Версия 1.05 от 13.04.2023

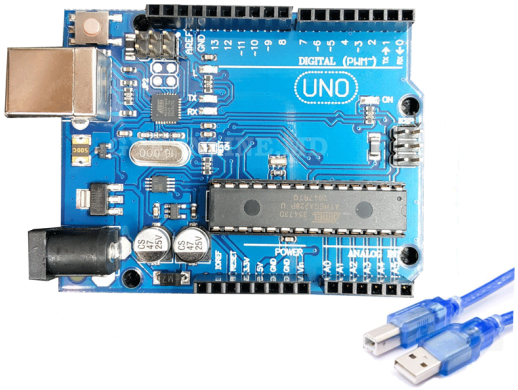
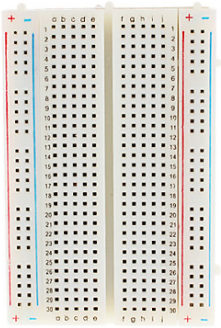
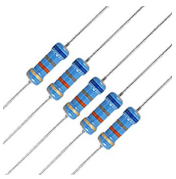
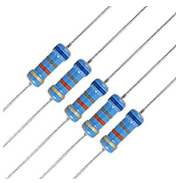
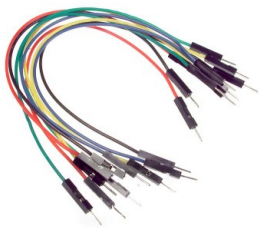
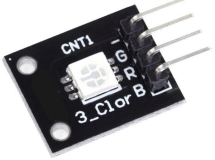
UNO STARTER KIT

ИНСТРУКЦИЯ

НА РУССКОМ

Содержание

1. Состав набора

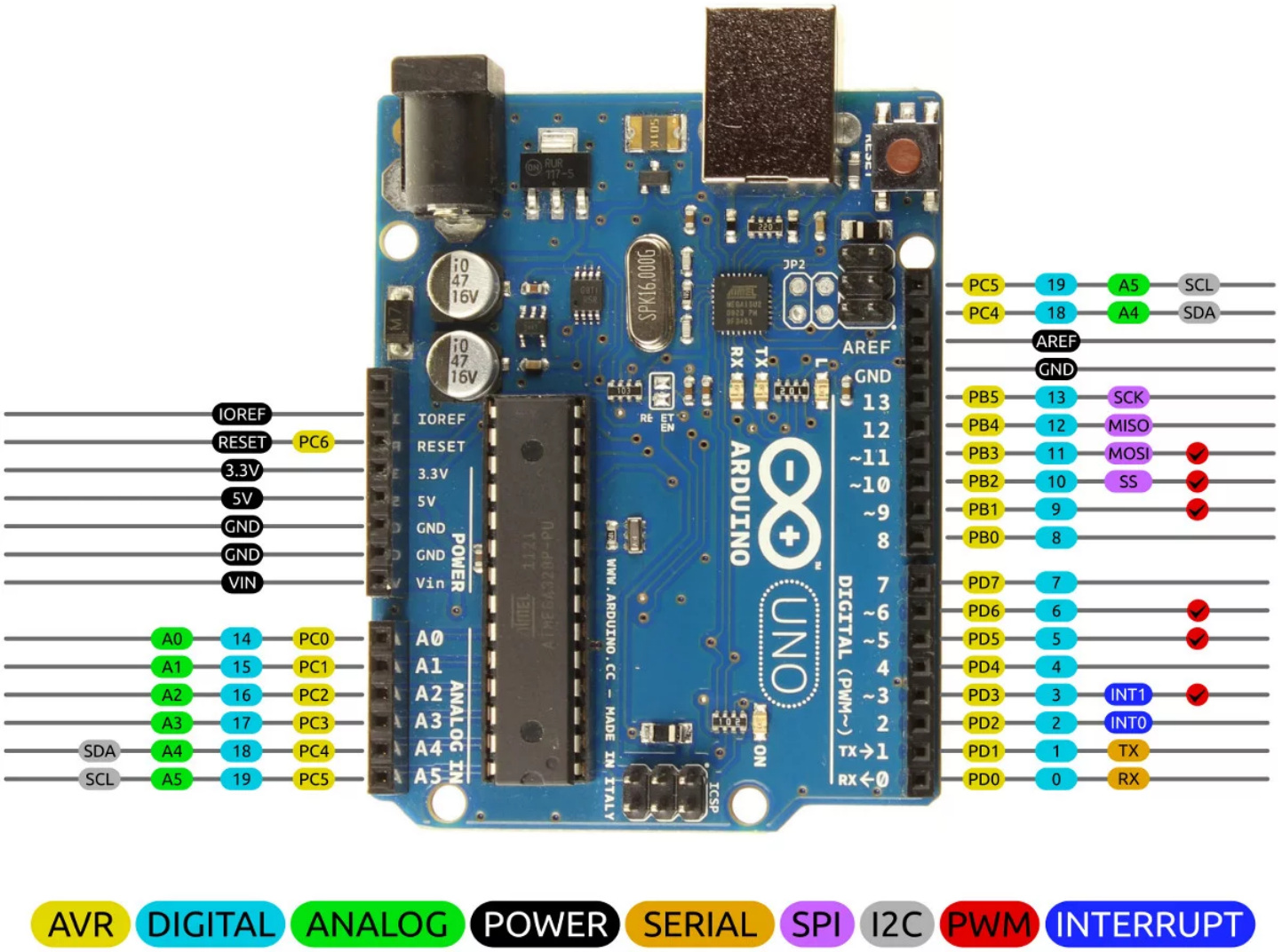
Uno R3 + USB кабель	Плата прототипирования
	
Светодиоды (15 шт)	Фоторезисторы (2 шт)
	
Резисторы 220 Ом (10 шт)	Резисторы 10 кОм (2 шт)
	
Соединительные провода	Бузер (пищалка)
	
Джойстик	RGB-светодиодный модуль
	

2. Основная плата uno

Плата UNO (arduino совместимая плата) – это инструмент для проектирования электронных устройств, плотно взаимодействующий с окружающей физической средой. Это небольшая плата с собственным процессором и памятью, к которой можно подключить разные компоненты: лампочки, датчики, моторы, чайники, роутеры, магнитные дверные замки и вообще всё, что работает от электричества.

