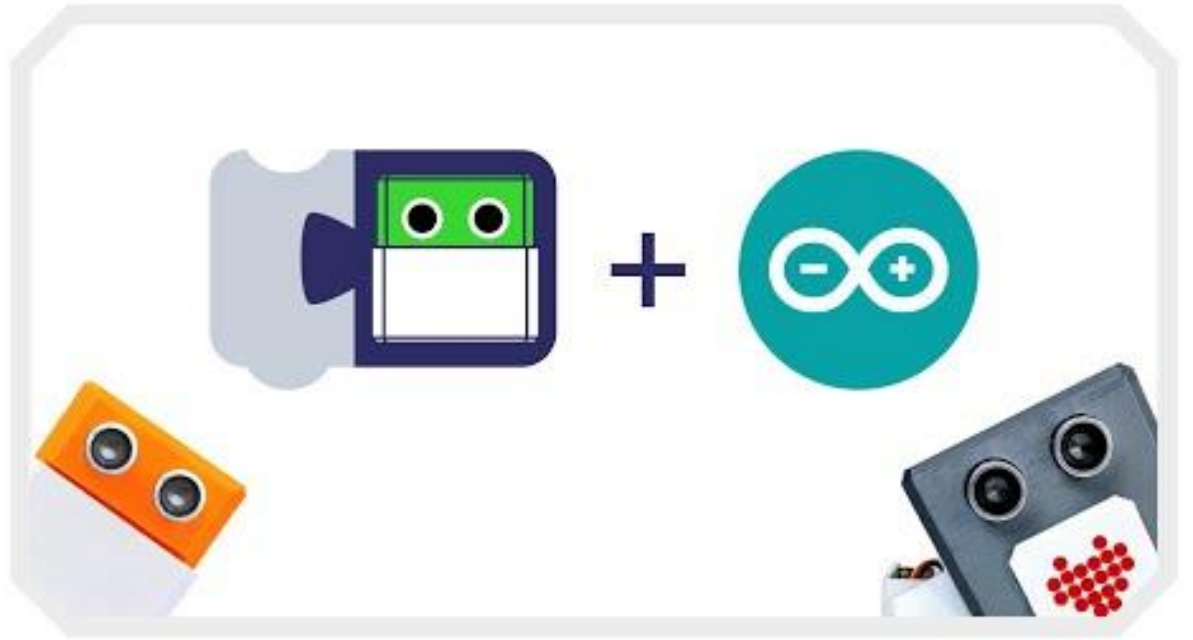


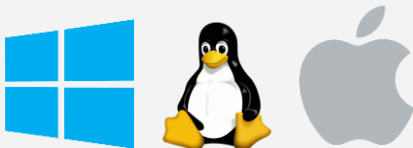
Otto Blockly



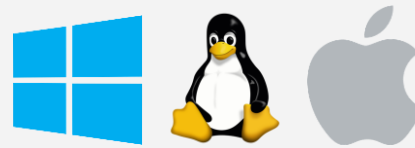
Для загрузки кода, есть **2** способа...



Otto Blockly Standalone: работает в автономном режиме, теперь совместим со всеми ОС, прост в использовании, все в одном кодировании и загрузке через **USB** (нет необходимости в библиотеках **Arduino**)

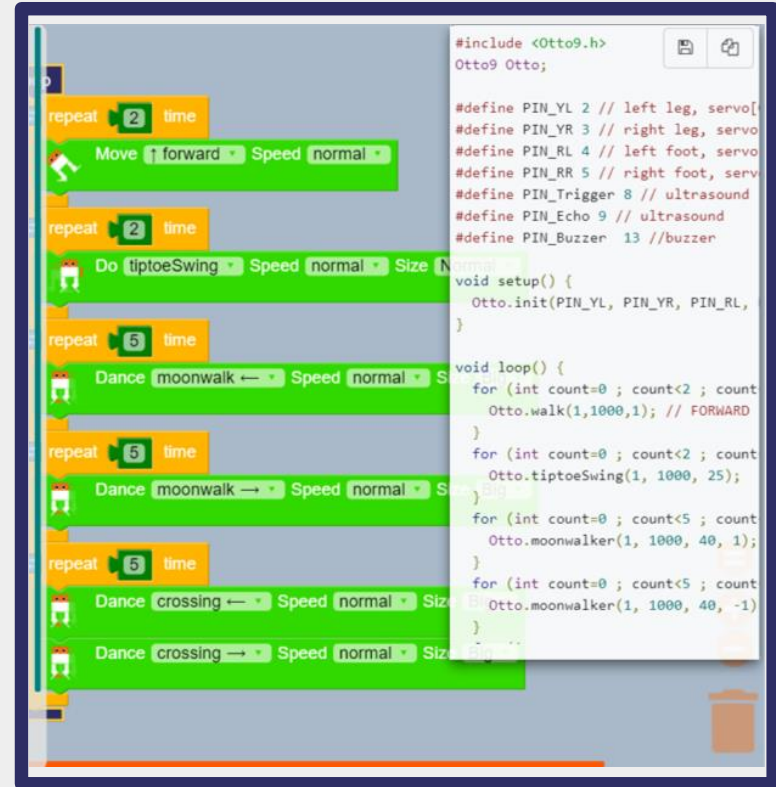


Arduino IDE: Работает в автономном режиме, совместим со всеми ОС, требует дополнительной установки библиотек и опыта программирования на **C++**

