

Введение в программирование роботов Ev3 на языке MicroPython

Автор:

Трофимов Александр Артемович

ПДО робототехника

МБУДО «ГЦДТТ им. В.П. Чкалова» г.Казани

Оглавление

Подготовка

VS Code

MicroPython

Стадийная модель работы робота

Полезные ссылки

Подготовка

EV3DEV

Установка образа на MicroSD флэш-карту

EV3DEV

- EV3DEV - это образ операционной системы Linux, собранный специально для роботов EV3.
- На EV3DEV установлена простая программа-интерфейс Brickman версии 0.1.
- Скачать образ можно либо с сайта разработчика, либо по ссылке, которую предоставляет сама компания LEGO:

<https://le-www-live-s.legocdn.com/sc/media/files/ev3-micropython/ev3micropythonv100sdcardimage-4b8c8333736fafa1977ee7accbd3338f.zip>

Установка образа на MicroSD флэш-карту

- Скачанный образ не получится просто скинуть на флэш-карту. Прежде всего надо его распаковать, чтобы был отдельно файл `ev3-micropython-v1.0.0-sd-card-image.img`
- Теперь требуется скачать программу Win32 Disk Imager по ссылке:
<https://sourceforge.net/projects/win32diskimager/>
- Установив ее, запускаем.
- !!! Следующая операция переразовет флэш-карту на два раздела и использовать ее по назначению станет невозможно до повторного форматирования. Чтобы уменьшить головную боль, перед следующим шагом, сделайте образ флэш-карты на кнопку «Read». Потом вы сможете восстановить функционал флэш-карты через эту же программу.
- Выбираем файл, выбираем букву флэш-карты, жмем кнопку «Write».
- После сообщения о конце записи, можно вставлять флэш-карту в блок EV3 и включать его обычным способом. Если загрузилась обычная программа - требуется почистить контакты путем вытаскивания и вставления флэш-карты несколько раз.

VS Code

О VS Code

Установка

Настройка

Подключение Ev3

O VS Code

- Visual Studio Code - бесплатное открытое программное обеспечение от компании Microsoft, которое позволяет быстро и удобно программировать на большинстве языков программирования.
- VS Code изначально не сильно отличается от навороченного блокнота, но за счет расширений - скачиваемых дополнительных подпрограмм, ее можно использовать для написания приложений и сайтов, для написания и загрузки прошивок в устройства на базе Arduino и других микроконтроллеров, а главное - для программирования EV3-роботов с помощью Python.

