СОДЕРЖАНИЕ

Об авторе *11*

О техническом рецензенте *11*

Благодарности *12*

Введение *13*

1. Начало *18*

Экскурсия по micro:bit *19*
Верхняя сторона *19*
Нижняя сторона *21*
Подключение питания *21*
Подключение электронных устройств к входным и выходным контактам *24*
Встроенные периферийные устройства *27*
Основное оборудование *28*
Программирование micro:bit *29*
Подключение micro:bit *29*
Программирование с помощью Blocks: Hello World *30*
Программирование на MicroPython: Hello World *38*
Основные понятия в программировании *45*
Переменные *45*
Арифметика *48*
Условный оператор if *49*
Строки *51*
Массивы и списки *53*
В заключение о программировании *55*
Скачивание программ *55*
Скачивание программ на языке Blocks *55*
Скачивание программ на языке MicroPython *57*
Итоги *59*

2. Мир звука *60*

Подключение динамика к micro:bit *61*
Тихий способ: наушники *61*
Громкий способ: колонки *63*
Эксперимент 1: генерация звуков *64*
Что понадобится *64*
Конструирование *64*
Программа *65*
Что можно попробовать *66*
Как это работает: частота и звук *67*

Эксперимент 2: оно говорит! ... 69
Что понадобится ... 69
Конструирование ... 69
Программа ... 70
Проект: музыкальный дверной звонок ... 70
Что понадобится ... 71
Конструирование ... 72
Программа ... 74
Что можно попробовать ... 75
Проект: шумомер ... 77
Что понадобится ... 77
Конструирование ... 77
Программа ... 79
Как это работает: выход микрофона ... 81
Итоги ... 82

3. Да будет свет ... 83

Эксперимент 3: датчик освещенности ... 84
Что понадобится ... 84
Конструирование ... 84
Программа ... 85
Как это работает ... 85
Проект: автоматический ночник ... 85
Что понадобится ... 86
Конструирование ... 86
Программа ... 87
Проект: световая гитара ... 88
Что понадобится ... 89
Конструирование ... 89
Программа ... 93
Проект: бесконечные отражения ... 95
Что понадобится ... 96
Конструирование ... 98
Программа ... 105
Как это работает ... 107
Итоги ... 108

4. Волшебный магнетизм ... 109

Проект: компас ... 110
Что понадобится ... 110
Конструирование ... 111
Программа ... 113
Что можно попробовать ... 116
Как это работает: магнитное поле Земли ... 116
Эксперимент 4: измерение магнитных полей ... 117
Что понадобится ... 117
Конструирование ... 118

Программа 121
Что можно попробовать 123
Как это работает: сила магнитов 123
Проект: магнитная сигнализация открывания двери 124
Что понадобится 124
Конструирование 125
Программа 126
Что можно попробовать 128
Итоги 128

5. Удивительное ускорение 129

Эксперимент 5: жесты 130
Что понадобится 131
Конструирование 131
Программа 131
Что можно попробовать 134
Как это работает: сила, ускорение и гравитация 134
Эксперимент 6: построение графика ускорения в реальном времени 137
Что понадобится 137
Конструирование 137
Программа 139
Как это работает: расчет суммарного ускорения 140
Проект: детектор тщательности чистки зубов 142
Что понадобится 142
Конструирование 143
Программа 143
Что можно попробовать 147
Эксперимент 7: запись данных об ускорении в файл 147
Что понадобится 148
Конструирование 148
Программа 151
Что можно попробовать 152
Проект: акселерометр 153
Что понадобится 154
Конструирование 154
Программа 155
Итоги 157

6. Волшебство движения 158

Эксперимент 8: запуск сервомотора 159
Что понадобится 160
Конструирование 160
Программа 162
Как это работает: сервомоторы и импульсы 164
Проект: аниматронная голова (робот Mike на плате micro:bit) 168
Что понадобится 168
Конструирование 169

Программа 180
Что можно попробовать 184
Проект: робот-вездеход 185
Что понадобится 186
Конструирование 187
Как это работает: электромоторы и поток электроэнергии 194
Итоги 195

7. Путешествие во времени 196
Эксперимент 9: счет времени 197
Что понадобится 198
Конструирование 198
Программа 199
Как это работает: счет времени 200
Проект: двоичные часы 201
Как читать показания двоичных часов 202
Что понадобится 203
Конструирование 203
Программа 204
Как это работает: вывод времени в двоичном формате 208
Проект: говорящие часы 210
Что понадобится 211
Конструирование 211
Программа 212
Как это работает: обучаем micro:bit говорить 215
Итоги 216

8. Игры разума 217
Эксперимент 10: скорость реакции 218
Что понадобится 219
Конструирование 219
Проверка вашей нервной системы 221
Программа 223
Что можно попробовать 226
Как это работает: измерение времени реакции 226
Проект: детектор лжи 229
Что понадобится 229
Конструирование 230
Программа 231
Как это работает: обнаружение лжи по напряжению и сопротивлению 233
Итоги 235

9. Помешательство на экологии 236
Эксперимент 11: измерение температуры 237
Что понадобится 237
Конструирование 237

Программа ... 239
Проект: регистратор температуры и освещенности ... 241
Что понадобится ... 243
Конструирование ... 243
Программа ... 245
Как это работает: датчики ... 248
Проект: автоматический полив растений ... 250
Что понадобится ... 251
Конструирование ... 252
Программа ... 256
Что можно попробовать ... 260
Как это работает: измерение влажности почвы ... 261
Итоги ... 262

10. Радиосвязь ... 263
Эксперимент 12: определение дальности радиосвязи ... 264
Что понадобится ... 264
Конструирование ... 265
Программа ... 266
Как это работает: радиосигналы ... 269
Проект: беспроводной дверной звонок ... 270
Что понадобится ... 271
Конструирование ... 271
Программа ... 272
Что можно попробовать ... 274
Как это работает: отправка и получение ... 274
Проект: радиоуправляемый робот-вездеход ... 274
Что понадобится ... 276
Конструирование ... 276
Программа ... 277
Что можно попробовать ... 280
Как это работает: блоки управления электромотором ... 280
Итоги ... 282

Приложение ... 283

Предметный указатель ... 289

ОБ АВТОРЕ

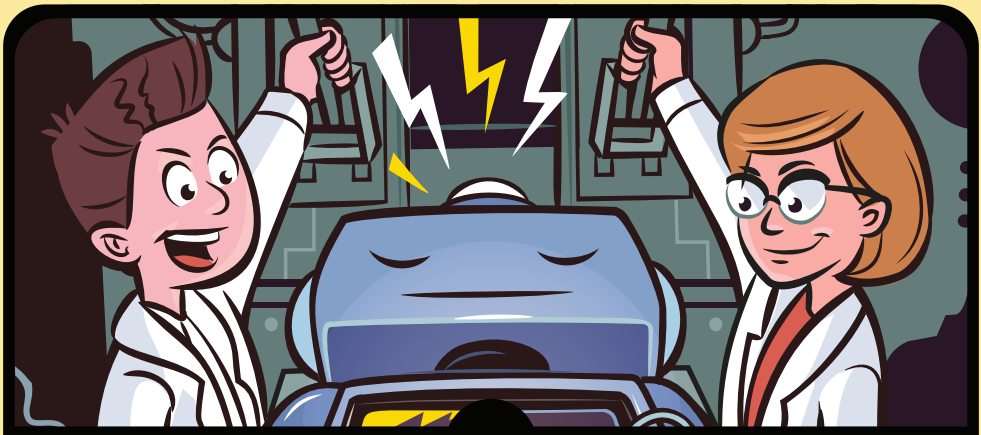
Саймон Монк имеет инженерное образование в области кибернетики и информатики, а также докторскую степень в области программной инженерии. Много лет занимался разработкой программного обеспечения и основал компанию Momote, производящую программное обеспечение для мобильных устройств. В настоящее время Саймон пишет книги об электронике и программировании и помогает своей жене Линде (Linda) управлять их совместной компанией Monk Makes (*https://www.monkmakes.com/*), где занимается разработкой комплектов электронных компонентов и принадлежностей для BBC micro:bit и Raspberry Pi.

Вы можете последовать за Саймоном в Twitter (@simonmonk2) и узнать больше о его книгах на сайте *https://www.simonmonk.org/*.