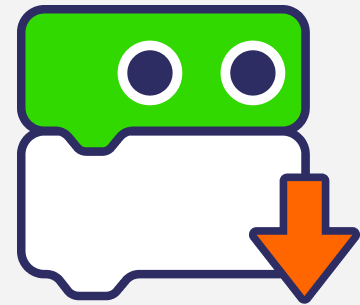


Программируйте своего собственного робота

1. Установка и настройка программного обеспечения.
2. Ознакомление с средой блочного программирования.
3. Первые простые проекты, например, программирование танца с Отто.
4. Разбираемся, как действовать последовательно и применять условия.
5. Подготовка более сложных проектов, включая взаимодействие с датчиками и использование более продвинутых методов программирования.



Загружаем Otto Blockly



ottoblockly

** Ваш антивирус может сказать вам,
что программное обеспечение
небезопасно, но это не так 😊,
просто продолжайте*

Установите

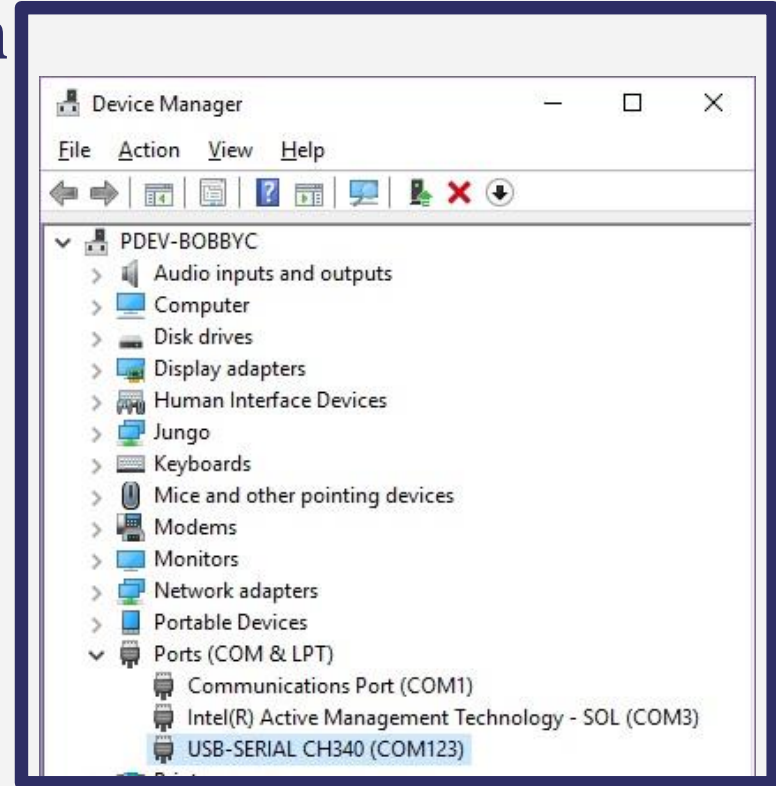


Установка USB-драйвера

Некоторые компьютеры автоматически распознают **Otto** как **USB-устройство** при подключении, но некоторые этого не делают. Если компьютер не распознал **USB-устройство**, необходимо скачать и установить драйвер **USB**:

[USB serial CH340, download it here](#)

[This guide covers all OS step by step](#)



Пользовательский
интерфейс **Otto**
Blockly

Уровни

языки

Примеры

Микроконтроллер

USB port

Загрузка

Монитор

Код

The screenshot shows the Otto Blockly IDE interface. At the top, there are navigation tabs: "Уровни" (Levels) with buttons 1, 2, and 3; "языки" (Languages) with a dropdown set to "English GB"; "Примеры" (Examples) with a search icon; "Микроконтроллер" (Microcontroller) with a dropdown set to "ESP8266"; "USB port" with a dropdown set to "select USB"; "Загрузка" (Upload) with a green arrow button; "Монитор" (Monitor) with a magnifying glass icon; and "Код" (Code) with a code icon. On the left, a "Меню" (Menu) lists various sensors and actuators: Ultrasonic, Line following, Bumpers, Knob, Noise, Temperature, Light, Color, Gesture, Tilt, Joystick, Timing, Arduino, and Control. The main workspace, labeled "Рабочая область" (Working Area), contains a "Блоки" (Blocks) palette with "ultrasonic RGB D1 IO D2", "distance", and "fill ultrasonic color" blocks. A "Setup" block contains "biped leg left D4 right RX foot left D2 right D1 buzzer" and "# 1 ultrasonic trigger 8 echo 9". A "loop" block contains a "then" block with a "gesture confused" block, a "repeat 2 time" block with "move backward speed normal", and another "repeat 4 time" block with "move turn left speed normal" and "move forward speed normal". On the right, the "save arduino code" section shows the corresponding C++ code. Annotations include orange arrows pointing to the "Level" buttons, "Language" dropdown, "Examples" search, "Microcontroller" dropdown, "USB port" dropdown, "Upload" button, "Monitor" icon, and "Code" icon. A red arrow points from the "ultrasonic RGB D1 IO D2" block in the palette to the "ultrasonic trigger 8 echo 9" block in the "Setup" block.