Установка

Visual Studio Code Docs

Code editing Redefine

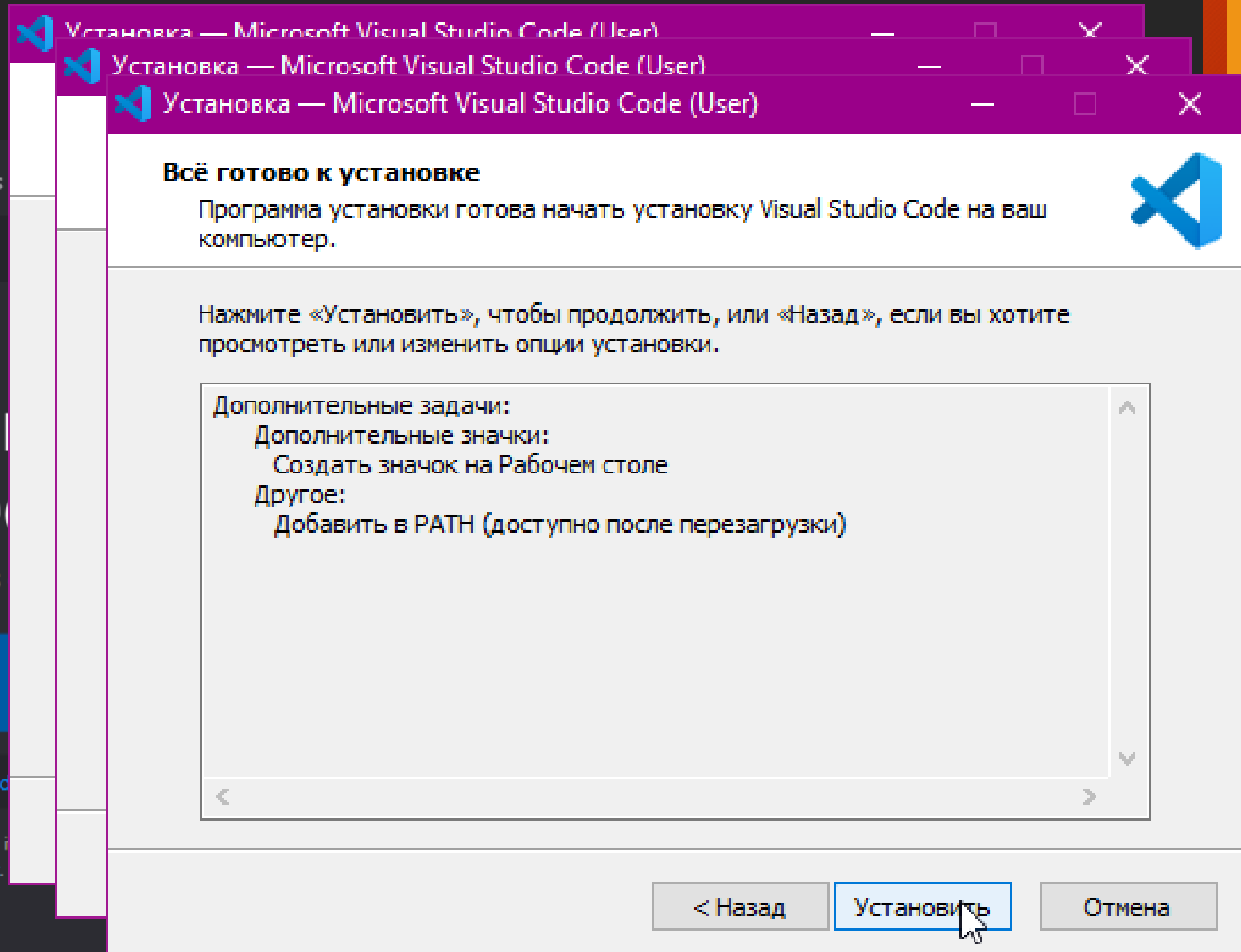
Free. Built on open source. Runs

Download for Windows

Stable Build

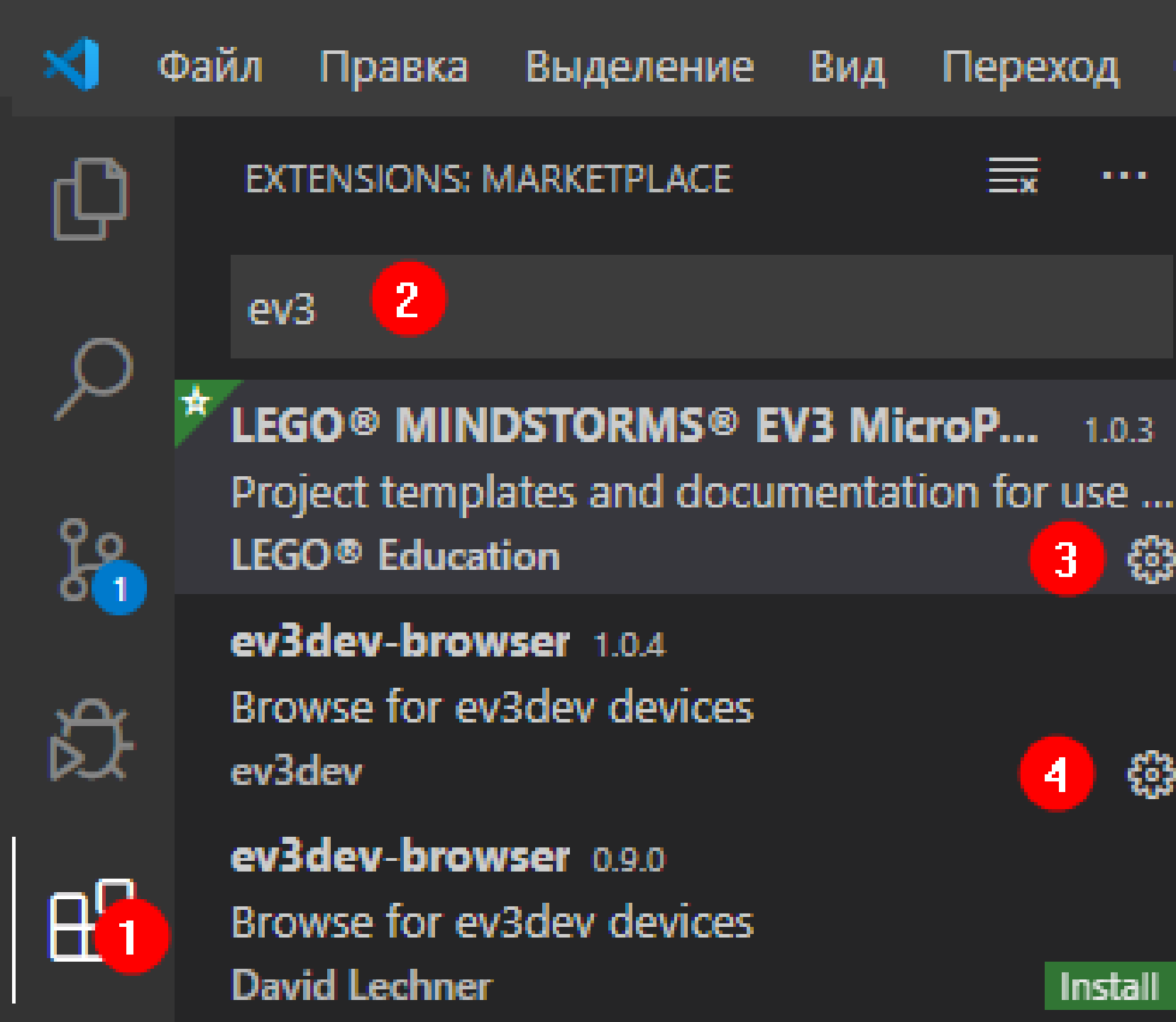
Other platforms and Insiders Edition

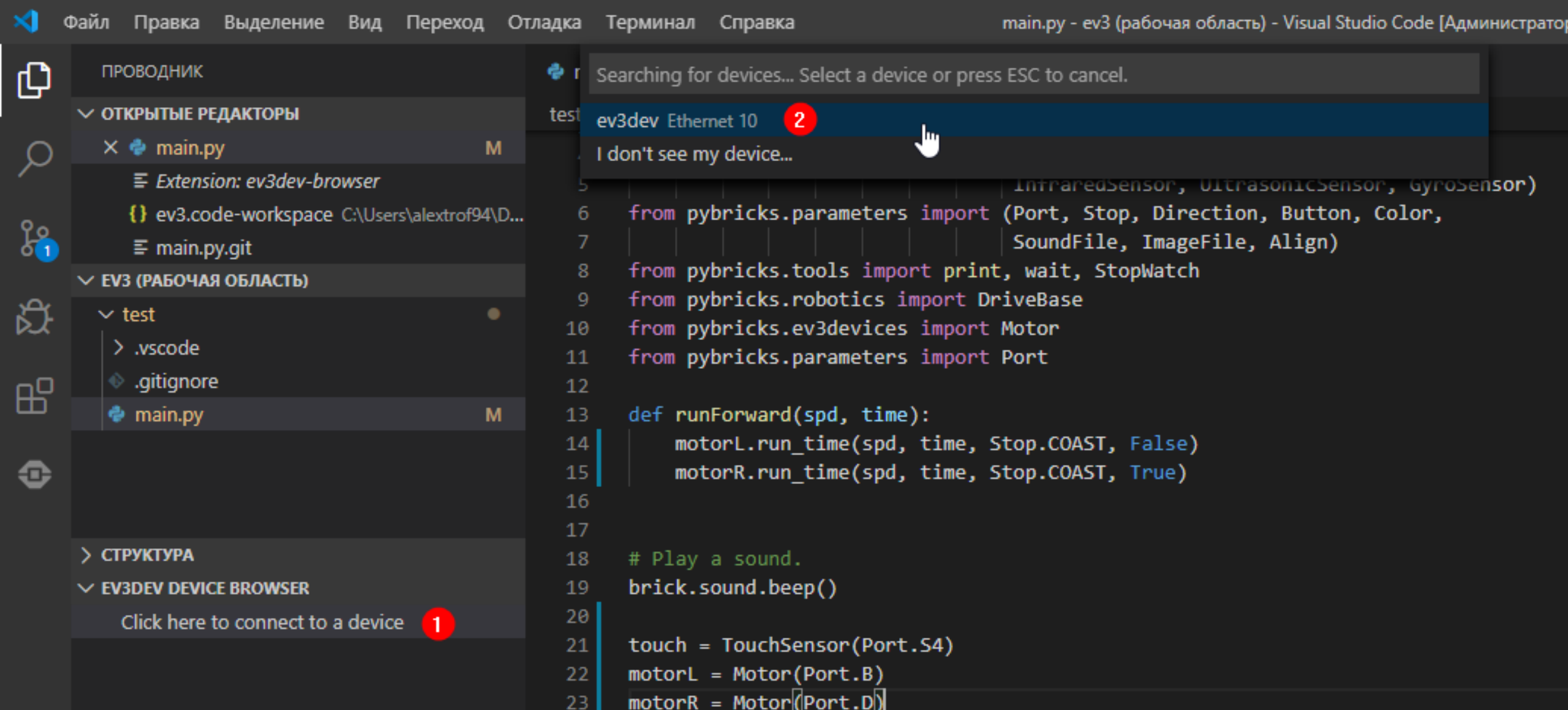
By using VS Code, you agree to our
license and privacy statement.



Настройка

- В первую очередь надо установить русский язык
 - Заходим в расширения
 - Вводим в фильтр часть
 - Нажимаем зеленую кнопку Pack for Visual Studio Code
- Аналогичным образом для EV3
 - LEGO® MINDSTORMS® EV3 MicroPython
 - ev3dev-browser





MicroPython

О MicroPython

Основные конструкции языка

Классы и объекты

Заставляем робота двигаться

Функции

О MicroPython

- MicroPython - вариация языка Python, разработанная для микроконтроллеров.
- Программа пишется используя отступы. Внутренняя часть условия, цикла или отдельной функции обозначается только отступами.
- Точки с запятыми ставить в конце каждого оператора не требуется.
- Для работы с роботом, требуется импортировать некоторые библиотеки и классы.