О ТЕХНИЧЕСКОМ РЕЦЕНЗЕНТЕ

Дэвид Вэйл (David Whale) – инженер-разработчик программного обеспечения для встраиваемых устройств и на добровольных началах занимается популяризацией точных наук в школах Великобритании. Активный член сообществ Raspberry Pi и micro:bit с момента их основания. Участвовал в развитии проекта BBC micro:bit, консультировал компании IET и BBC, а также помогал писать книги и оказывал помощь в повышении квалификации учителей по всей стране. Он написал для детей очень интересную книгу по программированию «Adventures in Minecraft»[1] (Wiley) и рецензирует самые разные книги и статьи известных авторов. Цель Дэвида – вдохновить подрастающее поколение инженеров и ученых, потому что наше будущее скоро окажется в их руках.

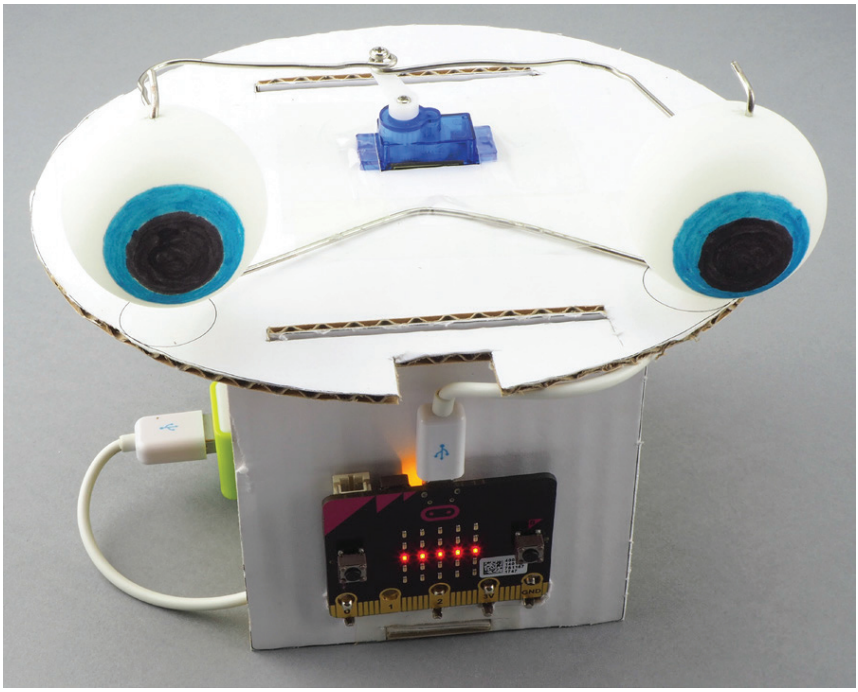
[1] *Вэйл Дэвид, О'Хэнлон Мартин.* Minecraft. Программируй свой мир на Python. СПб.: Питер, 2018. ISBN 978-5-4461-0951-7. – *Прим. перев.*



ВВЕДЕНИЕ

С момента появления BBC micro:bit в 2016 году были выпущены миллионы этих устройств. Они нравятся и детям, и взрослым во всем мире. Устройство micro:bit создавалось с целью обучения детей навыкам программирования. Самое большое его преимущество состоит в том, что для работы с ним не нужно ничего, кроме USB-кабеля и компьютера. Кроме того, запрограммировав это устройство, его можно отключить от компьютера – и оно благополучно продолжит работать от батареек.

В комплект micro:bit входит небольшой светодиодный дисплей, а также датчики света, движения и магнитного поля, то есть в нем есть все необходимое для создания интересных проектов. Набравшись немного опыта, вы с легкостью сможете подключать к устройству электромоторчики, датчики и динамики с помощью простых зажимов «крокодилов» – вам не придется ничего паять. Другими словами, ваше устройство micro:bit может стать *мозгом* для самых разных проектов и изобретений.



ОБ ЭТОЙ КНИГЕ

В этой книге вы найдете описание множества занимательных экспериментов и проектов. В экспериментах я расскажу, как все работает, а в проектах мы с вами используем полученные знания, чтобы создать что-нибудь необычное.

Книга разделена на 10 глав. В главе 1 я расскажу все, что нужно знать о подключении и использовании micro:bit. Эти знания потребуются нам во всех экспериментах и проектах, описанных в данной книге. Каждая следующая глава посвя-

щена определенной теме, например свету, звуку и движению. С помощью micro:bit можно сконструировать множество забавных и полезных устройств!

Эксперименты

Вот список экспериментов в этой книге.

Генерация звуков. Здесь вы узнаете, как воспроизводить музыкальные ноты и другие звуки с помощью micro:bit.

Оно говорит! Тут вы научите micro:bit говорить!

Датчик освещенности. Здесь я расскажу, как использовать встроенный датчик освещенности.

Измерение магнитных полей. Тут мы будем учиться использовать встроенный магнитометр для измерения магнитных полей.

Жесты. Здесь мы используем программу распознавания жестов, чтобы научить micro:bit выполнять разные действия при встряхивании или падении.

Построение графика ускорения в реальном времени. В этом эксперименте вы познакомитесь с программой Mu, предназначенной для визуализации данных.

Запись данных об ускорении в файл. Здесь мы будем учиться записывать в файл данные об ускорении, получаемые с устройства micro:bit, чтобы иметь возможность просмотреть их позже.

Запуск сервомотора. Эксперимент с сервомотором!

Счет времени. Здесь я расскажу, как micro:bit определяет время.

Скорость реакции. Тут вы сможете проверить время своей реакции.

Измерение температуры. В данном эксперименте мы воспользуемся датчиком температуры в micro:bit и сконструируем электронный термометр.

Определение дальности радиосвязи. Здесь вы узнаете, как из micro:bit сконструировать простой радиопередатчик.

Проекты

Вот список проектов в этой книге.

Музыкальный дверной звонок. При нажатии на кнопку воспроизводит выбранную вами мелодию.

Шумомер. Измеряет и показывает громкость звука.

Автоматический ночник. Автоматически включает освещение, когда в комнате становится темно.

Световая гитара. Издает звуки в такт движениям руки над светодиодами micro:bit.

Бесконечные отражения. Создает с помощью света иллюзию бесконечной глубины.

Компас. Настоящий рабочий компас!

Магнитная сигнализация открывания двери. Срабатывает при открытии двери, когда магнит отдаляется от micro:bit.

Детектор тщательности чистки зубов. Подсчитывает движения зубной щетки, чтобы убедиться, что вы сохраните ваши жемчужно-белые зубы в хорошем состоянии.

Акселерометр. Измеряет и показывает, с каким ускорением перемещается micro:bit.

Аниматронная голова. Голова робота с движущимися глазами и говорящим ртом.

Робот-вездеход. Двухколесный робот, управляемый micro:bit!

Двоичные часы. Показывают время на светодиодном дисплее.

Говорящие часы. Объявляют время каждый час и всякий раз, когда вы нажимаете кнопку.

Детектор лжи. Измеряет электропроводимость кожи, чтобы определить правдивость испытуемого.

Регистратор температуры и освещенности. Автоматически регистрирует уровень освещенности и температуру.

Автоматический полив растений. Включает полив растений, когда датчик влажности определяет, что почва слишком сухая. (Ваши растения будут счастливы!)

Беспроводной дверной звонок. Беспроводная версия проекта дверного звонка, в которой используются радиоволны.

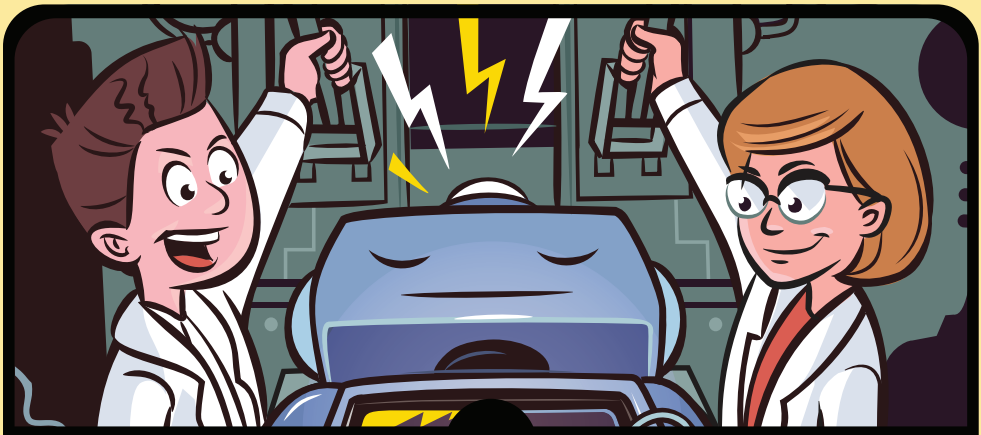
Радиоуправляемый робот-вездеход. Беспроводная версия робота-вездехода, который получает команды по радио.