Рабочая область

Блоки

Drag, Drop & Connect

- Тащим
- Кладём
- Соединяем

save arduino code

```
#include <Otto.h>
Otto Otto;

#define LeftLeg 2 // left leg pin, servo[0]
#define RightLeg 3 // right leg pin, servo[1]
#define LeftFoot 4 // left foot pin, servo[2]
#define RightFoot 5 // right foot pin, servo[3]
#define Buzzer 13 //buzzer pin

long ultrasound_distance_simple() {
  long duration, distance;
  digitalWrite(8,LOW);
  delayMicroseconds(2);
  digitalWrite(8, HIGH);
  delayMicroseconds(10);
  digitalWrite(8, LOW);
  duration = pulseIn(9, HIGH);
  distance = duration/58;
  return distance;
}

void setup() {
  Otto.init(LeftLeg, RightLeg, LeftFoot, RightFoot,
  Otto.home());

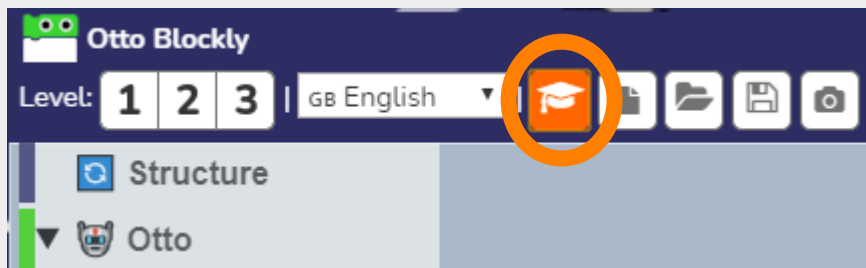
  pinMode(8, OUTPUT);
  pinMode(9, INPUT);
}

void loop() {
  if (ultrasound_distance_simple() < 15) {
    Otto.playGesture(OttoConfused);
    for (int count=0 ; count<2 ; count++) {
      Otto.walk(1,1000,-1); // BACKWARD
    }
    for (int count=0 ; count<4 ; count++) {
      Otto.turn(1 1000 1); // LEFT
    }
  }
}
```


Тест быстрого старта



Hi Otto Builder
Я вернулся, готов помочь вам.



Examples

👤 Level 1 | 🤖 Servo centering | Otto DIY Starter

👤 Level 1 | 🔊 Buzzer | Otto DIY Starter

👤 Level 1 | 🎵 Melody | Otto DIY Starter

👤 Level 1 | 🚶 Walk | Otto DIY Starter

👤 Level 1 | 🦿 Legs calibration | Otto DIY Starter

👤 Level 1 | 🕺 Dance | Otto DIY Starter

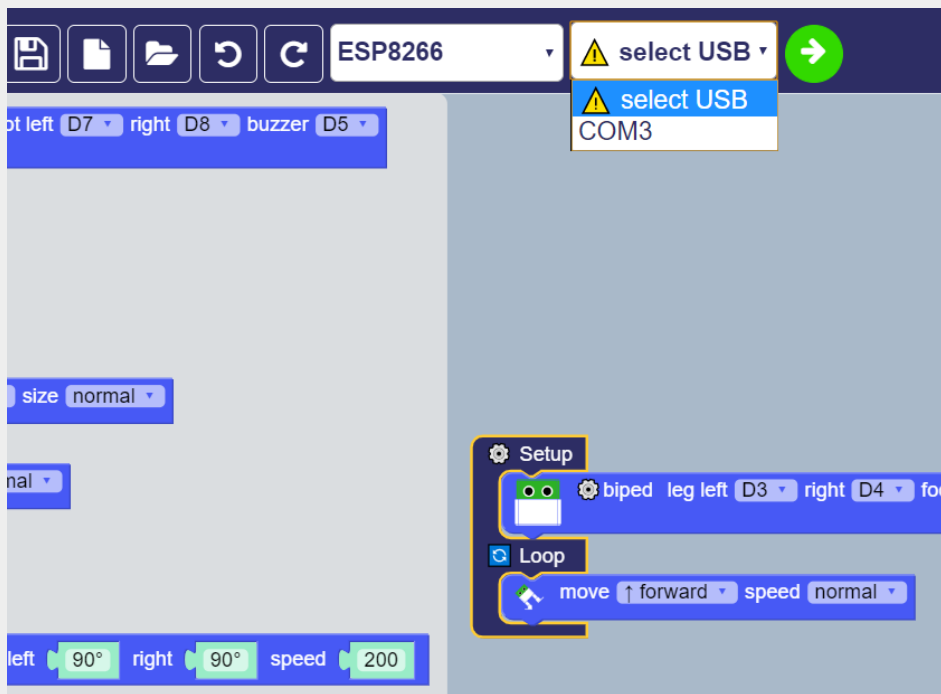
👤 Level 1 | 🦾 Avoid obstacles | Otto DIY Starter

1 Давайте откроем пример, нажав кнопку шляпы.