Принцип работы таков: **ЕСЛИ ЧТО-ТО СЛУЧИЛОСЬ - НУЖНО ЧТО-ТО СДЕЛАТЬ**. Как сделать, что сделать, на что реагировать - решает сам программист, подключая разные компоненты к плате и закачивая в нее алгоритм работы.

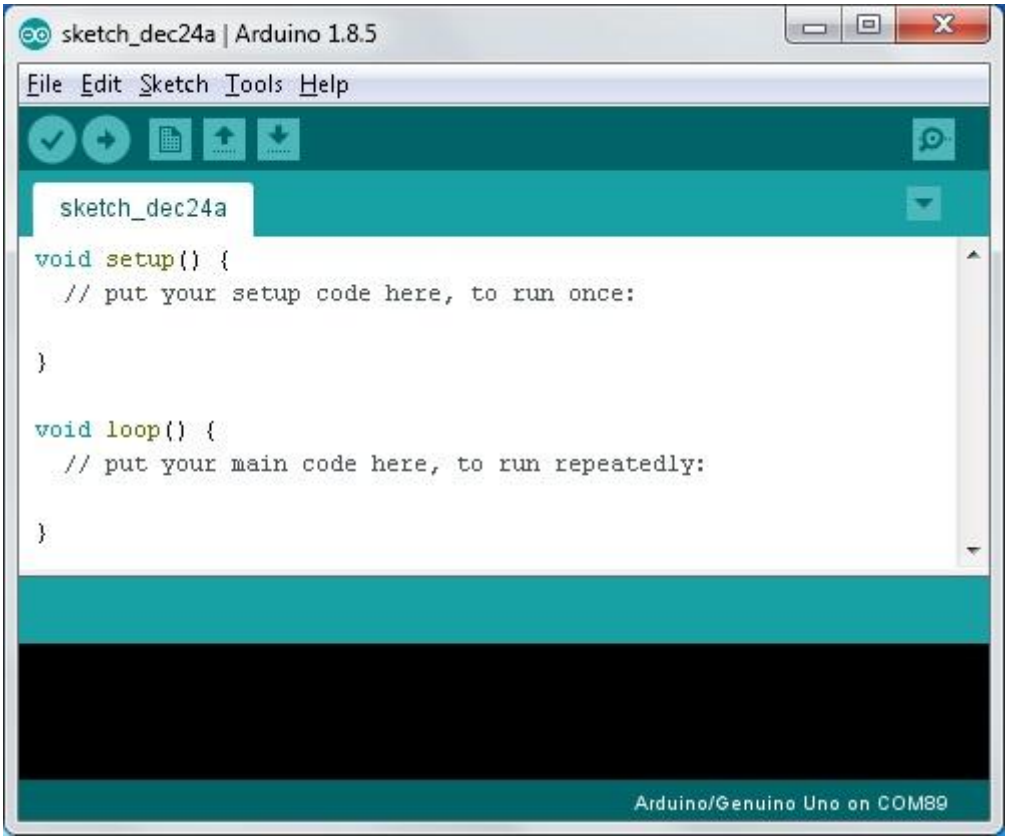
Рис 1. Схема arduino UNO



3. Среда программирования Arduino IDE

Алгоритм работы пишется в среде программирования Arduino IDE. Это программа, которая скачивается с сайта **arduino.cc** (раздел software) и устанавливается на компьютер пользователя. В настройках программы есть выбор языка интерфейса.

Программа, написанная в среде Arduino, называется **скетч**.



Это текстовый редактор: в нем пишется алгоритм работы и закачивается в arduino (Sketch -> Upload). Если в алгоритме работы есть синтаксические ошибки - arduino IDE об этом сообщит в нижней части программы на черном фоне.

4. Первая программа: мигание светодиодом

Встроенный в arduino светодиод.

Итак, среда программирования arduino IDE (далее Программа) скачана, установлена на компьютер и запущена.

A) Подключаем плату arduino к USB-порту компьютера. На плате должен загореться зеленый светодиод **ON**. Оранжевый светодиод **L** будет мигать 1 раз в секунду (встроенный светодиод на 13 пине).

B) В меню (Tools -> Board) выбираем (Arduino/Genuino UNO).

C) В меню (Tools -> Port) выбираем доступный COM-порт.

D) Изменяем скетч:

```
void setup()
{
  pinMode(13, OUTPUT); // определяем пин как выход
}
void loop()
{
  digitalWrite(13, HIGH); // включаем светодиод
  delay(200); // пауза 200 мс
  digitalWrite(13, LOW); // выключаем светодиод
  delay(200); // пауза 200 мс
}
```

E) Чтобы загрузить код в arduino, в меню (Sketch) выбираем (Upload). Или нажимаем CTRL+U. Ждем загрузки кода.

F) Теперь оранжевый светодиод мигает чаще!

Замечание. *Описание команд в коде можно найти в справочнике функций в конце Инструкции. Не поленитесь вводить код вручную - это поможет быстрее его понять!*