ИСХОДНЫЙ КОД ПРОЕКТОВ

Наиболее популярные компьютерные языки для программирования micro:bit – это Makecode Blocks (далее мы будем называть его просто *Blocks*) и MicroPython.

Я подготовил программы для проектов и экспериментов на обоих языках, Blocks и MicroPython. Это означает, что вам не придется вводить код вручную, если, конечно, вы этого не захотите.

Весь исходный код для этой книги вы найдете на странице GitHub, по адресу: *https://github.com/simonmonk/mbms/*. В главе 1 я подробно расскажу, как скачать и использовать этот код.



1
НАЧАЛО

В этой главе я расскажу, как начать работу с BBC micro:bit, и подготовлю вас к встрече с экспериментами и проектами в следующих главах. Я подскажу вам несколько идей, что можно сделать с micro:bit, и расскажу, как программировать это устройство. Здесь вы также узнаете, как использовать код на языках Blocks и MicroPython.

Неугомонный ученый обычно очень неусидчив, чтобы вручную набирать длинные программы, поэтому исходный код всех программ, используемых в этой книге, доступен для загрузки. В данной главе я расскажу, как скачать и использовать этот код.

ЭКСКУРСИЯ ПО MICRO:BIT

А теперь давайте рассмотрим плату micro:bit и посмотрим, что на ней написано.

Верхняя сторона

На рис. 1.1 показана верхняя сторона платы micro:bit.

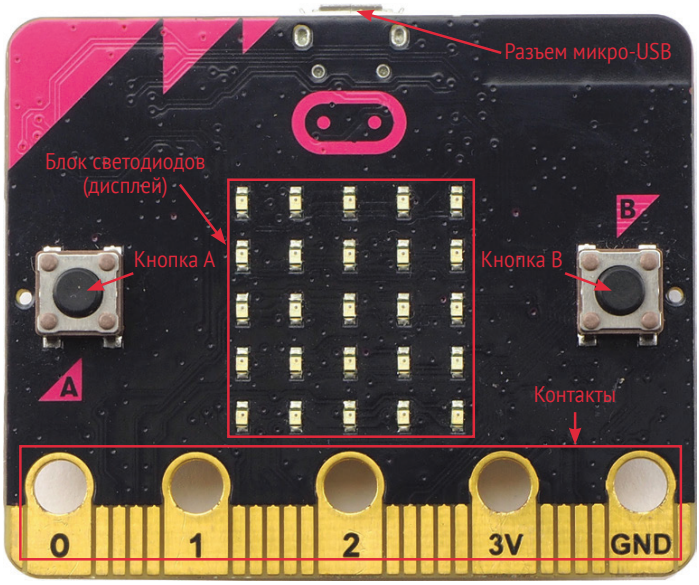


Рис. 1.1. Плата micro:bit

Вверху (на рис. 1.1) находится разъем микро-USB, с помощью которого micro:bit подключается к компьютеру. Программы для micro:bit вы будете писать на компьютере, поэтому вам придется переносить их на micro:bit, подключив устройство к компьютеру с помощью USB-кабеля. Кроме того, разъем микро-USB может использоваться для питания micro:bit.

Слева и справа (на рис. 1.1) находятся две кнопки, подписанные буквами A и B соответственно. Мы можем запрограммиро-

вать micro:bit так, чтобы при нажатии этих кнопок выполнялись некоторые действия, например мигали светодиоды или звенел дверной звонок.

Между кнопками находится блок из 25 светодиодов в виде решетки, состоящей из 5 рядов и 5 столбцов. Это дисплей micro:bit. Несмотря на то что на нем всего 25 светодиодов, этот дисплей может отображать текстовые сообщения в виде бегущей строки, небольшие изображения, узоры – много всего!

Внизу (на рис. 1.1) находится позолоченная (да, да, позолоченная настоящим золотом!) полоска, которая называется *краевым разъемом*. На полоске имеется пять отверстий, подписанных 0, 1, 2, 3V и GND. Эти большие контакты позволяют подключать дополнительные устройства к плате micro:bit с помощью зажимов типа «крокодил». Например, вы можете подключить динамик, чтобы устройство micro:bit могло издавать звук, или мотор, чтобы оно могло двигать что-то. Контакты меньшего размера, узкие полоски между отверстиями, можно использовать только с помощью специального адаптера. В этой книге мы будем применять только большие разъемы, и лишь в двух проектах, в которых мы будем конструировать роботов, нам потребуется адаптер, чтобы подключить электромотор.

Нижняя сторона

Теперь перевернем плату micro:bit и посмотрим, что имеется на обратной стороне (рис. 1.2).

Рис. 1.2. Обратная сторона платы micro:bit, версий 1.3B (слева) и 1.5 (справа)

Когда писались эти строки, существовало две версии micro:bit. Обе работают совершенно одинаково, и для этой книги абсолютно не важно, какая версия у вас. Последняя версия (1.5) отличается лишь немного упрощенным дизайном. Разницу можно увидеть в левом нижнем углу на рис. 1.2.

Слева вверху (на рис. 1.2) – это разъем микро-USB. Правее него находится микропереключатель. Это *кнопка сброса*. Нажав эту кнопку, можно заставить micro:bit перезапустить установленную в нем программу. Справа от кнопки сброса находится разъем для подключения батарейки или аккумулятора с напряжением 3 В.

Теперь мы подробно рассмотрим особенности использования micro:bit, начав с подключения питания.

Подключение питания

Подать напряжение электропитания на плату micro:bit можно через разъем микро-USB или разъем для подключения батарейки, в зависимости от того, что вы собираетесь делать.

Питание через микро-USB

При подключении micro:bit к компьютеру с помощью кабеля USB на плату подается напряжение питания 5 В (5 вольт). Однако микроконтроллеру требуется всего 3,3 В, а не 5 В, и слишком большое напряжение может повредить его. Поэтому на плате имеется микросхема интерфейса USB, которая преобразует эти 5 В в 3,3 В.

Когда плата micro:bit подключена к компьютеру кабелем USB, то контакт 3V на краевом разъеме можно использовать для подачи питания на слаботочные электронные устройства, такие как внешние светодиоды или динамики, предназначенные для работы с micro:bit.

ПРИМЕЧАНИЕ *Основная причина, почему контакт подписан как* 3V*, а не* 3,3V*, заключается в том, что для второй цифры просто не хватает места, а также потому, что схема защиты снижает напряжение 3,3 В до 3 В.*

Контакт с подписью GND – это *земля*, то есть линия электропитания 0 В. Когда к micro:bit подключено какое-то дополнительное устройство, ток, питающий это устройство, вытекает с контакта 3V и должен вернуться в micro:bit, чтобы замкнуть цепь. Контакт GND как раз и есть та точка, куда возвращается ток.

Питание от батареек

После того как вы запрограммируете свое устройство micro:bit, вам может понадобиться перенести его в другое место, далеко от компьютера, и в этом случае вам потребуются батарейки. Вы можете использовать блок батареек AAA, подобный изображенному на рис. 1.3. Чтобы включить электропитание, просто подключите батарейки к разъему для батареек, как показано на рис. 1.3.

В приложении (в конце книги) я перечислил несколько мест, где можно купить такие батарейки и аккумуляторы. Очень удобно, если на корпусе блока батареек будет иметься дополнительный выключатель, с помощью которого вы сможете включать и выключать micro:bit, не выдергивая провода, что намного безопаснее и проще.