2 Выберите пример «Viped dance».



Если вы любите приключения, выберите другой пример



3 Подключите робота и убедитесь, что в выпадающем меню **USB** - порта выбран **Com port**

* Если ничего не появляется, значит, USB-драйвер не был правильно установлен в вашем компьютере, необходимо перезагрузить компьютер и повторить попытку.



4 Нажмите кнопку загрузки.



Процесс загрузки может занять около минуты.

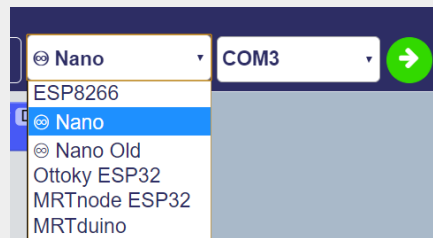


```
Error: Command failed: arduino-cli.exe upload --port COM7 --fqbn
arduino:avr:nano sketch/sketch.ino avrdude: stk500_rcv(): programmer is
not responding avrdude: stk500_getsync() attempt 1 of 10: not in sync:
resp=0x55 avrdude: stk500_rcv(): programmer is not responding
avrdude: stk500_getsync() attempt 2 of 10: not in sync: resp=0x55
avrdude: stk500_rcv(): programmer is not responding avrdude:
stk500_getsync() attempt 3 of 10: not in sync: resp=0x55 avrdude:
stk500_rcv(): programmer is not responding avrdude: stk500_getsync()
attempt 4 of 10: not in sync: resp=0x55 avrdude: stk500_rcv():
programmer is not responding avrdude: stk500_getsync() attempt 5 of 10:
not in sync: resp=0x55 avrdude: stk500_rcv(): programmer is not
responding avrdude: stk500_getsync() attempt 6 of 10: not in sync:
resp=0x55 avrdude: stk500_rcv(): programmer is not responding
avrdude: stk500_getsync() attempt 7 of 10: not in sync: resp=0x55
avrdude: stk500_rcv(): programmer is not responding avrdude:
stk500_getsync() attempt 8 of 10: not in sync: resp=0x55 avrdude:
stk500_rcv(): programmer is not responding avrdude: stk500_getsync()
attempt 9 of 10: not in sync: resp=0x55 avrdude: stk500_rcv():
programmer is not responding avrdude: stk500_getsync() attempt 10 of
10: not in sync: resp=0x55 Error during upload: uploading
error: exit status 1
```

Ой! Не беспокойтесь, мы
можем это исправить



4 Иногда, при использовании блочного программирования и попытке загрузить программу на порт, появляется ошибка в красном цвете в Otto Blockly.



Возможное решение:

- убедитесь, что ваш робот подключен к компьютеру
- выберите правильный порт в настройках программы
- Выбран ESP8266

ottoblockly



Load the settings for the board ESP12E NodeMcu ?

OK

Отмена



ESP8266

COM3



not left D7 right D8 buzzer D5

Uploading...: OK code uploaded

5

Успешная загрузка кода, когда выбрана правильная плата – **ESP8266** или по другому – **ESP12E**.

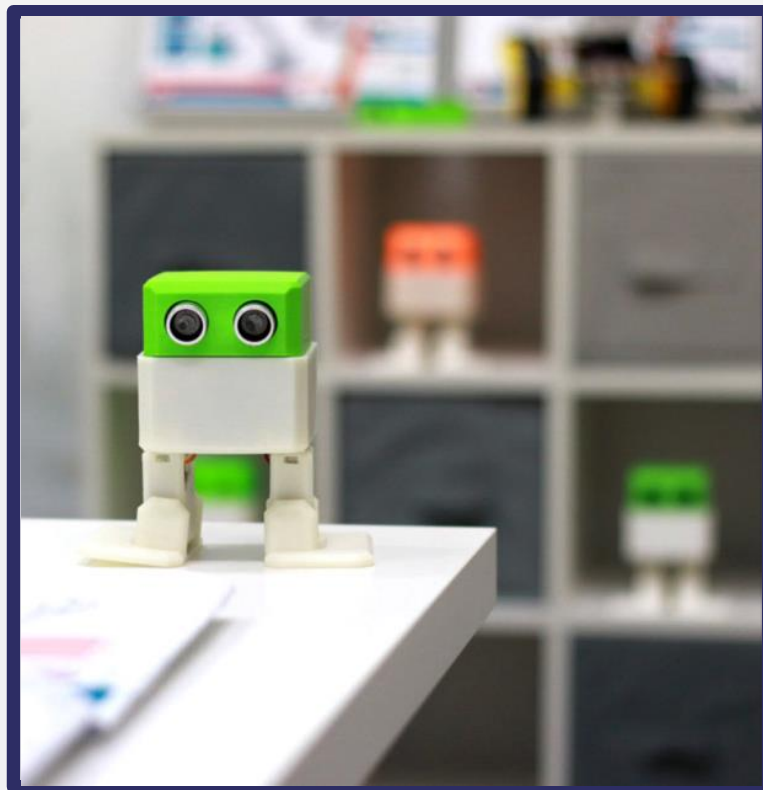
Это означает, что ваш код был загружен в Otto! И теперь ваш робот должен танцевать.