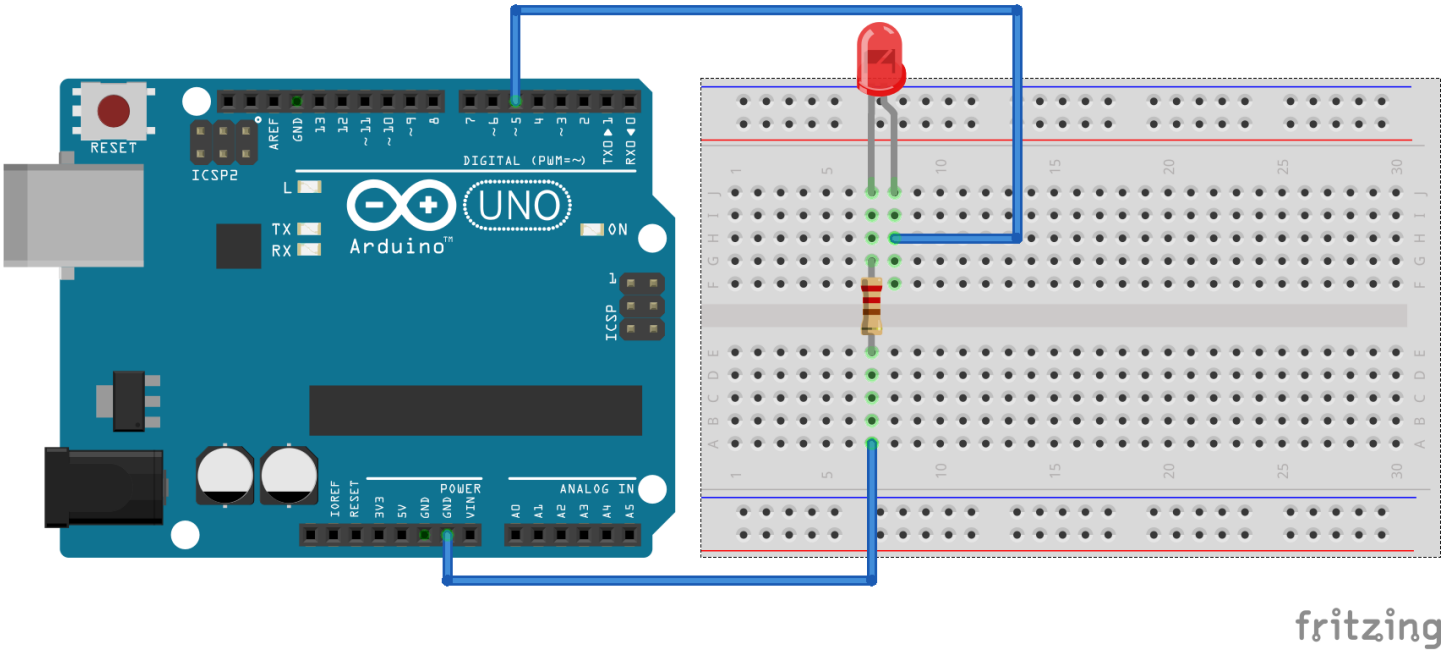
Внешний светодиод.

Светодиод - это маленькая “лампочка”, которая светится при прохождении тока через нее. Ток возникает на участке цепи при подаче напряжения (+5v и GND). Светодиод будет работать только если +5v подключить к более длинной ножке, а GND подключить к более короткой ножке.

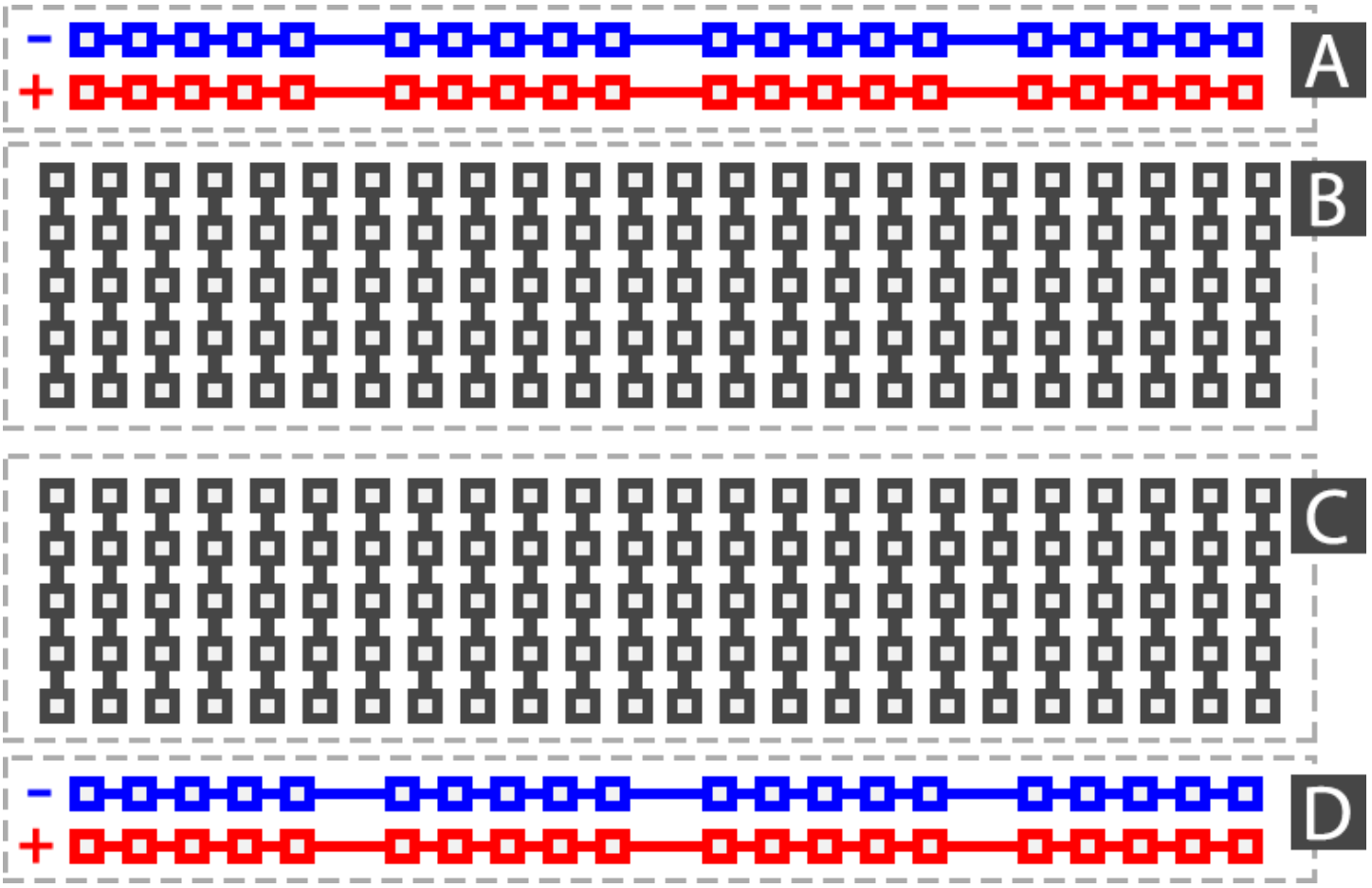
Резистор - это ограничитель тока. Ток, проходящий через светодиод нужно ограничивать резистором (220 Ом), иначе светодиод перегреется и сгорит!