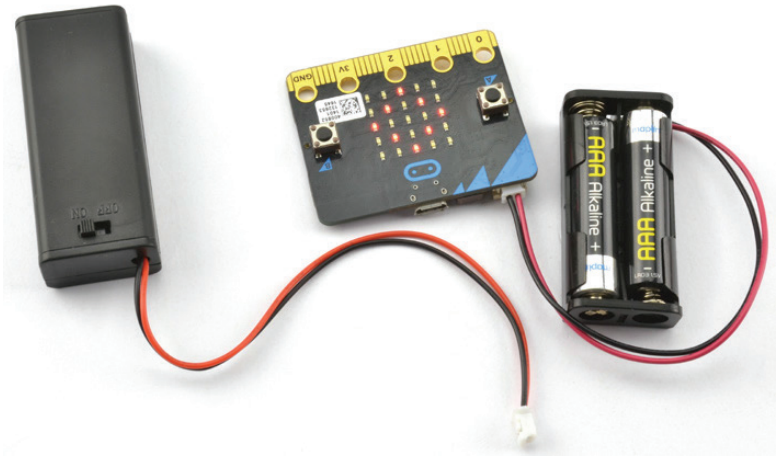


Рис. 1.3. Подключение блока батареек к micro:bit

Не используйте перезаряжаемые аккумуляторы AAA, потому что они, как правило, дают слишком низкое напряжение, недостаточное для питания micro:bit. Кроме того, эти аккумуляторы могут порождать опасно высокие токи при случайном коротком замыкании в цепи. Если вы все же решите использовать перезаряжаемые аккумуляторы, то выбирайте специализированный внешний аккумулятор с разъемом USB, подобный изображенному на рис. 1.4.

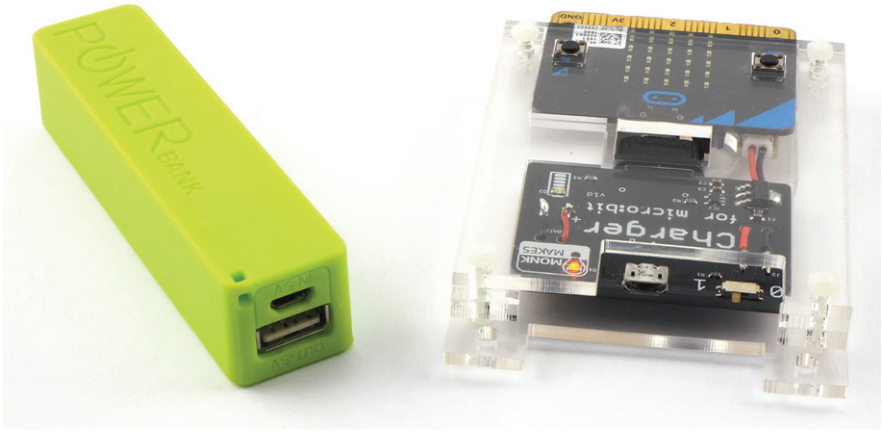


Рис. 1.4. Использование внешнего аккумулятора для питания micro:bit

Такие аккумуляторы можно подключать к micro:bit через разъем микро-USB. Для питания micro:bit вполне можно использовать недорогие внешние аккумуляторы небольшой емкости. Устройство micro:bit потребляет так мало электроэнергии, что более дорогие и интеллектуальные внешние аккумуляторы могут автоматически отключиться, посчитав, что к ним ничего не подключено и батарея находится на хранении.

Справа на рис. 1.4 – это блок питания Monk Makes Charger для micro:bit. В нем используется литий-полимерный (LiPo) аккумулятор, который автоматически заряжается при подключении к компьютеру кабелем USB. Если у вас есть такой блок питания, то после отключения USB-кабеля micro:bit будет питаться от этого аккумулятора.

В приложении (в конце книги) вы найдете еще несколько способов питания вашего устройства micro:bit.

ВНИМАНИЕ! *Самые ранние версии micro:bit могут выйти из строя при попытке подключить внешний аккумулятор к разъему микро-USB. Если у вас такое устройство, отличное от устройств версий 1.3B и 1.5, то избегайте любых способов подачи питания на плату, кроме подключения USB-кабелем к компьютеру или аккумуляторной батарее с напряжением 3 В.*

На платах micro:bit ранних версий не было номера версии. Переверните micro:bit и посмотрите справа внизу рядом с разъемом 0. *Если там написано* V1.3B *или* V1.5*, то можно использовать внешние аккумуляторы USB и блоки питания. Если номер версии отсутствует, не используйте эти блоки питания.*

В любом случае держитесь подальше от мощных источников питания и USB-аккумуляторов.

Исчерпывающее руководство по безопасности при работе с micro:bit можно найти по адресу: https://microbit.org/guide/safety-advice/.

Подключение электронных устройств к входным и выходным контактам

Одна из замечательных особенностей платы micro:bit – возможность подключения к ее контактам дополнительных элект-

ронных устройств. В данной книге мы будем использовать эти контакты для управления электромоторами, освещением и динамиком, а также для получения сигналов с датчиков, измеряющих освещенность, громкость звука и температуру.

Контакты 3V и GND предназначены для подачи питания. Контакты 0, 1 и 2 обычно называют *входами/выходами*, и мы будем подключать к ним электронные устройства.

ПРИМЕЧАНИЕ *Иногда контакты называют ножками, хотя они совсем не похожи на ножки. Термин «ножка» происходит от микросхем, установленных на плате micro:bit. Микросхемы имеют ножки, соединяющие их с платой, и каждый из этих трех контактов на плате соединен со своей ножкой своей микросхемы.*

Контакты 0, 1 и 2 могут использоваться для:

- вывода дискретных (цифровых) сигналов, например для включения и выключения внешнего светодиода;
- вывода аналоговых сигналов, например для управления яркостью свечения светодиода;
- генерации импульсов, например для управления сервомотором;
- ввода дискретных сигналов, например для определения факта нажатия внешней кнопки;
- ввода аналоговых сигналов, например для измерения температуры с помощью аналогового датчика;
- сенсорного ввода, например для определения момента прикосновения к контакту или проводу, соединенному с контактом.

Ввод дискретных и аналоговых сигналов

Когда контакт используется для вывода дискретного сигнала, в программах можно писать команды включения напряжения (3 В) на контакте и выключения (0 В). Причем напряжение можно только включить или выключить – на контакт нельзя подать никакое другое напряжение между 0 В и 3 В. То же относится к случаям, когда контакты используются для ввода

дискретных сигналов: сигнал может иметь только два состояния – включен и выключен. Если на контакт, используемый для ввода дискретных сигналов, подать напряжение, которое ближе к 3 В, чем к 0 В, то программа, прочитав состояние этого контакта, будет считать сигнал включенным, в противном случае – выключенным.

Аналоговые сигналы могут иметь множество промежуточных состояний между «включено» и «выключено». Аналоговые входы на micro:bit могут давать любые значения в диапазоне от 0 до 1023, в зависимости от уровня напряжения на контакте.

Вывод аналогового сигнала: широтно-импульсная модуляция

Micro:bit, как и все остальные цифровые электронные устройства, может работать только с дискретными цифровыми сигналами, имеющими два состояния – «включено» и «выключено». Чтобы вывести аналоговый сигнал с напряжением от 0 до 3 В, электронные устройства *имитируют* его, генерируя последовательности коротких дискретных импульсов. Чем длиннее импульс, тем больше мощности отдается устройству, подключенному к аналоговому выходу. Этот способ генерации аналоговых сигналов называют *широтно-импульсной модуляцией*, или *ШИМ*.

На рис. 1.5 показан принцип действия ШИМ.

Если к контакту подключен светодиод и длительность импульса с напряжением 3 В составляет всего 5 % от общего времени, то светодиод будет светиться очень тускло. Напротив, если длительность импульса с напряжением 3 В составит 90 % от общего времени, то светодиод будет светиться почти с максимальной яркостью.

Самое интересное, что в действительности в обоих случаях светодиод будет мигать 50 раз в секунду, но человеческий глаз не различает вспышки, следующие так быстро, и просто видит тусклый или яркий свет.

Если трех контактов 0, 1 и 2 окажется недостаточно для вашего проекта, то можно использовать дополнительный адаптер для подключения к другим контактам, находящимся между этими тремя пронумерованными контактами (см. рис. 1.1).

