.widget.Toast;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.OutputStream;  
import java.util.ArrayList;  
import java.util.Set;  
import java.util.Timer;  
import java.util.TimerTask;  
import java.util.UUID;  
import android.widget.Switch;  
import static android.R.layout.\*;  
  
  
public class MainActivity extends AppCompatActivity {  
  
private static final int *REQUEST\_ENABLE\_BT* = 1;  
  
BluetoothAdapter bluetoothAdapter;  
  
ArrayList<String>pairedDeviceArrayList;  
  
ListView listViewPairedDevice;  
FrameLayout mButtonsContainer;  
  
ArrayAdapter<String>pairedDeviceAdapter;  
 private UUID myUUID;  
  
ThreadConnectBTdevice myThreadConnectBTdevice;  
ThreadConnected myThreadConnected;  
  
 private StringBuilder sb = new StringBuilder();  
  
 public TextView textInfo;  
  
 public ImageButton heatBtn;  
 public ImageButton coldBtn;  
 public ImageButton ventBtn;  
 public ImageButton rootVentBtn;  
 public ImageButton rootIrrigationBtn;  
 public ImageButton evaporatorBtn;  
 public TextView temperatureLabel;  
 public TextView humidityAirLabel;  
 public TextView humidityRootsLabel;  
  
 public Switch auto;  
  
 public Integer[] dataToSend = {1,3,3,3,3,3,3};  
  
 public String[] lastIncoming = {"3","3","3","3","3","3"};  
  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
super.onCreate(savedInstanceState);  
setContentView(R.layout.*activity\_main*);  
  
Timer myTimer = new Timer();  
myTimer.schedule(new TimerTask() { // Определяемзадачу  
@Override  
public void run() {  
if(myThreadConnected!=null) {  
myThreadConnected.write(joinArrayOfInts(dataToSend));  
}  
 }  
 }, 0L, 2L \* 1000);  
  
 final String UUID\_STRING\_WELL\_KNOWN\_SPP = "00001101-0000-1000-8000-00805F9B34FB";  
  
textInfo = (TextView)findViewById(R.id.*textInfo*);  
  
heatBtn = (ImageButton) findViewById(R.id.*heatButton*);  
coldBtn = (ImageButton) findViewById(R.id.*coldButton*);  
ventBtn = (ImageButton) findViewById(R.id.*ventButton*);  
rootVentBtn = (ImageButton) findViewById(R.id.*rootVentButton*);  
rootIrrigationBtn = (ImageButton) findViewById(R.id.*rootIrrigationButton*);  
evaporatorBtn = (ImageButton) findViewById(R.id.*evaporatorButton*);  
  
auto = (Switch) findViewById(R.id.*switch1*);  
  
temperatureLabel = (TextView) findViewById(R.id.*temperature*);  
humidityAirLabel = (TextView) findViewById(R.id.*humidityAir*);  
humidityRootsLabel = (TextView) findViewById(R.id.*humidityRoots*);  
  
listViewPairedDevice = (ListView)findViewById(R.id.*pairedlist*);  
  
mButtonsContainer = (FrameLayout) findViewById(R.id.*buttonsContainer*);  
  
 if (!getPackageManager().hasSystemFeature(PackageManager.*FEATURE\_BLUETOOTH*)){  
 Toast.*makeText*(this, "BLUETOOTH NOT support", Toast.*LENGTH\_LONG*).show();  
finish();  
 return;  
}  
  
myUUID = UUID.*fromString*(UUID\_STRING\_WELL\_KNOWN\_SPP);  
  
bluetoothAdapter = BluetoothAdapter.*getDefaultAdapter*();  
  
 if (bluetoothAdapter == null) {  
 Toast.*makeText*(this, "Bluetooth is not supported on this hardware platform", Toast.*LENGTH\_LONG*).show();  
finish();  
 return;  
}  
  