ВНИМАНИЕ! Необходимо отключать плату от источника питания перед тем как соединять или разъединять элементы на плате прототипирования или на arduino! В противном случае плата arduino или элементы могут выйти из строя!

Соберем на плате прототипирования схему:



Принцип соединений в плате прототипирования показан на рисунке:



Закачаем в Arduino код:

```
void setup()
{
  pinMode(5, OUTPUT); // определяем пин как выход
}
void loop()
{
  digitalWrite(5, HIGH); // включаем светодиод
  delay(200); // пауза 200 мс
  digitalWrite(5, LOW); // выключаем светодиод
  delay(200); // пауза 200 мс
}
```

***Замечание**. Обратите внимание, что в коде изменился пин с 13 на 5.*

Описание работы. В коде каждые 200 мс на пин 5 подается напряжение +5v (HIGH). Ток проходит через

светодиод, заставляя его гореть. Следующие 200 мс на пин подается ноль или GND (LOW). Ток больше не проходит через светодиод, и он тухнет. Эта ситуация повторяется постоянно, так как код записан в функции loop (бесконечное повторение).

5. Плавное включение светодиода

Светодиод можно не только включать и выключать, а можно плавно регулировать его яркость свечения.

Оставим предыдущую схему, но изменим программный код:

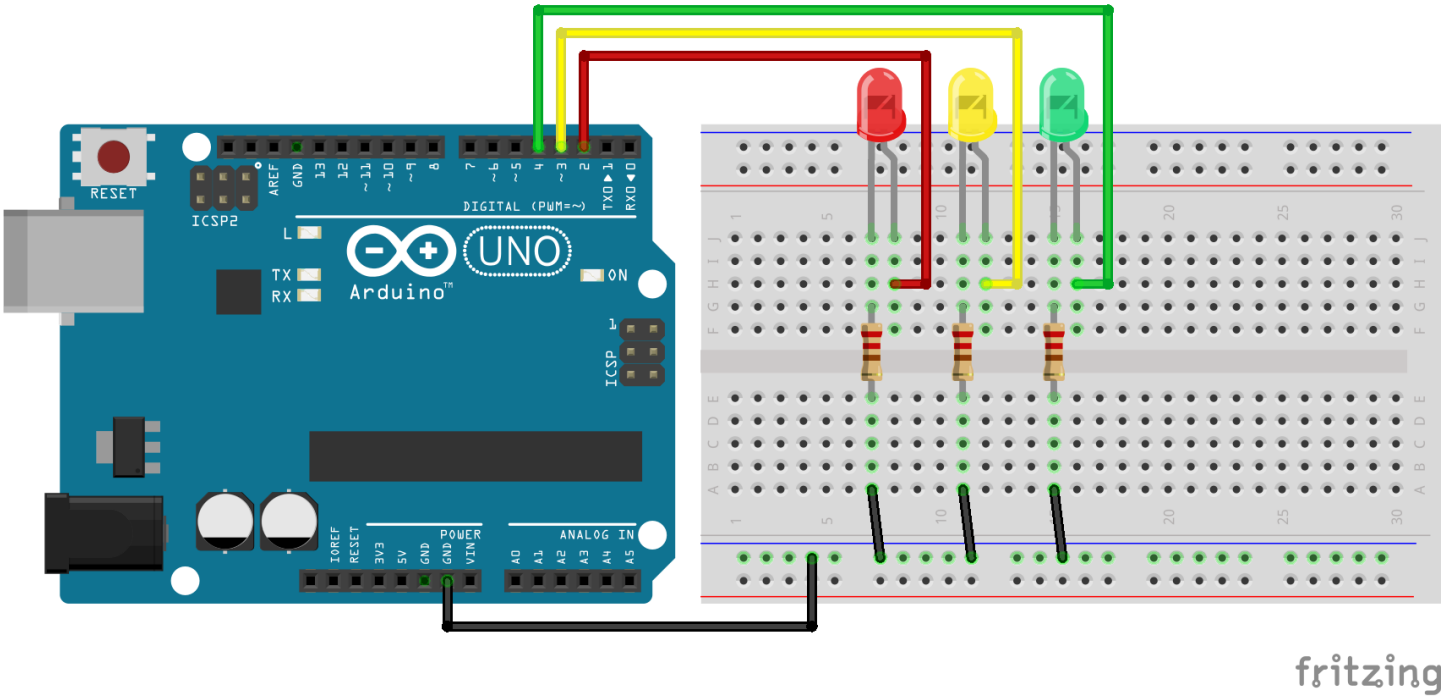
```
void setup()
{
  pinMode(5, OUTPUT); // определяем пин как выход
}
void loop()
{
  // цикл для переменной i от 0 до 255
  for (int i=0; i<=255; i++)
  {
    analogWrite(5, i);
    delay(3); // пауза 3 мс
  }
  // цикл для переменной i от 255 до 0
  for (int i=255; i>=0; i--)
  {
    analogWrite(5, i);
    delay(3); // пауза 3 мс
  }
}
```

Замечание. *Светодиод будет плавно включаться/выключаться на цифровых пинах 3, 5, 6, 9, 10, 11, которые поддерживают сигналы PWM (пины отмечены знаком тильда ~ на плате arduino).*

Описание работы. В первом цикле переменная i изменяет значение от 0 до 255. Это значение записывается в пин 5. Ноль - означает что светодиод не горит, 255 - светодиод горит на полную мощность. Светодиод плавно загорается. Во втором цикле переменная i изменяет значение от 255 до 0, то есть светодиод плавно тухнет. На сколько быстро светодиод загорается/тухнет - зависит от задержки delay.