Ура, вы это сделали, поздравляю! Увидимся



Быстрый старт резюме

1. Установить **Otto Blockly**
2. Открыть пример
3. Подключите свой **Otto**
4. Выберите **ESP8266**
5. Выберите **USB COM#**
6. Нажмите «Загрузить»
7. И немного подождите



Распространенные проблемы:

1. Отто не подключается?

- Установите драйвер **USB CH340**, чтобы увидеть его в **COM**-порту.

2. Не можете загрузить код?

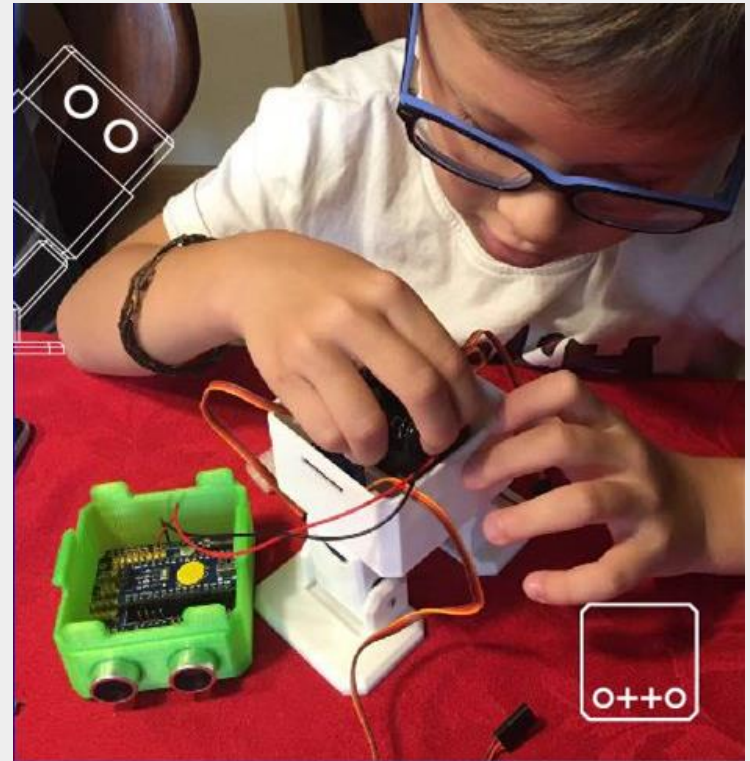
- Выбрана неправильная плата (попробуйте выбрать **ESP8266**)

3. Сбрасывает ли **Otto** время от времени?

- Низкое качество **USB** разъёма или плохое соединение проводом.

4. У Отто вывернуты ноги и ступни?

- Убедитесь, что вы отцентрировали и выровняли сервоприводы при сборке.



Теперь давайте
изучим ОСНОВЫ

Уровни и язык

The image displays three sequential screenshots of a software interface. Each screenshot shows a 'Level' selector at the top with options 1, 2, and 3. The first screenshot has level 1 selected. The second has level 2 selected. The third has level 3 selected. To the right of the level selector is a language dropdown menu currently set to 'GB English'. In the third screenshot, this dropdown menu is open, showing a list of languages: GB English, CN 汉语, CZ Čeština, DE Deutsch, ES Español, FR Français, GB English (highlighted in blue), HU Magyar, IL עברית, IT Italiano, PL Polski, PT Português, RU Русский, TR Türk, and TW 漢語.

Помогите нам
улучшить и
добавить новые
языки здесь

Категории в меню (Toolbar)

Structure

Otto

Sensors

Displays

Math

Logic

Variable

Function

Audio

Text

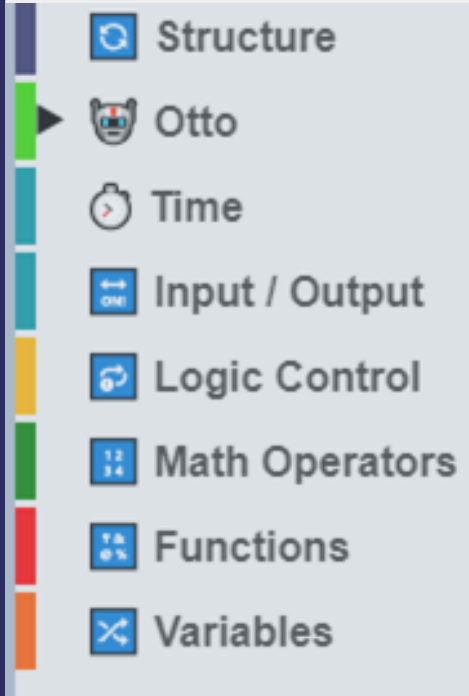
Motor

Time

IoT

Cameras

Communication



Level 1 (Уровень 1)

