Система команд программирования Arduino IDE. Словарь терминов и определений

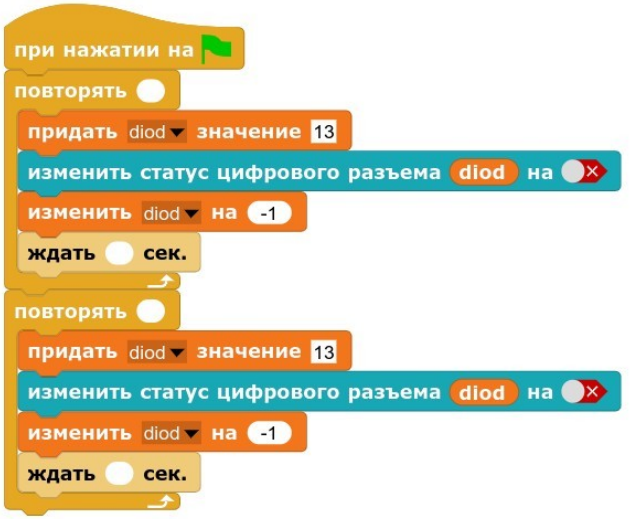
Воронин Игорь Вадимович
@igor_voronin
http://wiki.umki-kit.ru/

Программирование Arduino

- Код для всех плат семейства Arduino пишется на языке Arduino, созданном на базе C++ и фреймворка Wring
- В Arduino IDE можно писать код, используя систему команд.
- Отправлять и получать данные из контроллера; устанавливать сторонние библиотеки.
- Arduino IDE можно скачать с официального сайта https://www.arduino.cc/

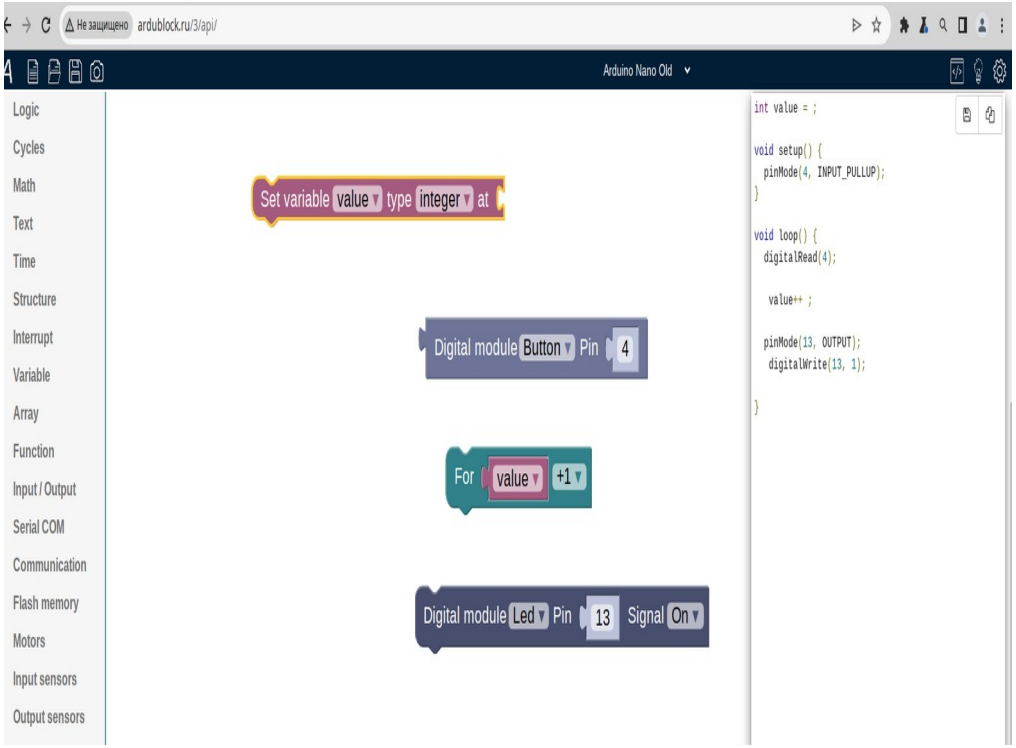
Визуальное программирование

- В прошлом году мы изучали систему команд визуального программирования Snap4arduino
- Узнали что такое циклы, переменные, порты, время ожидания.



Визуальное программирование в Arduino

- Проект ArduBlock, от российских разработчиков, позволяет программировать визуально. И при этом создает исполняемый программный код.
- Программирование осуществляется с помощью блоков.
- Работает в браузере.