6. Светофор

Напишем программу, аналогичную работе светофора на улице. Соберем на плате прототипирования схему:



Закачаем в arduino код:

```
// определяем переменные
int red = 2;
int yellow = 3;
```

```
int green = 4;
void setup()
{
  // определяем пины как выходы
  pinMode(red, OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green, OUTPUT);
}
void loop()
{
  digitalWrite(red, HIGH); delay(10000);
  digitalWrite(yellow, HIGH); delay(2000);
  digitalWrite(red, LOW);
  digitalWrite(yellow, LOW);
  digitalWrite(green, HIGH); delay(10000);
  digitalWrite(green, LOW); delay(500);
  digitalWrite(green, HIGH); delay(500);
  digitalWrite(green, LOW); delay(500);
  digitalWrite(green, HIGH); delay(500);
  digitalWrite(green, LOW);
  digitalWrite(yellow, HIGH); delay(2000);
  digitalWrite(yellow, LOW);
}
```

***Замечание**. Если светодиод не горит - проверьте правильность соединения, провода, полярность светодиода (короткая ножка - GND). Также, возможно, светодиод сгорел - замените другим из набора.*

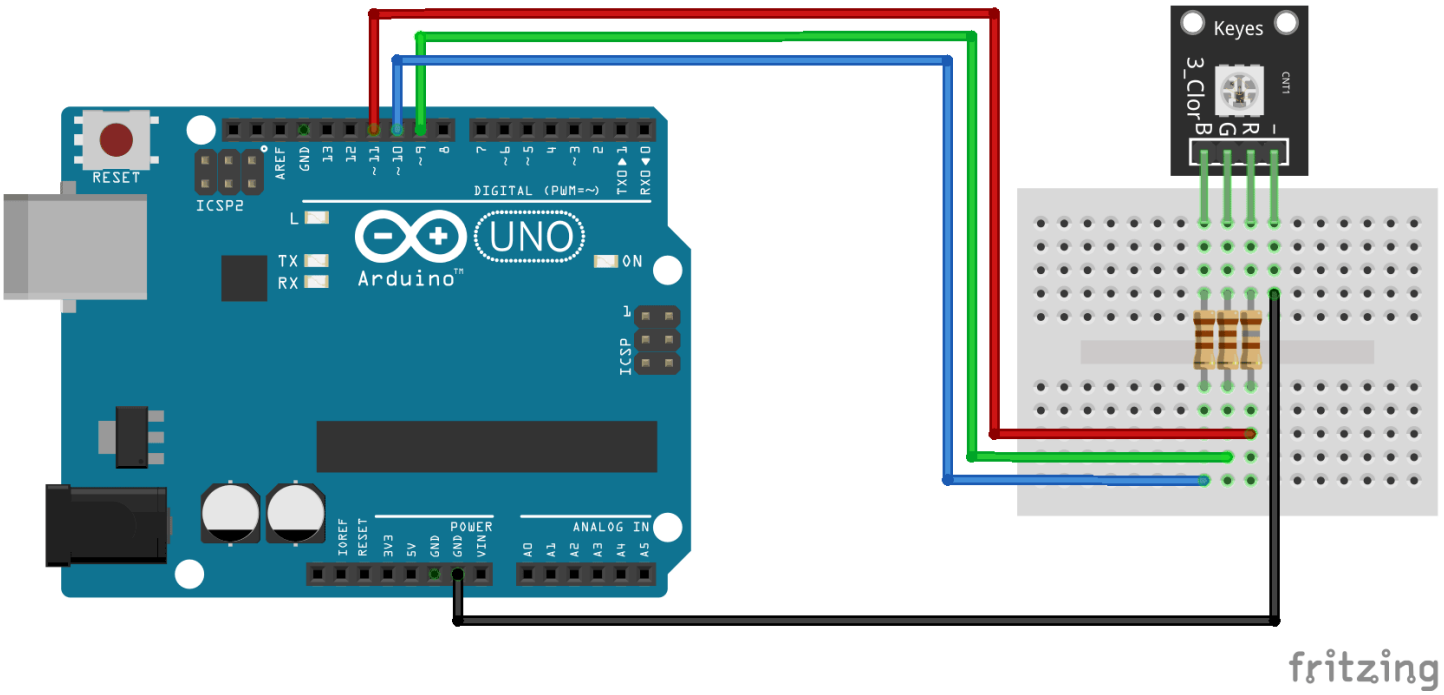
Описание работы. В начале кода объявляем переменные, давая им имена цветов, а значение - номер пина. Использование таких переменных в цикле loop облегчает

чтение кода. Определяем пины как выходы (arduino отдает команды светодиодам, значит, пины - выходы). В функции loop включаем и выключаем нужные светодиоды с заданным интервалом.

7. Трехцветный светодиод (RGB led)

Особенность этого модульного светодиода в возможности смешения цветов. Подключаем резисторы 220 Ом (не обязательно).

Соберем на плате прототипирования схему:



Закачаем в arduino код:

```
// определяем переменные
int red = 9; // G
int blue = 10; // B
int green = 11; // R
void setup()
{
  // определяем пины как выходы
  pinMode(red, OUTPUT);
```

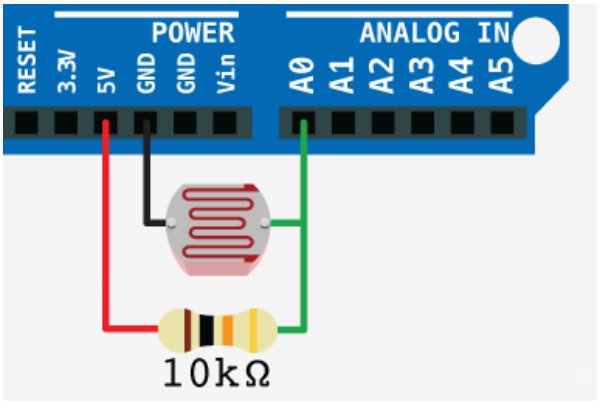
```
  pinMode(blue, OUTPUT);
  pinMode(green, OUTPUT);
}
// функция записи значений в пины
void show(int r, int g, int b)
{
  analogWrite(red, r);
  analogWrite(green, g);
  analogWrite(blue, b);
}
void loop()
{
  show(255, 0, 0); delay(1000); // красный
  show(0, 255, 0); delay(1000); // зеленый
  show(0, 0, 255); delay(1000); // синий
  show(0, 0, 0); delay(1000); // выключить
  show(255, 255, 0); delay(1000); // желтый
  show(255, 0, 255); delay(1000); // фиолетовый
}
```