 String stInfo = bluetoothAdapter.getName() + " " + bluetoothAdapter.getAddress();  
textInfo.setText(String.*format*("Этоустройство: %s", stInfo));  
  
} // END onCreate  
  
  
@Override  
protected void onStart() { // Запроснавключение Bluetooth  
super.onStart();  
  
 if (!bluetoothAdapter.isEnabled()) {  
 Intent enableIntent = new Intent(BluetoothAdapter.*ACTION\_REQUEST\_ENABLE*);  
startActivityForResult(enableIntent, *REQUEST\_ENABLE\_BT*);  
}  
  
 setup();  
}  
  
private void setup() { // Созданиеспискасопряжённых Bluetooth-устройств  
  
Set<BluetoothDevice> pairedDevices = bluetoothAdapter.getBondedDevices();  
  
 if (pairedDevices.size() >0) { // Еслиестьсопряжённыеустройства  
  
pairedDeviceArrayList = new ArrayList<>();  
  
 for (BluetoothDevice device : pairedDevices) { // Добавляемсопряжённыеустройства - Имя + MAC-адрес  
pairedDeviceArrayList.add(device.getName() + "\n" + device.getAddress());  
}  
  
pairedDeviceAdapter = new ArrayAdapter<>(this, *simple\_list\_item\_1*, pairedDeviceArrayList);  
listViewPairedDevice.setAdapter(pairedDeviceAdapter);  
  
listViewPairedDevice.setOnItemClickListener(new AdapterView.OnItemClickListener() { // Кликпонужномуустройству  
  
@Override  
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
  
listViewPairedDevice.setVisibility(View.*GONE*); // Послекликаскрываемсписок  
  
String itemValue = (String) listViewPairedDevice.getItemAtPosition(position);  
String MAC = itemValue.substring(itemValue.length() - 17); // Вычленяем MAC-адрес  
  
BluetoothDevice device2 = bluetoothAdapter.getRemoteDevice(MAC);  
  
myThreadConnectBTdevice = new ThreadConnectBTdevice(device2);  
myThreadConnectBTdevice.start(); // Запускаемпотокдляподключения Bluetooth  
}  
 });  
}  
 }  
  
@Override  
protected void onDestroy() { // Закрытиеприложения  
super.onDestroy();  
 if(myThreadConnectBTdevice!=null) myThreadConnectBTdevice.cancel();  
}  
  
@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
if(requestCode == *REQUEST\_ENABLE\_BT*){ // Еслиразрешиливключить Bluetooth, тогда void setup()  
  
if(resultCode == Activity.*RESULT\_OK*) {  
 setup();  
}  
  
else { // Еслинеразрешили, тогдазакрываемприложение  
  
Toast.*makeText*(this, "BlueTooth невключён", Toast.*LENGTH\_SHORT*).show();  
finish();  
}  
 }  
 }  
  
  
private class ThreadConnectBTdevice extends Thread { // Потокдляконнектас Bluetooth  
  
private BluetoothSocket bluetoothSocket = null;  
  
 private ThreadConnectBTdevice(BluetoothDevice device) {  
  
try {  
bluetoothSocket = device.createRfcommSocketToServiceRecord(myUUID);  
}  
  
catch (IOException e) {  
 e.printStackTrace();  
}  
 }  
  
  
@Override  
public void run() { // Коннект  
  
boolean success = false;  
  
 try {  
bluetoothSocket.connect();  
success = true;  
}  
  
catch (IOException e) {  
 e.printStackTrace();  
  
runOnUiThread(new Runnable() {  
  
@Override  
public void run() {  
 Toast.*makeText*(MainActivity.this, "Нетконнекта, проверьте Bluetooth-устройствоскоторымхотитесоединиться!", Toast.*LENGTH\_LONG*).show();  
listViewPairedDevice.setVisibility(View.*VISIBLE*);  
}  
 });  
  