Меню для начинающих со всеми основными движениями Отто

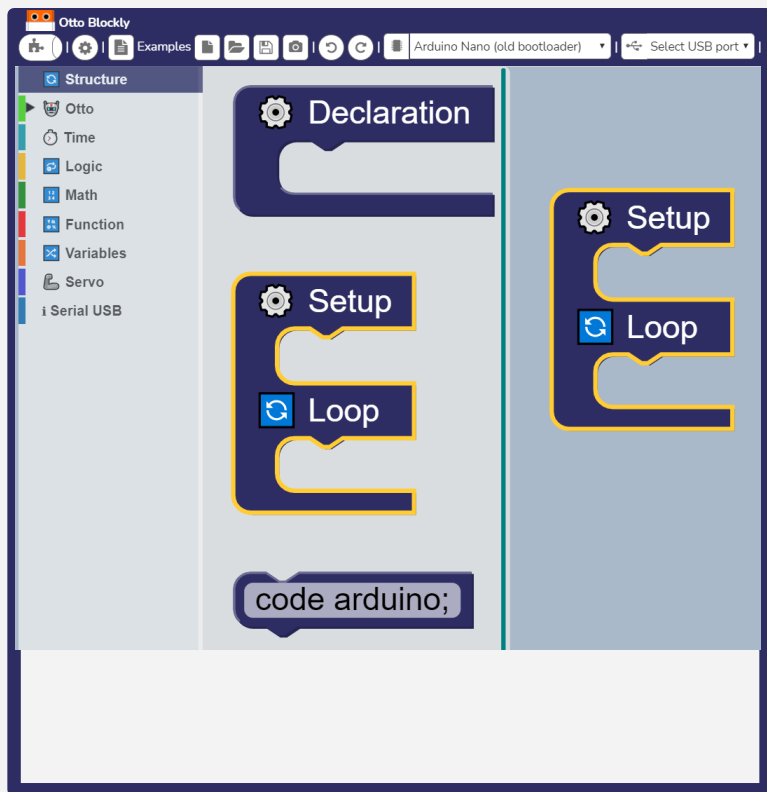
- **Structure** (Структура): Настройка и основы программирования.
- **Otto** (Отто): Движения, звуки и жесты.
- **Sensors** (Датчики): Работа с ультразвуковыми, сенсорными, звуковыми и гироскопическими датчиками.
- **Time** (Время): Использование времени в программах.
- **Input/Output** (Ввод/вывод): Управление портами **Arduino GPIO**.
- **Logic** (Логика): Создание условий, циклов и логического программирования.
- **Math** (Математика): Выполнение математических операций и вставка чисел.
- **Functions** (Функции): Создание повторяющихся процедур.
- **Variables** (Переменная): Работа с именованными значениями, которые можно изменять.



- ▶ Sensing
- ▶ Motor
- ▶ LED
- ▶ Displays
- ▶ Audio Sound
- ▶ Communication
- ▶ Mouse & Keys
- ▶ MuVision

Уровни 2 и 3 включают в себя более продвинутые инструменты для программирования Arduino и более широкий выбор возможностей:

- **Sensing** (Датчики): Возможность интеграции различных датчиков в проекты Отто или в создание других проектов с использованием Arduino.
- **Motor** (Мотор): Управление всеми видами сервоприводов, шаговых двигателей и постоянных моторов.
- **LED** (Светодиоды): Возможность управления RGB-светодиодами, Neopixels и светодиодными матрицами.
- **Displays** (Дисплеи): Использование дисплеев, таких как ЖК-дисплеи, OLED и другие виды экранов.
- **Audio** (Звук): Воспроизведение звука с помощью звуковых излучателей или воспроизведение MP3-файлов.
- **Communication** (Связь): Взаимодействие с различными модулями, такими как Bluetooth, IrDa или последовательная связь.
- **Keyboard and Mouse** (Клавиатура и мышь): Доступно только для Arduino Leonardo и Micro, это позволяет управлять Отто с клавиатуры и мыши.
- **MuVision** (MuVision): Запись видео или более сложное взаимодействие с распознаванием изображений.



Setup / Loop

Важно всегда использовать этот блок. Сначала перейдите в раздел **"Structure"** на левой боковой панели. Для начала выберите блок **"Setup / Loop"** и кликните на него. Блок **"Loop"** открывает и завершает последовательность функций, которая будет повторяться бесконечно. Если вы хотите, чтобы ваш код выполнялся только один раз, используйте блок **"Setup"**. Этот блок также используется для инициализации компонентов и настройки пинов.

Перетаскивание блоков

Из панели инструментов выберите "Otto", перетащите блок конфигурации из списка в блок "Setup" и другие блоки в структуру "Loop". Пространство внутри Setup и Loop англичане называют Брекеты.