***Замечание**. На RGB модуле ошибочно указаны буквы цветов. Правильные значения: G - красный, B - синий, R - зеленый.*

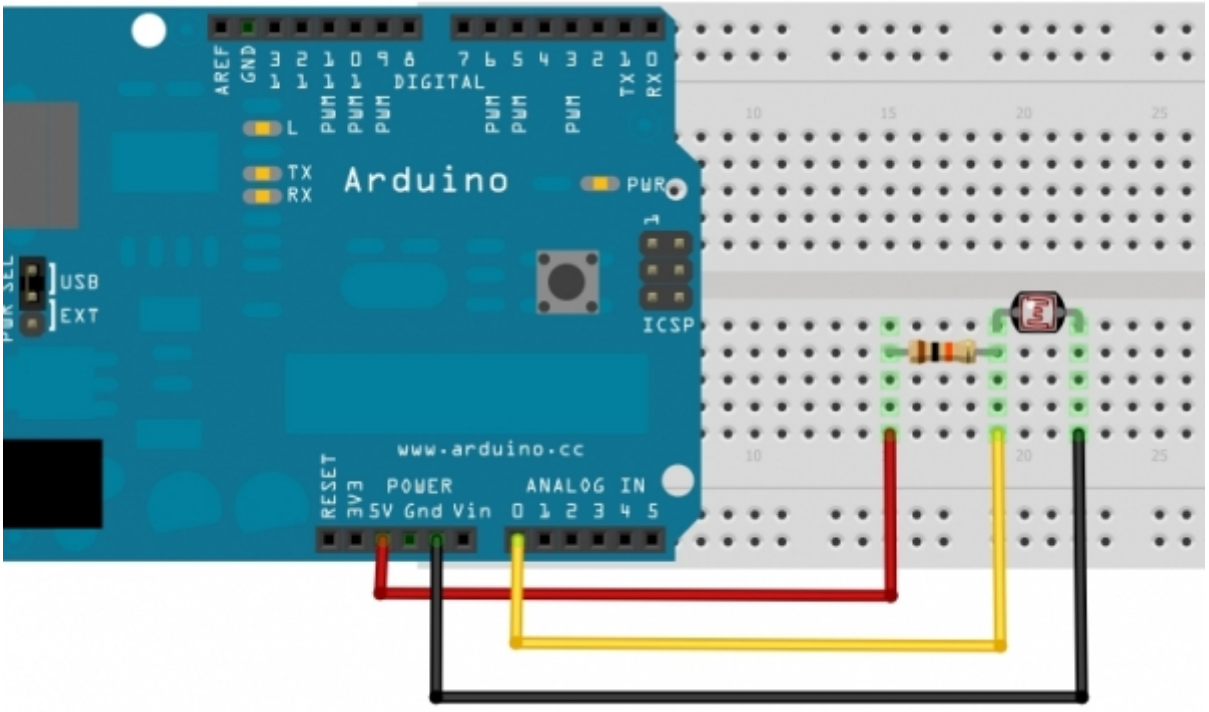
Описание работы. В начале кода объявляем переменные, давая им имена цветов, а значение - номер пина. В функции setup определяем пины как выходы. Создаем свою функцию show, которая будет принимать 3 параметра: значения красного, зеленого и синего цветов от нуля до 255. Смешивая эти 3 цвета в разных пропорциях - получаем новый цвет. Посмотреть палитру цветов можно в программе MSPaint в Windows. В функции loop перебираем цвета с интервалом 1 сек.

8. Фоторезистор (LDR)

Фоторезистор - это резистор, сопротивление которого зависит от количества света, падающего на его поверхность. LDR - light dependent resistor. Подключается к arduino с помощью резистора 10 кОм.



Соберем на плате прототипирования схему:

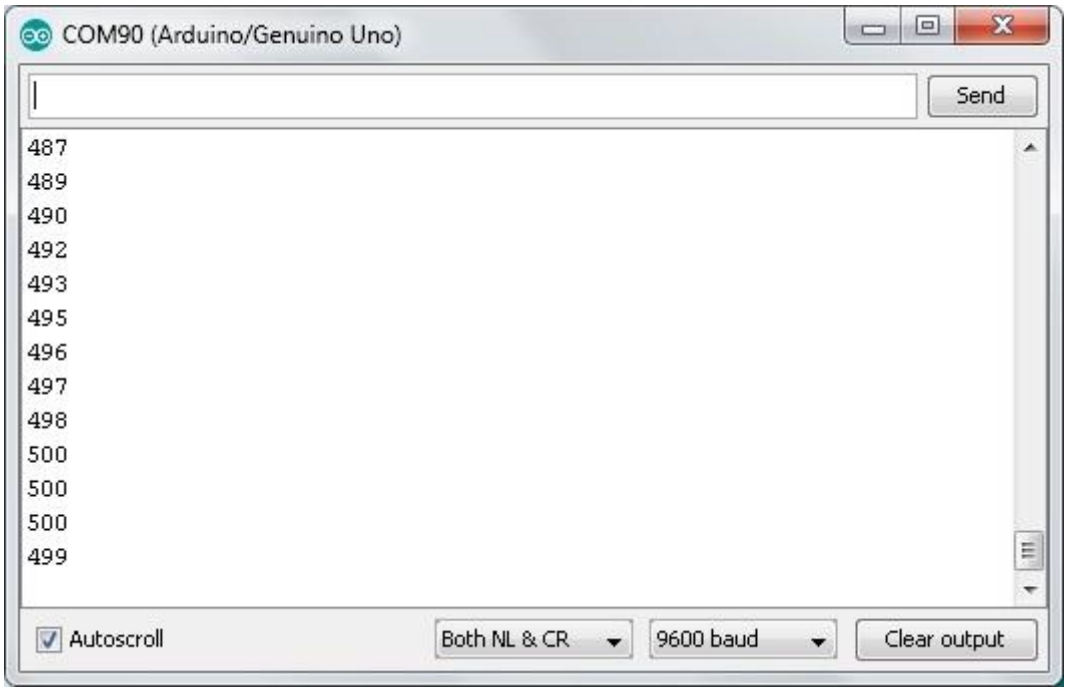


Закачаем в arduino код:

```
void setup()
{
  // запускаем порт на скорости 9600
  Serial.begin(9600);
```

```
}
void loop()
{
  // читаем значение фоторезистора
  int ldr = analogRead(A0);
  // выводим значение в порт
  Serial.println(ldr);
  // пауза 100 мс
  delay(100);
}
```

После закачки кода открываем монитор порта (Tools -> Serial monitor) или нажимаем CTRL+SHIFT+M. В монитор порта будут поступать значения, полученные от фоторезистора.