 try {  
bluetoothSocket.close();  
}  
  
catch (IOException e1) {  
  
 e1.printStackTrace();  
}  
 }  
  
if(success) { // Еслизаконнектились, тогдаоткрываемпанельскнопкамиизапускаемпотокприёмаиотправкиданных  
  
runOnUiThread(new Runnable() {  
  
@Override  
public void run() {  
mButtonsContainer.setVisibility(View.*VISIBLE*); // открываемпанельскнопками  
textInfo.setVisibility(View.*INVISIBLE*);  
}  
 });  
  
myThreadConnected = new ThreadConnected(bluetoothSocket);  
myThreadConnected.start(); // запускпотокаприёмаиотправкиданных  
}  
 }  
  
  
public void cancel() {  
  
 Toast.*makeText*(getApplicationContext(), "Close - BluetoothSocket", Toast.*LENGTH\_LONG*).show();  
  
 try {  
bluetoothSocket.close();  
}  
  
catch (IOException e) {  
 e.printStackTrace();  
}  
 }  
  
 } // END ThreadConnectBTdevice:  
  
  
  
private class ThreadConnected extends Thread { // Поток - приёмиотправкаданных  
  
private final InputStream connectedInputStream;  
 private final OutputStream connectedOutputStream;  
  
 private String sbprint;  
  
 public ThreadConnected(BluetoothSocket socket) {  
  
 InputStream in = null;  
OutputStream out = null;  
  
 try {  
 in = socket.getInputStream();  
out = socket.getOutputStream();  
}  
  
catch (IOException e) {  
 e.printStackTrace();  
}  
  
connectedInputStream = in;  
connectedOutputStream = out;  
}  
  
  
@Override  
public void run() { // Приёмданных  
  
while (true) {  
try {  
byte[] buffer = new byte[1];  
 int bytes = connectedInputStream.read(buffer);  
String strIncom = new String(buffer, 0, bytes);  
sb.append(strIncom); // собираемсимволывстроку  
int endOfLineIndex = sb.indexOf("\r\n"); // определяемконецстроки  
  
if (endOfLineIndex >0) {  
  
sbprint = sb.substring(0, endOfLineIndex);  
sb.delete(0, sb.length());  
  
runOnUiThread(new Runnable() { // Выводданных  
  
@Override  
public void run() {  
//////////reading  
 //temperatureLabel.setText(sbprint);  
String[] words = sbprint.split(" ");  
temperatureLabel.setText("Temperature " + words[0] + "°");  
humidityAirLabel.setText("Air humidity " + words[1] + "%");  
humidityRootsLabel.setText("Roots humidity " + words[2] + "%");  
 if (words[7].equals("1")) {  
rootIrrigationBtn.setImageResource(R.drawable.*irrigation\_active*);  
} else if (words[7].equals("-1")) {  
rootIrrigationBtn.setImageResource(R.drawable.*irrigation\_inactive*);  
dataToSend[5] = -1;  
}  
if (dataToSend[0] == 1) {  
if (words[3].equals("1")) {  
heatBtn.setImageResource(R.drawable.*heat\_active*);  
} else if (words[3].equals("-1")) {  
heatBtn.setImageResource(R.drawable.*heat\_inactive*);  
}  
  
if (words[4].equals("1")) {  
coldBtn.setImageResource(R.drawable.*cold\_active*);  
} else if (words[4].equals("-1")) {  
coldBtn.setImageResource(R.drawable.*cold\_inactive*);  
}  
  
if (words[5].equals("1")) {  
ventBtn.setImageResource(R.drawable.*vent\_active*);  
} else if (words[5].equals("-1")) {  
ventBtn.setImageResource(R.drawable.*vent\_inactive*);  
}  
  
if (words[6].equals("1")) {  
rootVentBtn.setImageResource(R.drawable.*vent\_root\_active*);  
} else if (words[6].equals("-1")) {  
rootVentBtn.setImageResource(R.drawable.*vent\_root\_inactive*);  
}  
  