Файл ардуино - скетч

- Файл с кодом проекта для Arduino в сообществе принято называть скетчем.
- Скетчи пишутся с использованием системы команд, обеспечивающих корректную работу исполнителя (контроллера).
- Имя файла: ВСЕГДА на английском языке.

Базовая структура скетча

```
void setup() {
}

void loop() {
}
```

- Два обязательных раздела (функции) присутствуют в каждом скетче.
- Раздел setup() - выполняется один раз. В нем устанавливаем значения переменных.
- Раздел loop() - работает в бесконечном цикле. В нем пишем сам код программы.

Базовая структура скетча

```
void setup() {
}

void loop() {
}
```

- Каждый раздел всегда ограничиваем фигурными скобками
  { }
- Сколько скобок открывающих, столько же должно быть закрывающих.
- В одной строке, пишется только одна команда кода.
- Каждая команда оканчивается символом ;
- Комментарии в строке обозначаются символами две косых черты //

Перед setup() можно задать тип переменных

- **int** Описывает целые числа в диапазоне от -32768 до 32767

  int blink_PIN = 13 ;

- **float** Описывает дробные числа (с плавающей точкой)

  float blink_PIN = 33.45;

- Имеет значение, буква ЗАГЛАВНАЯ или строчная

```
// #define blink_PIN 13
int blink_PIN=13;

void setup()
{
  pinMode(blink_PIN, OUTPUT);
}

void loop()
{
  digitalWrite(blink_PIN, 1);
  delay (1000);
  digitalWrite(blink_PIN, 0);
  delay (2000);
}
```

В переменную можно записать адрес порта

- Аналоговый порт имеет адреса:
  A0, A1, A2, A3, A4, A5, A6, A7
  int sensor=A7;
- Цифровой порт имеет адреса:
  D6, D7, D8, D9, D10, D11, D12, D13
- ПРАВИЛО. Цифровой порт записывается без символа D — только число.
  int sensor=13;

Инициализация переменных и портов

- В разделе setup() используем функцию инициализации переменных для портов:
  int sensor1 = A0;
  int sensor2 = A1;

- *pinMode(sensor1, OUTPUT);*
  pinMode(sensor2, INPUT);

- OUTPUT - Выход данных из порта, например когда подключен светодиод.
  INPUT - Прием данных в порт, например когда подключен датчик.

```
int sensorR = A0; // Присоединяем правый датчик к A0
int sensorL = A1; // Присоединяем левый датчик к A1

int motorR  = 2; // 2 контакт определяет направление вращения  первого правого мотора
int motorL  = 4; // 4 контакт определяет направление вращения  первого левого мотора
int mRspeed = 3; // 3 управляет вращением  правого мотора
int mLspeed = 5; // 5 управляет вращением  левого мотора

void setup()
{
  pinMode(mLspeed, OUTPUT); // Настраиваем контакты как работающие на выход
  pinMode(mRspeed, OUTPUT);
  pinMode(motorR,  OUTPUT);
  pinMode(motorL,  OUTPUT);

  pinMode(sensorR , INPUT); // Настраиваем контакты как работающие на вход
  pinMode(sensorL , INPUT);
}
```

Функции чтения-записи

В разделе loop() используем аналоговые и цифровые функции чтения и записи данных в порт:

- Читаем цифровое значение digitalRead(sensorL);
- Записываем цифровое значение digitalWrite(motorR, 1);
- Читаем аналоговое значение analogRead(sensor);
- Записываем аналоговое значение analogWrite(mLspeed, 125);

```
void loop()
{
 if (digitalRead(sensorL)) // ec
 {
  if (digitalRead(sensorR)) // e
  {
   // движение вперед
   digitalWrite(motorR, HIGH );
   analogWrite (mRspeed, Vmax);
   digitalWrite(motorL, HIGH );
   analogWrite (mLspeed, Vmax);
  }
```

Оператор ожидания

- Оператор бездействия всей системы в миллисекундах:
  delay(300);
- Можно задать переменную, например:

  int secunda=2000;
  delay(secunda);

  Здесь период ожидания составить 2 секунды.

```
  delay(300);
}
```

Оператор условия

- if ()

  {}

  Условия сравнения в круглых скобках. Выполняемое действие в фигурных скобках

- if (sensor > 100)
  { digitalWrite(ledPin, HIGH); }

Оператор условия

Сравнение

- Равно ==
- Больше >=
- Меньше <=

```
void loop()
{
  sensor=digitalRead(Knopka_PIN);
  if (sensor == 1)
  {
   digitalWrite(ledPin, LOW);
  }
}
```