***Замечание**. Аналоговый пин arduino читает значения от нуля до 1023. После открытия монитора порта проверьте, чтобы скорость порта, указанная в коде (Serial.begin(9600)) совпадала со скоростью в мониторе (9600 baud).*

Описание работы. В функции setup запускаем обмен данными по порту Serial - теперь наша Arduino сможет выводить любые значения в этот порт, а мы сможем их видеть через монитор порта. Порт в данном случае - это интерфейс, по которому arduino подключается к компьютеру (USB кабель, цифровые пины 0 и 1). В функции loop мы создаем переменную ldr и записываем в нее значение, которое читаем из аналогового пина A0 arduino. Каждые 100 мс выводим значение ldr в монитор порта. Если прикрыть фоторезистор рукой - значение переменной ldr будет близко к 1023. Если подсветить фоторезистор фонариком - значение переменной ldr будет близко к нулю.

9. Программа: меньше света - ярче светодиод

Соберем на плате прототипирования схему:

Подключим светодиод как в **разделе 5**, подключим фоторезистор как в **разделе 8**.

Закачаем в arduino код:

```
void setup()
{
  pinMode(3, OUTPUT);
}
void loop()
{
  // читаем значение фоторезистора
  int ldr = analogRead(A0);
  // приводим диапазон значений фоторезистора 0..1023
  // к диапазону яркости светодиода 0..255
  int led = map(ldr, 400, 1023, 0, 255);
```

```
  // меняем яркость светодиода
  analogWrite(3,led);
  delay(5);
}
```

***Замечание**. Обратите внимание, что в команде **map** мы не указываем начальный диапазон 0..1023, а указываем 400..1023. Дело в том, что у меня в помещении при дневном свете фоторезистор выдает значение 490. Поэтому, указав это значение, и прикрывая фоторезистор рукой - я добиваюсь большей разницы свечения светодиода, чем если бы указал диапазон 0..1023.*

Описание работы. В переменную ldr записывается значение фоторезистора. В переменную led с помощью функции **map** пропорционально переносится значение из диапазона 400..1023 в диапазон 0..255. То есть, если в помещении светло (ldr=490) - светодиод горит почти незаметно. По мере уменьшения света (прикрываем фоторезистор рукой) - светодиод горит все ярче.

10. Бузер-пищалка (buzzer)

Бузер - устройство воспроизводящее звук при подаче на него напряжения.

Соберем на плате прототипирования схему:

Arduino uno	Бузер
+5v	+
11	OUT
GND	-

Закачаем в arduino код:

```
int buzzer = 11; // бузер подключен к пину 11
void setup()
{
  pinMode(buzzer, OUTPUT); // определяем пин как выход
}
void loop()
{
  // звук 1
  analogWrite(buzzer,10);
  delay(1000);
  // звук 2
  analogWrite(buzzer,100);
  delay(1000);
}
```

Описание работы. Вначале создается переменная buzzer со значением 11 - это пин, к которому подключен контакт OUT на бузере. В функции loop воспроизводится звук 1 длительностью 1 сек, затем звук 2 длительностью 1 сек.

11. Джойстик

Джойстик - устройство ввода информации, которое представляет собой качающуюся в двух плоскостях вертикальную ручку. Данный модуль джойстика имеет кнопку, которая срабатывает при нажатии сверху на ручку. Плоскости качения ручки (пины VRx и VRy) - это два переменных резистора. Пин SW - это кнопка.

Соберем на плате прототипирования схему:

Arduino uno	Джойстик
GND	GND
+5v	+5v
A0	Vrx
A1	Vry
2	SW

Закачаем в arduino код:

```
int button = 2;//кнопка подключена к цифровому пину 2
void setup()
{
  // определяем пин как вход вход с резистором
  pinMode(button, INPUT_PULLUP);
  Serial.begin(9600); // запускаем монитор порта
}
void loop()
{
  // читаем значение по X из пина A0
  int x = analogRead(A0);
  // читаем значение по Y из пина A1
  int y = analogRead(A1);
  // присваиваем переменной b обратное значение пина кнопки
  int b = !digitalRead(button);
  // выводим в порт все переменные
  Serial.println("x="+String(x)+" y="+String(y)+" bt="+String(b));
  delay(10);
}
```

***Замечание**. Обратите внимание, в функции setup мы не просто определяем пин для кнопки, как вход (INPUT), но и программно добавляем подтягивающий резистор (INPUT_PULLUP). Это связано с тем, что на многих джойстиках отсутствует резистор R5.*

Описание работы. Вначале создается переменная button со значением 2 - это пин, к которому подключен контакт SW на джойстике. В функции loop мы поочередно читаем значения переменных резисторов по оси X и Y, а также кнопки. Измените положение ручки джойстика чтобы увидеть как изменяются значения X и Y. Что интересно, не нажатая кнопка джойстика возвращает значение 1, а нажатая - 0. Для удобства мы присвоим переменной b обратное значение полученного значения - с помощью отрицания НЕ (восклицательный знак).

12. Заключение

Попытайтесь самостоятельно продолжить эксперименты с компонентами в наборе! В зависимости от положения ручки джойстика - менять яркость светодиода, если нажата кнопка - пищать бузером. Подключите много разноцветных светодиодов и в случайном порядке включайте их - получится дискотека!

Если вас заинтересовала тема arduino - заходите на сайт **greenline.md**, где можно найти большое количество разных датчиков, модулей, сервоприводов и платформ для роботов!

На сайте google.com, введя запрос “*arduino название_компонента*”, можно найти способ подключения и уже готовый скетч для большинства электронных модулей.