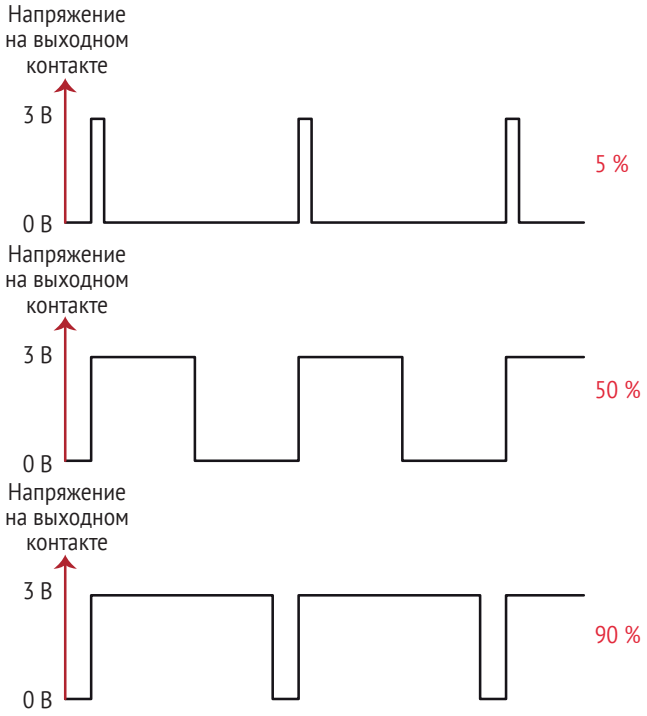


Рис. 1.5. Аналоговые выходы на micro:bit позволяют выводить разную мощность

Встроенные периферийные устройства

Надписи на обратной стороне платы micro:bit подсказывают нам о некоторых интересных возможностях micro:bit. Слева внизу (см. рис. 1.2) на плате можно видеть две надписи: *compass* (компас) и *accelerometer* (акселерометр).

В действительности компас – это *магнитометр*, то есть он измеряет силу магнитного поля. Его можно использовать как компас, а также для обнаружения магнитов.

Акселерометр измеряет силы, действующие на микросхему акселерометра. Поскольку гравитация – это постоянная сила, притягивающая все вниз, с помощью акселерометра можно, измеряя действующие силы и их направление, определить, когда и с каким ускорением переместилась плата micro:bit, а также когда она находится в свободном падении или ударяется обо что-то.

Также на обратной стороне можно увидеть надпись BLE Antenna. На плате micro:bit имеется оборудование BLE (Bluetooth

Low Energy – Bluetooth с низким энергопотреблением), которое позволяет micro:bit обмениваться данными по беспроводной сети с другими micro:bit или телефонами с поддержкой Bluetooth.

Обратите внимание, что технология обмена данными между двумя устройствами micro:bit на самом деле не является технологией Bluetooth; просто она использует ту же частоту. Подробнее об этой особенности micro:bit рассказывается в главе 10.

ОСНОВНОЕ ОБОРУДОВАНИЕ

Почти во всех экспериментах и проектах, описанных в этой книге, вам понадобится несколько вещей:

- **плата micro:bit**;
- **кабель с разъемами микро-USB и USB** для подключения micro:bit к компьютеру. Это должен быть стандартный кабель для передачи данных, а не кабель для зарядки, в котором отсутствуют соединения, необходимые для передачи данных. Кабель, предназначенный для зарядки, нельзя использовать для загрузки программ в micro:bit;
- **провода с зажимами типа «крокодил».** В идеале провода должны быть не длиннее 10–15 сантиметров, чтобы не запутаться в них;
- **батарейный блок 3V AAA** с двумя батарейками AAA;
- **блок питания USB.** Понадобится только в некоторых проектах.

В начале описания каждого проекта или эксперимента я буду приводить список необходимых предметов, а в приложении (в конце книги) вы найдете дополнительную информацию о том, где их можно приобрести.

В этой книге я постарался максимально упростить конструкции проектов, и, за исключением робота-вездехода из главы 6, вам не придется ничего паять. Для большинства проектов вам понадобятся только провода с зажимами «крокодил» для соединения устройств. При подключении с помощью зажимов «крокодил» лучше всего располагать зажим вертикально, как

показано на рис. 1.6, потому что при этом уменьшается вероятность случайно отсоединить провод.

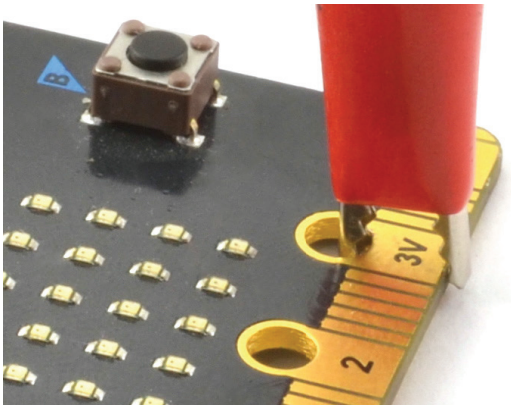


Рис. 1.6. Безопасное подключение зажима «крокодил»

ПРОГРАММИРОВАНИЕ MICRO:BIT

Неугомонный ученый не отличается терпением, поэтому давайте прямо сейчас заставим наше устройство micro:bit что-нибудь сделать. Для этого сначала запрограммируем micro:bit.

Одна из замечательных особенностей micro:bit заключается в том, что для начала работы с ним вам понадобится только USB-кабель и компьютер с браузером и подключением к интернету. Можно использовать компьютер под управлением Windows, macOS или Linux. Если на вашем компьютере установлен современный браузер, например Chrome, этого будет более чем достаточно.

Сначала подключим micro:bit к компьютеру. Затем напишем небольшую программу, используя два метода: перетаскивая мышью блоки кода в Blocks и вручную напечатав код на языке MicroPython.

Подключение micro:bit

Для начала подключите micro:bit к компьютеру с помощью USB-кабеля. Подойдет любой USB-кабель для передачи данных, но помните, что кабели, предназначенные только для за-

рядки, не имеют соединений, необходимых для передачи данных, и нам они не подойдут. Если у вас возникнут проблемы с программированием micro:bit, как описывается далее, попробуйте использовать другой USB-кабель.

После подключения micro:bit к компьютеру операционная система должна опознать его как USB-накопитель. Чтобы скопировать программу в micro:bit, найдите устройство micro:bit в своей файловой системе, которое должно выглядеть как флеш-накопитель или какое-либо другое съемное устройство. Затем скопируйте *шестнадцатеричный файл* в папку со значком micro:bit, и ваша программа автоматически будет установлена. Процесс загрузки программы в micro:bit также называют *прошивкой*.

А теперь создадим шестнадцатеричный файл и запишем его в micro:bit.

Программирование с помощью Blocks: Hello World

Программы для micro:bit можно создавать на веб-сайте, не устанавливая никакого программного обеспечения. Мы создадим программу, которая будет показывать на светодиодном дисплее бегущую строку с текстом. Запустите браузер и откройте адрес *https://makecode.microbit.org*. Щелкните на значке New Project (Новый проект), в открывшемся диалоге введите имя проекта «Hello World», щелкните на кнопке Create (Создать), и после этого должна появиться страница, как показано на рис. 1.7.

Это редактор, в котором создаются программы. Слева находится изображение виртуальной платы micro:bit, которая будет выполнять написанные вами программы.

Раздел посередине – это список категорий, таких как Basic (Основное), Input (Ввод) и Music (Музыка). В каждой из этих категорий имеются *блоки*, которые можно перетаскивать мышью в рабочую область справа. Каждый блок – это инструкция для micro:bit. Перетаскивая блоки и соединяя их, можно написать программу на языке Blocks.

Когда вы откроете редактор, то увидите, что в рабочем разделе справа (в разделе редактирования) уже имеются два блока: on start (при начале) и forever (постоянно). Любые блоки внутри

блока on start (при начале) будут выполняться один раз при первом включении micro:bit, при загрузке новой программы или после нажатия кнопки сброса на micro:bit. Любые блоки внутри блока forever (постоянно) будут запускаться снова и снова, пока вы не остановите программу.