Оператор условия:
Сравнение равенства два символа ==
Сравнение больше или равно >=
Сравнение меньше или равно <=

Обратите внимание, что оператор присваивания: это один символ =

Условие: Если - Иначе

- if (условие)
  {действие1}

  else
  {действие2}
- Если переменная статуса stateB равна 1, то включить блинк blink_PIN
- Иначе, выключить блинк blink_PIN

```
void loop()
{
 int stateB = digitalRead(Knopka_PIN);
 if (stateB == 1)
 {
  digitalWrite(blink_PIN, HIGH);
 }
 else
 {
  digitalWrite(blink_PIN, LOW);
 }
```

Монитор порта

- Для вывода данных на экран используем монитор порта
- Serial.begin(9600);
- Serial.print(value);
- ПРАВИЛО: Текст выводим в кавычках, и только латиницей!
- Команда печати с новой строки Serial.println(value);

```
//////////////////////////////////
int Knopka_PIN = 9;
int stateB;

void setup()
{
  pinMode(Knopka_PIN, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
 stateB = digitalRead(Knopka_PIN);
 Serial.print(" stateB = ");
 Serial.println(stateB);
 delay(300);
}
```

Управление моторами UMKI K6

- Осуществляем только через эти цифровые порты:
- D2 — Правый мотор, направление вращения
- D3 — Правый мотор старт/стоп вращения.
- D4 — Левый мотор, направление вращения.
- D5 — Левый мотор, старт/стоп вращения.

Вращение правого мотора

- Соединить с разъемом MR1
- Вперед вращение (по часовой стрелке)
- Стоп вращение
- Назад вращение (против часовой стрелки)
(0904)

```
 int motorR = 2; // направление вращения
 int mRspeed = 3; // скорость вращения
void setup()
{
  pinMode(motorR, OUTPUT);
  pinMode(mRspeed, OUTPUT);
}
void loop()
{
  digitalWrite (motorR, HIGH);
  digitalWrite (mRspeed, HIGH);
  delay(1000); //останов 1 сек.
}
```

Движение робота

- Проезд вперед
- Останов
- Проезд назад
- Останов
- Поворот влево
- Поворот вправо
- Проезд по квадрату (0901)

```
 int mLspeed = 5; // 5-й управляет вращением  левого м
 int motorL = 4; // 4-й контакт определяет направление
 int mRspeed = 3; // 3-й управляет вращением  правого
 int motorR = 2; // 2-й контакт определяет направление

void setup()
{
  pinMode(mLspeed, OUTPUT); // конфигурим контакты как
  pinMode(mRspeed, OUTPUT);
  pinMode(motorR, OUTPUT);
  pinMode(motorL, OUTPUT);
}

void loop()
{
  digitalWrite (motorR, HIGH);// Команда digitalWrite
  digitalWrite (motorL, HIGH);
  digitalWrite (mRspeed, HIGH); //  включаем вращение
  digitalWrite (mLspeed, HIGH);
  delay(1000); // Вращаем 1 сек
  digitalWrite (mRspeed, LOW); // Низкий уровень запре
  digitalWrite (mLspeed, LOW);
  delay(1000); //останов 1 сек.
}
```

Движение с разной скоростью

- ШИМ (PWM)
- Движение отрезками с разной скоростью
- Проезд змейкой
- Прямая езда с подворотами мотором
(1110)

```
 int motorR1 = 2; // 2 контакт определяет напра
 int motorL1 = 4; // 4 контакт определяет напра
 int mL_PWM = 3;  // 3 управляет вращением  лев
 int mR_PWM = 5;  // 5 управляет вращением  пра
void setup() {
  pinMode(mL_PWM, OUTPUT);
  pinMode(mR_PWM, OUTPUT);
  pinMode(motorR1, OUTPUT);
  pinMode(motorL1, OUTPUT);
}
void loop()
{
  digitalWrite (motorR1, HIGH);
  digitalWrite (motorL1, HIGH);
  analogWrite (mR_PWM, 0);  // Значение ШИМ ра
  analogWrite (mL_PWM, 0);
  delay(1000); // действие происходит  1 сек
  analogWrite (mR_PWM, 255);
  analogWrite (mL_PWM, 255);
  delay(1000); //время работы  1 сек.
```

Серво моторы

- Подключение
- Поворот на разные углы
- Изготовление манипулятора

(1310)

```
#include <Servo.h> // подключаем библиотеку для работы
Servo servo1; // объявляем переменную servo типа "servo

void setup() // процедура setup
{
 servo1.attach(A0); //Сервомотор подключен к выходу A0
}

void loop() // процедура loop
{
 servo1.write(0); // ставим угол поворота под 0
 delay(2000); // ждем 2 секунды
 servo1.write(90); // ставим угол поворота под 90
 delay(2000); // ждем 2 секунды
 servo1.write(180); // ставим угол поворота под 180
 delay(2000); // ждем 2 секунды
}
```