13. Справочник

Функции	
pinMode(номер пина, режим) Пример: pinMode(2, OUTPUT);	Установить режим работы пина. Режим: INPUT - вход INPUT_PULLUP - вход с резистором OUTPUT - выход
digitalWrite(номер пина, значение) Пример: digitalWrite(2, HIGH);	Подает напряжение на пин или убирает его (записывает значение). Значение: LOW - напряжение на пине 0 вольт HIGH - напряжение на пине +5 вольт
digitalRead(номер пина) Пример: int x = digitalRead(2);	Определяет наличие напряжения на пине (читает значение). Результат: HIGH (или 1) - на пине есть напряжение LOW (или 0) - на пине нет напряжения
analogWrite(номер пина, значение) Пример: analogWrite(3, 64);	Подает напряжение +5 вольт на пин (записывает) с определенным интервалом времени (ШИМ). Возможные **цифровые** пины (на плате отмечены значком тильда (~)): 3, 5, 6, 9, 10, 11 Значение: от 0 до 255 0 - пин выключен полностью 128 - пин включен на 50% времени 255 - пин включен на 100% времени
analogRead(номер пина) Пример: int x = analogRead(A0);	Определяет (читает) величину напряжения на **аналоговом** пине. Результат: число от 0 до 1023, где 0 - это 0 вольт, а 1023 - +5 вольт. Значение 512 = +2.5 вольт.
delay(количество миллисекунд) Пример: delay(1000);	Останавливает выполнение программы на заданное количество миллисекунд (мс). 1 секунда = 1000 мс
millis() Пример: unsigned long x = millis();	Возвращает количество миллисекунд с момента начала выполнения программы на плате Arduino. Это количество сбрасывается в ноль примерно через 50 дней работы платы без отключения питания.

map(значение, fromLow, fromHigh, toLow, toHigh)
Пример:
int val = analogRead(A0);
int x = map(val, 0, 1023, 0 , 255);
analogWrite(4, x);
	Пропорционально переносит Значение из одного диапазона значений (fromLow .. fromHigh) в другой диапазон (toLow .. toHigh).
	Пример: прочитать значение с аналогового пина A0 (подключен переменный резистор) в диапазоне 0..1023 и перевести в значение ШИМ на цифровой пин 4 (подключен светодиод); переменный резистор будет регулировать яркость светодиода

Serial.begin(скорость) Пример: Serial.begin(9600);	Инициирует последовательное соединение (COM-порт) на цифровых пинах 0 и 1 и задает скорость передачи данных в бит/с. Как правило, скорость выбирают 9600.
Serial.print(“текст”) Пример: Serial.print(“Привет”); Serial.print(“Как дела?”);	Отправляет в COM-порт текст. В примере 2 команды выведут в COM-порт текст (все в одну строку, без переносов): “ПриветКак дела?”
Serial.println(“текст”) Пример: Serial.println(“Привет”); Serial.println(“Как дела?”);	Отправляет в COM-порт текст и переносит курсор на следующую строку. В примере 2 команды выведут в COM-порт текст (каждый с новой строки): “Привет Как дела?”

randomSeed(случайное значение)
random(от, до)
Пример:
 	
int randNumber;
void setup()
{
  Serial.begin(9600);
  randomSeed(analogRead(3));
}
void loop()
{
  randNumber = random(10, 20);
  Serial.println(randNumber);
  delay(100);
}
	Функция random возвращает случайное число в диапазоне ОТ и ДО. Число ДО не попадет в результат. Важно! До вызова функции random используйте функцию randomSeed.
	// создаем переменную
	
	// инициализируем COM-порт
	// инициализируем генератор случайных чисел
	// значением на аналоговом пине 3
	
	// генерируем случайное число от 10 до 19
	// выводим в COM-порт это число
	// задержка 100 мс

Операторы	

setup()
Пример:
void setup()
{
  pinMode(buttonPin, INPUT);
}
	Функция setup() вызывается, когда стартует скетч. Используется для инициализации переменных, определения режимов работы выводов и т.д. Функция setup выполняется **только один раз**, после каждой подачи питания или сброса платы Arduino.
	Обязательно присутствует в каждой программе

loop()
Пример:
void loop()
{
  Serial.wrileln(analogRead(A0));
  delay(1000);
}
	Функция loop() вызывается **бесконечное количество раз** во время работы. В ней пишется основной код программы.
	Обязательно присутствует в каждой программе

if (условие)
{
  // этот код выполнится если условие верно
	Проверяет условие на верность. Если условие верно, то выполняется код в фигурных скобках.
	Условие:

}
Пример:
if (x > 3) { delay(1000); }
	x > 10 - переменная x больше 10
	x <= 3 - переменная x меньше или равна 3
	x **==** 7 - переменная x равна 7 (внимание! двойной знак равенства в IF) и др.

if (условие)
{
// этот код выполнится если условие верно
} **else**
{
// этот код выполнится если условие **НЕ**верно
}
Пример: if (x > 3) { delay(1000); } else { delay(3000);}
	Проверяет условие на верность. Если условие верно, то выполняется код в фигурных скобках, но если условие **не**верно - выполняется код в фигурных скобках после оператора else.
	Условие:
	x > 10 - переменная x больше 10
	x <= 3 - переменная x меньше или равна 3
	и др.

Операторы сравнения **==** - равно **!=** - не равно **<** - меньше **>** - больше **<=** - меньше или равно **>=** - больше или равно **Логические операторы** **and** (**&&**) - логическое И **or** (**||**) - логическое ИЛИ	**Арифметические операторы** **=** - переменной присваивается значение **+** - сложение **-** - вычитание ***** - умножение **/** - деление **%** - остаток от деления **Унарные операторы** **++** - увеличение на 1 **--** - уменьшение на 1

for (начальное знач.; условие; изменение знач.)
{
// этот код выполняется столько раз,
// пока условие верно
}
Пример:
int a = 1;
for (i =0; i<4; i++) { a = a + i; }
	Конструкция for используется для повторения участка кода, заключенного в фигурные скобки.
	
	В примере после выполнения цикла переменная “a” будет равна 7. Цикл выполнится 4 раза (от 0 до 3)

Типы данных	
int Пример: int x = 5;	Целое число в диапазоне от -32.768 до 32.767 В примере создаем переменную x, в которую записываем значение 5.
unsigned long Пример: unsigned long y = 10;	Положительное целое число в диапазоне: от 0 до 4.294.967.295 В примере создаем переменную y, в которую записываем значение 10.
boolean Пример: bool z = true;	Логический тип данных (да / нет). В примере создаем переменную z, в которую записываем значение true.

void
Пример:
void setup()
{
// код
}
	Ключевое слово void используется при объявлении функций, если функция **не возвращает** никакого значение при ее вызове