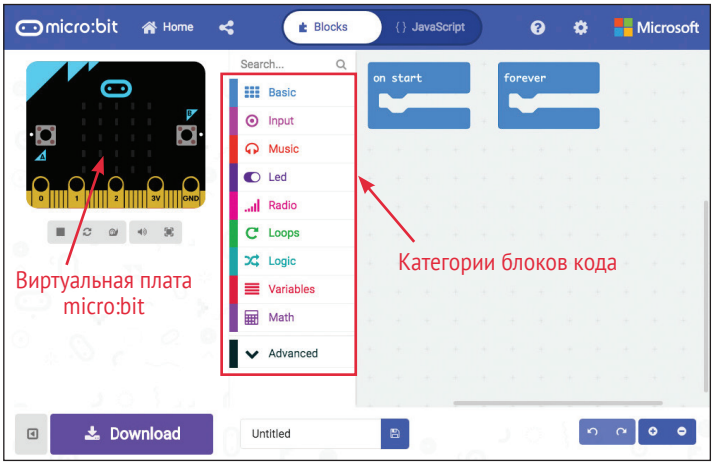


Рис. 1.7. Веб-страница https://makecode.microbit.org

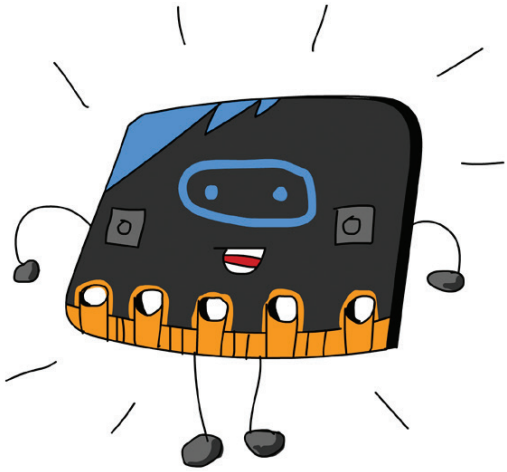
В нашей первой программе нам не понадобится блок forever (постоянно), поэтому щелкните на нем правой кнопкой мыши и в открывшемся контекстном меню выберите пункт Delete block (Удалить блок), чтобы удалить этот блок из программы. Затем добавьте в программу блок show string (показать строку) – *строкой* в языках программирования называется *текст*. Для этого щелкните на категории Basic (Основное), перетащите блок show string (показать строку) в область программирования и поместите его в блок on start (при начале), как показано на рис. 1.8. Если на вашем компьютере включен звук, то вы услышите приятный щелчок в момент соединения блоков.

Рис. 1.8. Блоки кода, отображающие бегущую строку с текстом «Hello World»

Теперь щелкните внутри пузыря с текстом и введите `Hello World`. Если хотите, можете ввести любой другой текст.

Как только блок show string (показать строку) окажется на месте, виртуальная плата micro:bit слева должна показать бегущую строку с вашим текстом, чтобы вы могли видеть, что делает ваша программа.

Теперь перенесем программу на настоящую плату micro:bit. Подключите micro:bit с помощью USB-кабеля и щелкните на кнопке Download (Скачать) слева внизу на веб-странице.



Файл программы загрузится из редактора точно так же, как любой другой файл, который вы можете загрузить из интернета. Папка, куда будет загружен файл, зависит от вашей операционной системы и настроек браузера, но обычно эта папка называется *Downloads* (*Загрузки*). Найдите эту папку, откройте ее, и вы должны увидеть в ней файл с именем *microbit.hex*. Затем выберите этот файл и перетащите его в папку со значком micro:bit (рис. 1.9).

Как только вы отпустите кнопку мыши, должно начаться копирование файла в micro:bit, при этом должен начать мигать светодиод на обратной стороне micro:bit. Когда мигание прекратится, micro:bit автоматически запустит программу, и по светодиодному дисплею побежит бегущая строка с текстом. Если вы захотите повторить, нажмите кнопку сброса на обратной стороне micro:bit – и программа будет выполнена повторно.

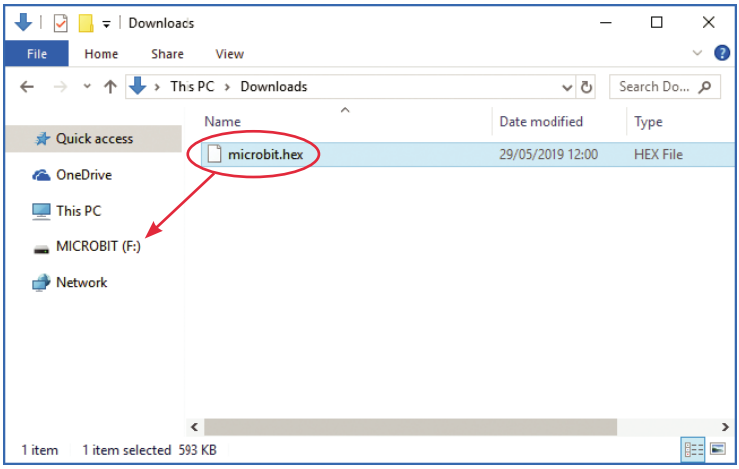


Рис. 1.9. Перетащите файл в папку со значком micro:bit

Скачивание непосредственно в micro:bit

Большинство браузеров позволяют выбрать папку для сохранения файлов перед их скачиванием. Вы можете использовать эту возможность, чтобы скачивать файлы прямо в micro:bit. Тогда вам не придется сначала скачивать файлы, а потом копировать их.

Чтобы выполнить необходимые настройки в браузере Chrome, введите в адресной строке *chrome://settings/*, на открывшейся странице выберите раздел Advanced (Дополнительные), прокрутите вниз до пункта Downloads (Скачанные файлы) и включите параметр Ask where to save each file before downloading (Всегда указывать место для скачивания). После этого, когда в следующий раз вы щелкнете на кнопке Download (Скачать) в редакторе Blocks, вам будет предложено указать место для сохранения файла, и вы сможете выбрать папку micro:bit.

Когда писались эти строки, разработчики micro:bit заканчивали работу над новой функцией, упрощающей прошивку программ для пользователей браузера Chrome. Прочитать об этой функции можно здесь: *https://support.microbit.org/support/solutions/articles/19000084059-beta-testing-web-usb*.

Добавляем графику

Чтобы отобразить сообщение, мы добавили внутрь блока on start (при начале) еще один блок – show string (показать строку). Блок on start (при начале) – это блок особого типа – *блок событий*, – который запускает связанный с ним код всякий раз, когда возникает определенное событие. В данном случае таким событием является запуск программы.

Давайте немного усложним программу, добавив новое событие, которое будет возникать при нажатии кнопки A. Для этого щелкните на категории Input (Ввод) и перетащите в область редактирования блок on button A pressed (кнопка A нажата). Затем перетащите блок show leds (показать светодиоды) из категории Basic (Основное) в блок on button A pressed (кнопка A нажата). Квадратики в блоке show leds (показать светодиоды) представляют светодиоды на плате. Вы можете выбрать светодиоды, которые должны загореться, щелкая мышью на квадратиках. Результат должен выглядеть примерно так, как показано на рис. 1.10.

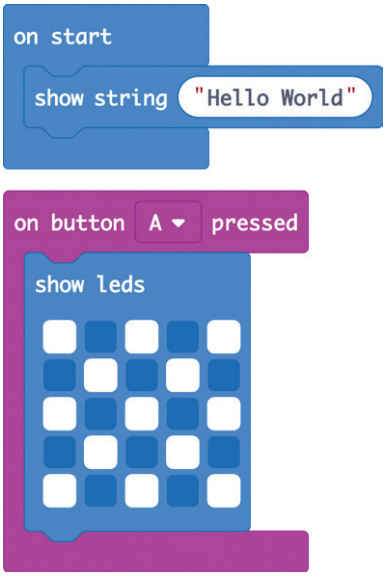


Рис. 1.10. Добавление в программу отображения узора на светодиодном дисплее

В блоке show icon (показать значок) имеется несколько готовых светодиодных изображений, и вы можете выбрать любое

из них. Снова щелкните на кнопке Download (Скачать) и скопируйте новый шестнадцатеричный файл в micro:bit. После загрузки новой программы вы сможете протестировать ее: нажмите кнопку A – и на светодиодном дисплее micro:bit должны загореться светодиоды, выбранные в блоке show leds (показать светодиоды), как показано на рис. 1.11.