Датчик линии

- Подключение
- Цифровой сигнал

(1502)

```
const int sensorR = A1; // Присоединяем правый датчик к A1
const int sensorL = A0; // Присоединяем левый датчик к A0
int Rsensor; // Проверяем правый датчик
int Lsensor; // Проверяем левый датчик

void setup()
{
 pinMode( sensorR , INPUT); // конфигурим  порт на вход
 pinMode( sensorL , INPUT);
 Serial.begin(9600);   // Для отладки. Подключаем монитор порта на скорости общения 9600 байт в сек
}

// Основной бесконечный цикл
void loop()
{
 Rsensor=digitalRead(sensorR); //  считываем значения для правого датчика. 0 - черный; 1 - белый
 Lsensor=digitalRead(sensorL); //  считываем значения для левого датчика. 0 - черный; 1 - белый
 if ((Lsensor == 0)&&(Rsensor==0)) {// если оба датчика видят черный
  Serial.println("Lsensor=0 Rsensor=0");
 }

 if ((Lsensor == 1)&&(Rsensor==0)) {// если  правый датчик видит черный
  Serial.println("Lsensor=1 Rsensor=0");
 }

 if ((Lsensor == 0)&&(Rsensor==1)) {// если  левый датчик видит черный
  Serial.println("Lsensor=0 Rsensor=1");
 }

 if ((Lsensor == 1)&&(Rsensor==1)) {// если  оба датчика видит белый
  Serial.println("Lsensor=1 Rsensor=1");
 }
 delay(100);
}
```

Датчик расстояния

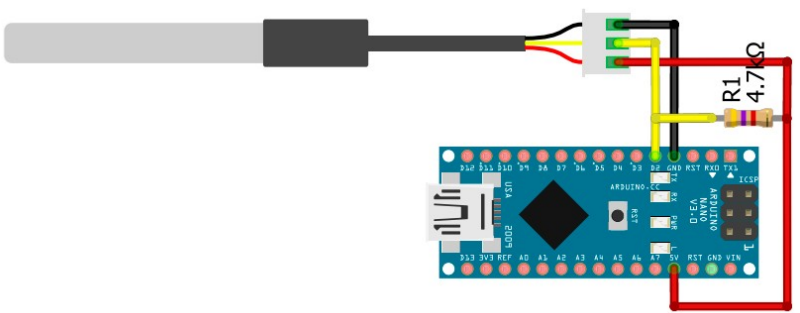
- Подключение
- Прием сигнала (0701)

```
int Trig A3
int Echo A4
int ledPin 13
void setup()
{
  pinMode(Trig, OUTPUT); //инициируем как выход
  pinMode(Echo, INPUT); //инициируем как вход
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600); // инициируем и задаем скорость общения с Монит
}
int impulseTime=0,  distance_sm=0;
void loop()
{
  digitalWrite(Trig, HIGH);
  delayMicroseconds(10); //Подаем импульс на вход trig дальномера рав
  digitalWrite(Trig, LOW); // Отключаем
  impulseTime=pulseIn(Echo, HIGH); // Замеряем длину импульса
  distance_sm=impulseTime/58; // Пересчитываем в сантиметры
  Serial.println(distance_sm); // Выводим сообщение в монитор порта
  if (distance_sm<30) // Если расстояние менее 30 сантиметром
  {
    digitalWrite(ledPin, HIGH); // Светодиод горит
  }
  else
  {
    digitalWrite(ledPin, LOW); // иначе не горит
  }
  delay(100);
 }
```

Датчик температуры воды

- Подключим датчик через нагрузочный резистор 4.7Ком к аналоговому порту A0
- Полученные значения необходимо откалибровать в диапазоне от 0 до 100 град С

  (1802)




```
int PinSensor = A0;
int  VALUE;
void setup() {
  Serial.begin(9600);
  pinMode(PinSensor, INPUT);
}

void loop() {
  VALUE = analogRead(PinSensor);
  Serial.println(VALUE);
  delay(1000);
}
```