if (words[8].equals("1")) {  
evaporatorBtn.setImageResource(R.drawable.*evaporation\_active*);  
} else if (words[8].equals("-1")) {  
evaporatorBtn.setImageResource(R.drawable.*evaporation\_inactive*);  
}  
lastIncoming[0] = words[3];  
lastIncoming[1] = words[4];  
lastIncoming[2] = words[5];  
lastIncoming[3] = words[6];  
lastIncoming[4] = words[7];  
lastIncoming[5] = words[8];  
//Toast.makeText(MainActivity.this, sbprint, Toast.LENGTH\_LONG).show();  
}  
  
 }  
 });  
}  
 } catch (IOException e) {  
break;  
}  
 }  
 }  
  
  
public void write(String input) {  
byte[] bytes = input.getBytes();  
 try {  
connectedOutputStream.write(bytes);  
} catch (IOException e) {  
// *TODO Auto-generated catch block*e.printStackTrace();  
}  
 }  
  
 }  
  
public String joinArrayOfInts(Integer[] array){  
 StringBuilder s = new StringBuilder("/");  
  
 for(Integer i: array){  
 s.append(Integer.*toString*(i));  
s.append("/");  
}  
  
return s.toString();  
}  
  
/////////////////// Нажатиекнопок /////////////////////  
////////////////////////////////////////////////////////  
public void onClickAutoBtn(View v){  
if(myThreadConnected!=null) {  
/\*  
 if (dataToSend[0] == -1){  
 for(int i = 1; i<7; i++){  
 dataToSend[i] = 3;  
 }  
 }  
 \*/  
dataToSend[0] \*= -1;  
 int c = 1;  
 for (String i: lastIncoming){  
dataToSend[c] = Integer.*parseInt*(i);  
c++;  
}  
 }  
 }  
  
public void onClickHeatBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[1] == 1){  
heatBtn.setImageResource(R.drawable.*heat\_inactive*);  
  
}  
else{  
heatBtn.setImageResource(R.drawable.*heat\_active*);  
  
}  
dataToSend[1] \*= -1;  
  
}  
 }  
  
public void onClickColdBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[2] == 1){  
coldBtn.setImageResource(R.drawable.*cold\_inactive*);  
  
}  
else{  
coldBtn.setImageResource(R.drawable.*cold\_active*);  
  
}  
dataToSend[2] \*= -1;  
}  
 }  
  
public void onClickVentBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[3] == 1){  
ventBtn.setImageResource(R.drawable.*vent\_inactive*);  
  
}  
else{  
ventBtn.setImageResource(R.drawable.*vent\_active*);  
  
}  
dataToSend[3] \*= -1;  
}  
 }  
  
public void onClickRootVentBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[4] == 1){  
rootVentBtn.setImageResource(R.drawable.*vent\_root\_inactive*);  
  
}  
else{  
rootVentBtn.setImageResource(R.drawable.*vent\_root\_active*);  
  
}  
dataToSend[4] \*= -1;  
}  
 }  
  
public void onClickRootIrrigationBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[5] == 1){  
rootIrrigationBtn.setImageResource(R.drawable.*irrigation\_inactive*);  
dataToSend[5] = -1;  
}  
else{  
rootIrrigationBtn.setImageResource(R.drawable.*irrigation\_active*);  
dataToSend[5] = 1;  
}  
//dataToSend[5] \*= -1;  
  
}  
 }  
  
public void onClickEvaporatorBtn(View v) {  
  
if(myThreadConnected!=null &&dataToSend[0] == -1) {  
if(dataToSend[6] == 1){  
evaporatorBtn.setImageResource(R.drawable.*evaporation\_inactive*);  
  
}  
else{  
evaporatorBtn.setImageResource(R.drawable.*evaporation\_active*);  
  
}  
dataToSend[6] \*= -1;  
}  
 }  
  
  
} // END