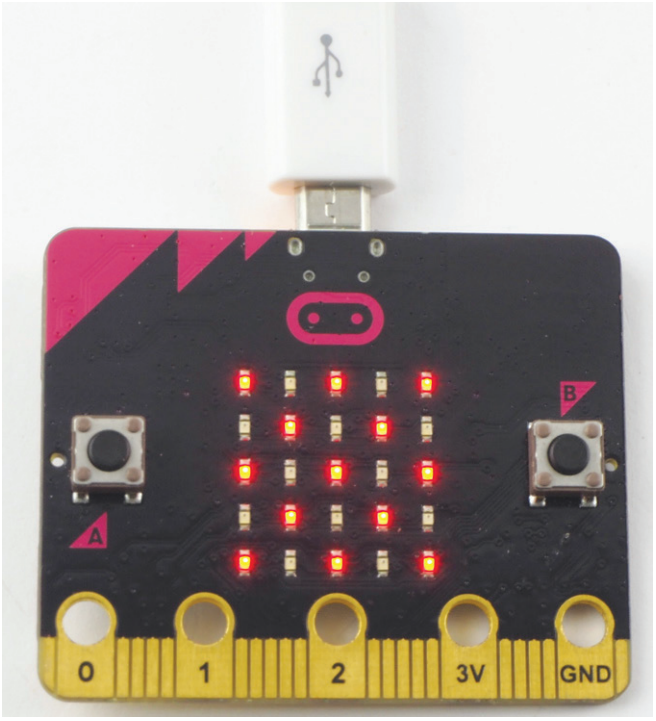


Рис. 1.11. Так выглядит узор из горящих светодиодов на настоящей плате micro:bit

Сохранение и публикация программ

Сайт *https://makecode.microbit.org* запоминает все ваши проекты. Выберите подходящее название для своего проекта, введите его в поле рядом с кнопкой Download (Скачать) – и ваш проект будет сохранен с этим именем. При каждом изменении программы она будет автоматически сохраняться, но для уверенности вы можете сохранить ее вручную, щелкнув на значке с изображением дискеты справа от поля с именем проекта. Обратите внимание, что после щелчка на этой кнопке вам будет предложено скачать шестнадцатеричный файл, но вы можете

отказаться от этого, щелкнув в открывшемся диалоге на кнопке Cancel (Отмена).

Учтите, что сохраненные программы на самом деле находятся в кеше вашего браузера, поэтому если вы очистите кеш, то потеряете свои программы.

Чтобы переключиться на другую программу или приступить к созданию новой, щелкните на кнопке Home (Главная) в верхней части страницы.

Чтобы опубликовать программу и сделать ее доступной для других, щелкните на кнопке Share (Поделиться), которая находится правее кнопки Home (Главная). После этого вам будет предложено подтвердить свое желание опубликовать проект. Щелкните на кнопке Share (Публикация проекта) – и перед вами появится ссылка, как показано на рис. 1.12.

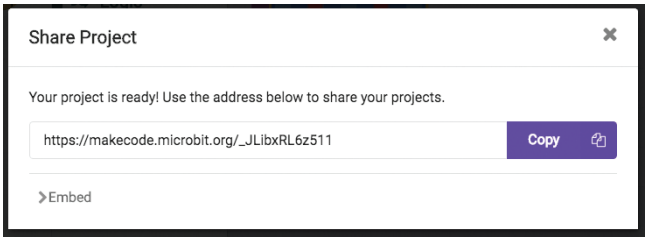


Рис. 1.12. Публикация проекта

Вы можете передать эту ссылку кому угодно. Когда кто-то введет ее в свой браузер, он получит копию вашей программы и сможет использовать ее. Обратите внимание: поскольку другим передаются лишь копии программ, они не смогут испортить вашу исходную программу.

Поиск блоков

Отыскать блоки show string (показать строку) и show leds (показать светодиоды), которые мы использовали, было несложно, но редактор Blocks содержит множество блоков и скрывает те, которые используются редко. Поэтому вы не увидите ошеломляющего множества вариантов в начале своего приключения с micro:bit.

Есть два места, в которых скрываются блоки. Во-первых, вы, возможно, заметили, что при выборе одной из категорий бло-

ков, например Basic (Основное), чуть ниже названия категории появляется значок с изображением троеточия и надписью more (еще; см. рис. 1.13).

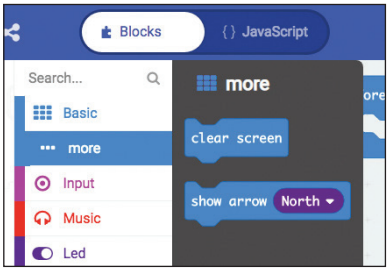


Рис. 1.13. Дополнительные блоки в подразделе more (еще)

Раздел more (еще) в категории Basic (Основное) содержит такие блоки, как clear screen (очистить экран) и show arrow (показать стрелку). Если попробовать выбрать другие категории, то можно заметить, что почти все они имеют раздел more (еще) и в некоторых из них довольно много дополнительных блоков.

Между прочим, если навести указатель мыши на блок и немного подождать, то откроется всплывающая подсказка, кратко описывающая, что делает блок. Потратив немного времени на знакомство с различными блоками, вы получите некоторое представление о том, что можно делать с micro:bit.

Второе место, где можно найти скрытые блоки, находится в категории Advanced (Расширенные), находящейся сразу под категорией Math (Математика; рис. 1.14). После щелчка на этой категории откроется список с множеством дополнительных категорий, содержащих более сложные конструкции языка Blocks, начиная с Functions (Функции). И снова потратьте немного времени на знакомство с блоками. Не волнуйтесь, если вам не удастся с первого раза понять назначение каких-то из них. Если вам нужен какой-то конкретный блок, воспользуйтесь полем Search (Поиск), чтобы найти его.

Из раздела Advanced (Расширенные) вы чаще всего будете использовать категории Text (Строки) и Pins (Контакты). В некоторых главах мы также используем блоки из категорий Functions (Функции) и Arrays (Массивы).

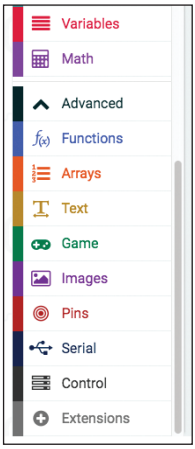


Рис. 1.14. Категория Advanced (Расширенные)

Программирование на MicroPython: Hello World

Blocks хорошо подходит для тех, кто только начинает осваивать программирование, потому что позволяет воплощать действительно впечатляющие проекты, перетащив мышью всего несколько блоков. Однако многие предпочитают писать программы в текстовом виде, а не перетаскивать блоки. Кроме того, ручной набор текстовых строк больше похож на настоящее программирование.

MicroPython – это версия языка программирования Python 3, которая включает все необходимое для программирования micro:bit. Фактически многие блоки в Blocks имеют прямые эквиваленты в языке MicroPython, поэтому переключиться с Blocks на MicroPython довольно просто. Python – популярный язык программирования для начинающих, и поэтому его часто преподают в школах.

Загрузка редактора для MicroPython

Мы будем использовать редактор *Mu* (*https://codewith.mu*), который имеет множество функций. Его можно скачать сразу на свой компьютер. Mu также позволяет прошить программу прямо в micro:bit без необходимости перетаскивать шестнад-

цатеричный файл мышью. Самое замечательное в Mu заключается в том, что он не требует доступа в интернет, чтобы прошить программу после того, как вы скачаете его к себе на компьютер.

Загрузите Mu с сайта *https://codewith.mu/#download*. На сайте доступно несколько версий Mu, поэтому выберите ту, которая подходит для вашего компьютера. На странице загрузки вам будут предложены разные версии для разных операционных систем. Если вы пользуетесь Windows, загрузите версию с пометкой «64-bit» для Windows Installer (рис. 1.15). Если вы пользуетесь macOS, то вам будет доступна только одна версия.