Основные конструкции языка

- Условие `if`
 - `if` условие :
- Цикл `while`
 - `while` условие :
- Цикл `for`
 - `for i in (начало, конец_не_включительно) :`
- Функции
 - `def funcName(param) :`
 - Функции вызываются просто по названию, например `funcName(127)`
 - `param` - переменная, которую можно использовать внутри функции

Классы и объекты

- Вся работа с роботом построена на классах
- Сначала мы должны объявить экземпляр объекта класса
 - `motorR = Motor(Port.B)`
 - Здесь `motorR` - наш объект для мотора, `Motor()` - конструктор класса, `Port` - статический класс и `B` - переменная из этого статического класса
- После объявления, можно использовать этот объект вызывая его методы и изменяя его поля, если они имеются:
 - `motorR.run_time(127, 3000, Stop.COAST, False)`
 - В данный метод класса мы передаем несколько параметров:
 - 127 - скорость
 - 3000 - время в мс
 - `Stop.COAST` - способ остановки
 - `False` - ожидать ли окончания данного действия

Заставляем робота двигаться

```
brick.sound.beep() # Издать стандартный звук
motorL = Motor(Port.B) # Мотор на порте B
motorR = Motor(Port.C, Direction.COUNTERCLOCKWISE, [8, 40]) # Мотор
на порте C с понижающей передачей 8:40
motorL.run_time(300, 1000, Stop.COAST, False) # Крутим левым мотором
не ожидая конца действия
motorR.run_time(300, 1000, Stop.COAST, True) # Крутим правым мотором
с ожиданием
brick.sound.beep(1000, 500) # Издать звук 1000Гц 0.5с
```

Функции

- Функции служат для уменьшения повторений одного и того же кода, что упрощает разработку. Например мы можем написать две функции для езды вперед и для поворота, а затем используя их писать программу используя каждый раз на 1 строку кода меньше. Это не очень показательный пример, но если мы попробуем создать функцию, которая будет повторять функцию блока рулевого управления из Ev3-G, она займет минимум 20 строк, а ее использование так и останется однострочным.

```
def runForward(spд, time):  
    motorL.run_time(spд, time, Stop.COAST, False)  
    motorR.run_time(spд, time, Stop.COAST, True)
```

```
def turn(spд, time):  
    motorL.run_time(spд, time, Stop.COAST, False)  
    motorR.run_time(-spд, time, Stop.COAST, True)
```


Стадийная модель работы робота

Основа

Пример кода

Основа

- У нас есть функции по количеству стадий и переменная, которая хранит номер нашей стадии.
- Переключая номер нашей стадии мы можем отслеживать определенные датчики в определенные моменты.
- Данная модель отлично подходит для сложных программ, с четко отслеживаемыми стадиями:
 - Со стадией настройки в многоуровневом меню и стадией работы
 - Со стадиями поиска и выталкивания оппонента
- Стадия может быть проходной, т.е. сделать какое-либо действие и сразу же передать управление на другую стадию.

Пример кода

- Функции-стадии:

```
def modeFinding() :  
    global mode  
    startTurn()  
    if us.getDistance() < 700 : #mm  
        mode = 1  
  
def modeKick() :  
    global mode  
    startRun()  
    if us.getDistance() > 700 : #mm  
        mode = 0  
    if color.getColor() < 10 : # percent  
        mode = 0
```

- Основной код программы:

```
mode = 0  
while (True) :  
    if mode == 0 :  
        modeFinding()  
    elif mode == 1 :  
        modeKick()  
    else :  
        modeError()
```


Заключение

Полезные ссылки

Основное руководство пользователя

Вики

Что дальше?

Полезные ссылки

- [Основное руководство пользователя](#)
- [Прошивка для флешки](#)
- [Win32 Disk Imager](#)
- [VS Code](#)

- [Страница на LEGO Education](#)
- [Wiki](#)

Что дальше?

- MicroPython позволяет использовать большинство библиотек для Python 3, что означает практически неисчерпаемый функционал:
 - Это и обработка изображений с камеры, подключенной к USB разъему
 - Продвинутая обработка данных, их хранение и передача, например по USB WiFi модулю, ведь из под Linux количество поддерживаемых USB WiFi модулей увеличивается с 2-х до огромного количества
- Помимо самого MicroPython, к блоку EV3 можно подключиться по SSH напрямую в ОС Linux, что расширяет функционал до полноценного, хоть и медленного, компьютера.

Спасибо за внимание!