The screenshot displays the LEGO Mindstorms software interface. On the left is a vertical toolbar with categories: Structure, Robot, Wheels, Ninja, Legs (highlighted in blue), Arms, Display, Sound, Sensing, Communicate, and Timing. The main workspace is divided into two sections. The left section shows a list of available blocks: a configuration block for 'biped' (with leg left D3, right D4, foot left D7, right D8, and buzzer D5), a 'home' block, a 'move' block (set to forward, normal speed), a 'dance' block (set to moonwalk, normal speed, normal size), a 'do' block (set to swing, normal speed, normal size), a 'gesture' block (set to happy1), and a detailed 'move' block (set to leg left 90°, right 90°, foot left 90°, right 90°, speed 200). The right section shows a 'Setup' loop containing the configuration block and a 'Loop' loop containing the 'move' block. Two orange arrows originate from the configuration block in the left section and point to the configuration block inside the 'Setup' loop and the 'move' block inside the 'Loop' loop, illustrating the drag-and-drop process.

Servo centering example

- Центровка Сервоприводов

10 | 🌈 RGB LED Circuit |

11 | 🦾 Servo centering |

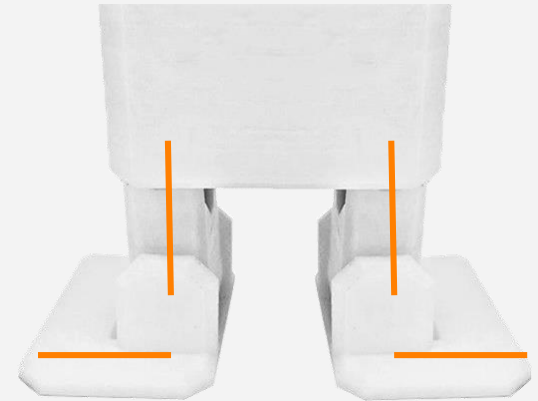
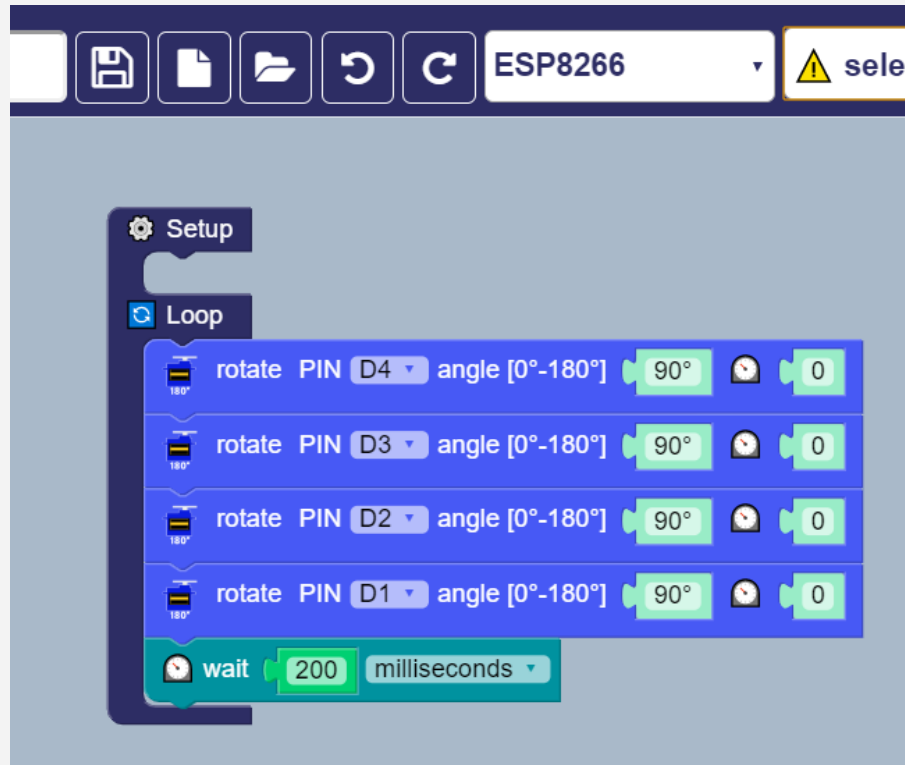
12 | 🦾 Servo sweep |

13 | 🤖 Biped calibration |

14 | 🤖 Biped walk |

15 | 🤖 Biped dance |

Upload code – Загрузите на робота этот скетч



После загрузки кода ноги робота должны быть выровнены и находиться в центре, насколько это возможно. Если это не так, вам, возможно, придется пересобрать сервоприводы под правильным углом.

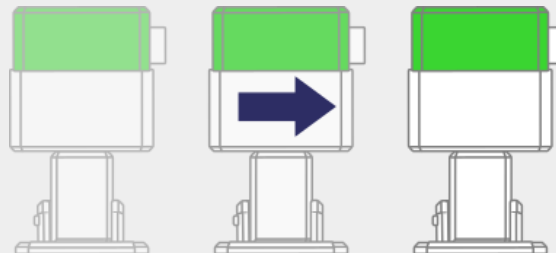
Ходьба перед бегом

Move – блок ходьбы

А в C++ коде:



```
Otto.walk(1,1000,1);
```



Сможете ли вы
заставить Отто
бежать?