Датчик температуры и влажности

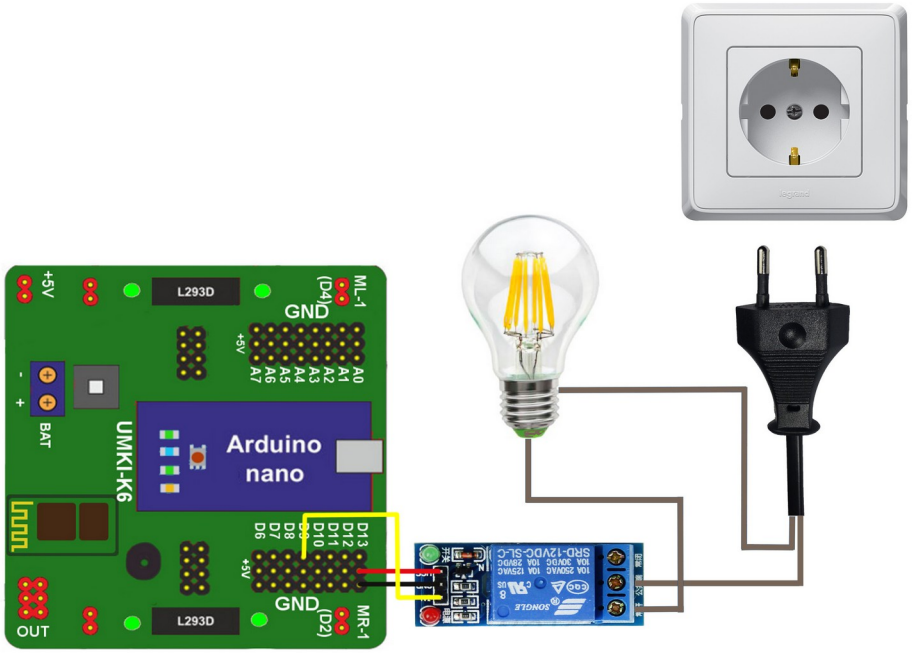
- Подключение датчика на порт D8
- Подключаем библиотеку DHT.h
- Снимаем показания температуры и влажности
- Отображаем их в последовательный порт

(1803)

```
#include "DHT.h"      //подключаем библиотеку для работы
int DHTPIN  = 8 ;
float HUMIDI;
float TEMPE;
DHT dht(DHTPIN, DHT22);    //Инициализация датчика
void setup() {
  Serial.begin(9600);
  dht.begin();
}
void loop() {
  HUMIDI = dht.readHumidity();
  TEMPE = dht.readTemperature();
  Serial.print(HUMIDI);
  Serial.print(" ");
  Serial.println(TEMPE);
  delay(2000);
}
```

Реле

- Техника безопасности
- Подключение
- Вкл света
- Выкл



Реле, программный код

Программируем включения и выключения электрической лампы, через 5 сек/

Наблюдая состояние лампы (ON или OFF) в мониторе порта.

(1805)

```
int Relay = 9;
void setup() {
  // контакт для реле выставляем в режим OUTPUT:
  pinMode(Relay, OUTPUT);
  digitalWrite(Relay, HIGH);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(Relay, LOW);
  Serial.println("ON");  //  "Включение лампы"
  delay(5000);

  digitalWrite(Relay, HIGH);
  Serial.println("OFF");  //  "Выключение лампы"
  delay(5000);
}
```

Условие с двумя переменными

- Используем логическое И для обработки условия с двумя и больше переменными
- if ((val1==0)&&(val2==1))// true, если оба выражения TRUE
- if (x > 0 || y > 0) // true, если любое из выражение TRUE
- if (!x > 0) // true, если только это выражение false

```
  Serial.begin(9600); // Подключаем монит
}

// Основной бесконечный цикл
void loop() {
 Rsensor=digitalRead(sensorR);// считывае
 Lsensor=digitalRead(sensorL);
 if ((Lsensor == 0)&&(Rsensor==0)) {// ес
  Serial.println("Lsensor =0 Rsensor=0");
```

Давайте напишем программу перебора, когда две переменные принимают значения либо 0 либо 1

Подключаем кнопку без резистора

- С внешним резистором, используем INPUT
- Используем встроенный подтягивающий резистор INPUT_PULLUP

  (0502)

```
 int knopkaPIN = 6;
 int blinkPIN = 13;
void setup()
{
  pinMode(knopkaPIN,   INPUT_PULLUP);
  pinMode(blinkPIN, OUTPUT);
}
void loop()
{
  int statusPIN ; // задаем целочисленную переменную
  statusPIN = digitalRead(knopkaPIN);//Статус:1-нажата,0-отпущ
  if (statusPIN == 1)
  {
    digitalWrite(blinkPIN, HIGH); // включаем светодиод
  }
  else
  {
    digitalWrite(blinkPIN, LOW);  // отключаем светодиод
  }
}
```