Loop

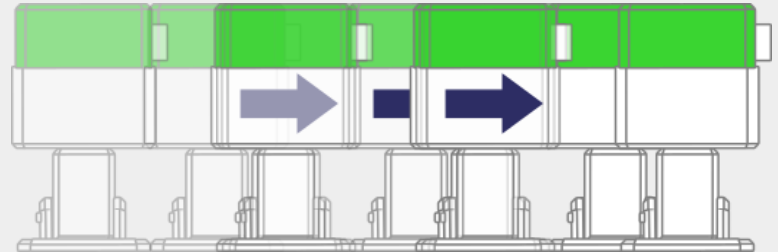


Move ↑ forward ▾

Speed very fast ▾

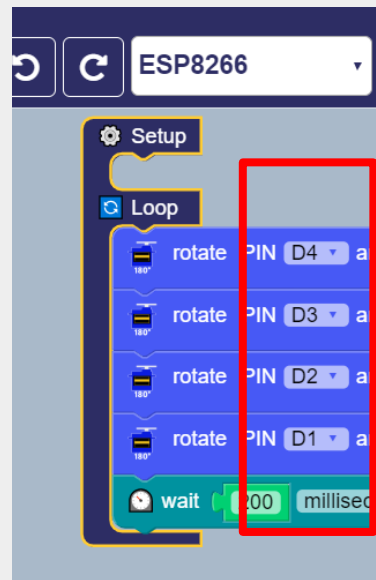
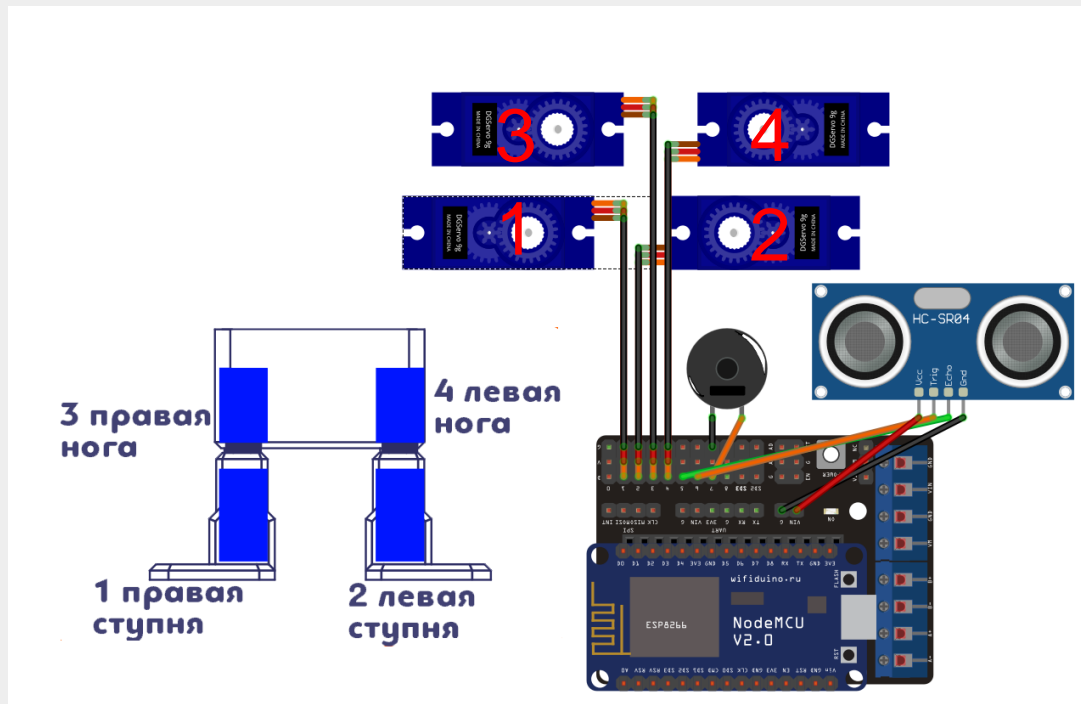
- normal
- slow
- very slow
- fast
- ✓ very fast
- way to fast

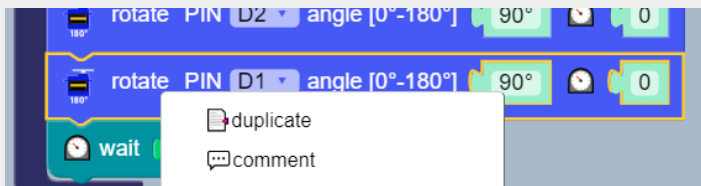
Otto.walk(1,500,1);



Подключение сервоприводов

Пример в Blockly для управления каждым двигателем





Добавьте ещё один блок Servo и Time delay

Нажмите дублировать (**duplicate**) на блоке вращение (**rotate**)

1. С помощью этого блока пошевелите левой ступнёй
2. Затем покрутите правой ступнёй
3. Затем вращайте левой и правой ногами

На этом этапе вы должны понимать:

- какой пин на моторшilde отвечает за какую часть какой ноги
- Как работает цикл и задержки в программе