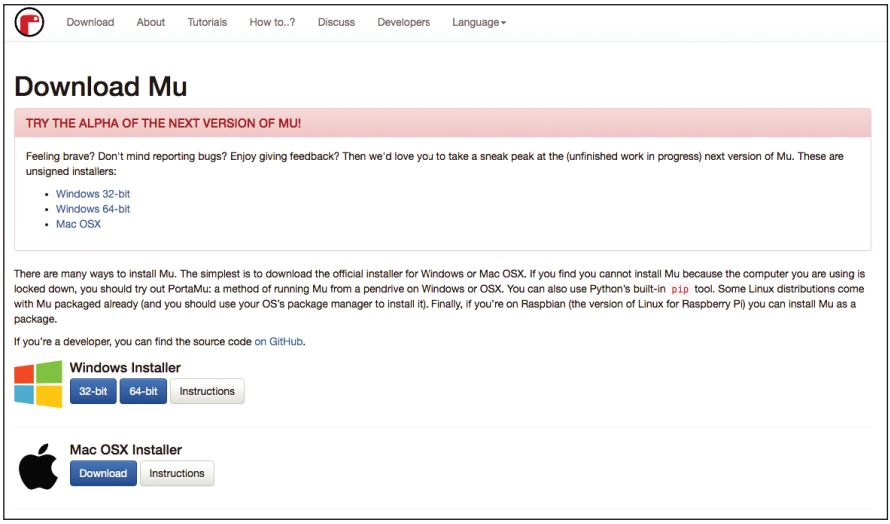


Рис. 1.15. Страница для загрузки Mu

Запустите установку и согласитесь с лицензионным соглашением. Оставьте остальные параметры как есть.

После первого запуска редактор Mu предложит вам выбрать *режим*. Выберите режим BBC micro:bit, как показано на рис. 1.16.

После этого Mu откроет пустое окно редактора, куда вы сможете ввести свою первую программу.

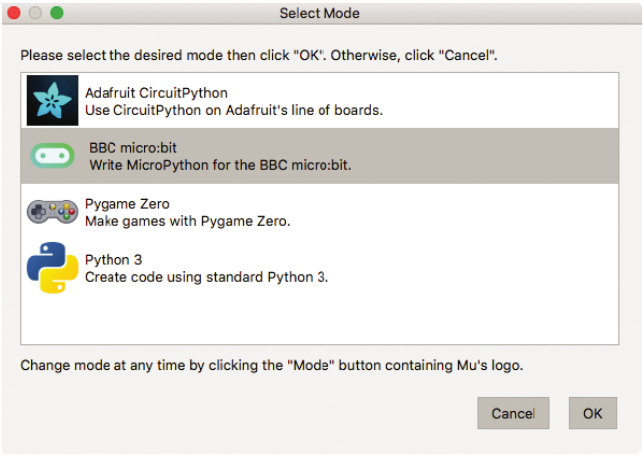


Рис. 1.16. Выбор режима в редакторе Mu

Ввод программы

Итак, приступим! Введите следующие строки в окне редактора Mu:

```
from microbit import *
display.scroll("Hello World")
```

Окно должно выглядеть, как показано на рис. 1.17.

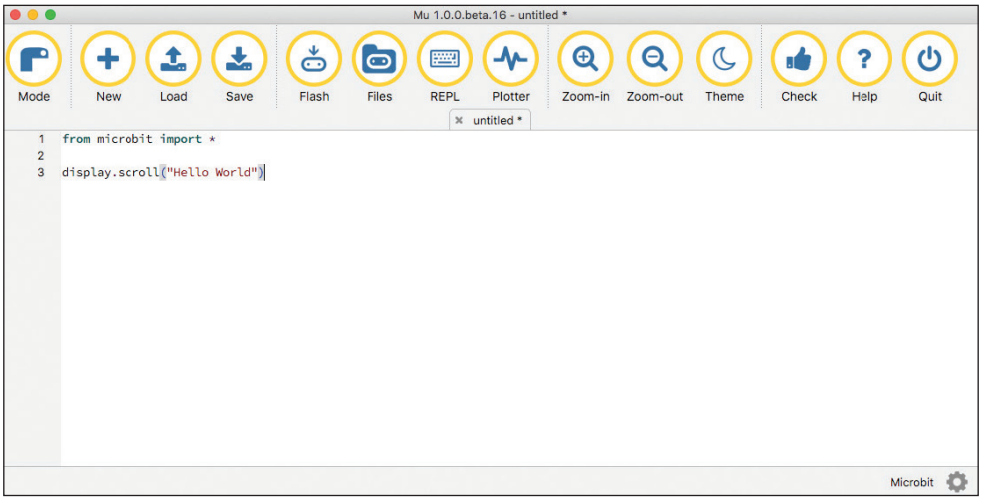


Рис. 1.17. Ввод программы на MicroPython в редакторе Mu

Сохраните программу, щелкнув на кнопке Save (Сохранить) в панели инструментов в верхней части окна Mu. В ответ вам будет предложено указать имя программы. Введите имя *hello.py*.

Теперь подключите свое устройство micro:bit к компьютеру, если оно еще не подключено, и щелкните на кнопке Flash (Прошить) в панели инструментов в окне Mu. После этого должен запуститься процесс загрузки шестнадцатеричного файла в micro:bit, как если бы вы перетащили файл, созданный редактором Blocks. Как только светодиод на обратной стороне micro:bit перестанет мигать, по светодиодному дисплею должно пробежать сообщение *Hello World*.

Давайте подробнее рассмотрим программу. Вот первая строка:

```
from microbit import *
```

С этой строки начинается почти каждая программа на MicroPython для micro:bit, потому что это она сообщает интерпретатору MicroPython, что тот должен подключить встроенную библиотеку, необходимую для работы со светодиодным дисплеем micro:bit и другим оборудованием. Эта строка не добавляется автоматически, поскольку MicroPython можно использовать с множеством других плат, а не только с micro:bit. На самом деле эта строка означает следующее: *из библиотеки* `mirobit` *импортировать все* (символ * означает «все»). Тот же эффект можно получить с помощью инструкции `import microbit`, но тогда вам придется перед всеми операциями, выполняемыми этой библиотекой, добавлять префикс `microbit`, а это лишний ручной ввод.

Вторая строка в нашей маленькой программе выводит сообщение на дисплей:

```
display.scroll("Hello World")
```

Команда `display.scroll` сообщает плате micro:bit, что та должна вывести (`display`) бегущую строку (`scroll`) на светодиодный дисплей. За командой следуют круглые скобки с текстом, за-

ключенным в кавычки. Круглые скобки используются, чтобы добавить дополнительную информацию для команды. В данном случае дополнительная информация (ее также называют *аргументом*) – это текст, который нужно вывести на дисплей. Текст также необходимо заключить в двойные кавычки, чтобы показать, что программа должна рассматривать его как простой текст, а не как дополнительные команды программирования.

Если после прошивки программы вы видите на дисплее micro:bit какой-то другой текст, не похожий на *Hello World* (или вообще ничего не увидите), вероятно, в вашу программу вкралась ошибка. При использовании текстового языка программирования необходимо соблюдать абсолютную точность при вводе команд. Например, если ошибиться в написании слова `display` или `scroll`, это вызовет ошибку при запуске программы. Ошибки в программах называют *жучками* (или *багами,* от англ. «bug» – «жучок»). Ошибки проявляются, только когда micro:bit попытается выполнить программу. Столкнувшись с ошибкой, внимательно сравните написанную вами программу с программой в книге, чтобы убедиться, что они в точности совпадают. Однако есть еще один способ поиска ошибок – REPL.

REPL

REPL (Read-Eval-Print-Loop – прочитать–выполнить–вывести–повторить) – это *интерфейс командной строки* для micro:bit – этот инструмент позволяет отправлять команды на языке MicroPython непосредственно в плату micro:bit, без необходимости оформлять их в виде программы. Если в окне этого инструмента ввести строку `1 + 2`, то в ответ он выведет `3`. Точно так же, если при попытке выполнить команду обнаружится ошибка, REPL прямо сообщит о ней.

Чтобы проверить, как это работает, давайте намеренно внесем ошибку в программу, записав команду `scroll` как `scrol` (удалив одну букву `l` в конце). Теперь снова щелкните на кнопке Flash (Прошить). Программа благополучно загрузится, даже притом что она содержит ошибку, но на этот раз на дисплее micro:bit побежит строка: `AttributeError: 'MicroBitDisplay' object`

has no attribute 'scrol' (AttributeError: у объекта 'MicroBitDisplay' нет атрибута 'scrol'). Прокрутка этого сообщения занимает много времени, поэтому, чтобы получить более четкое представление, щелкните на кнопке REPL в верхней панели инструментов Mu, а затем нажмите кнопку сброса на обратной стороне платы micro:bit. После этого в окне REPL должен появиться полный текст с сообщением об ошибке (внизу на рис. 1.18).

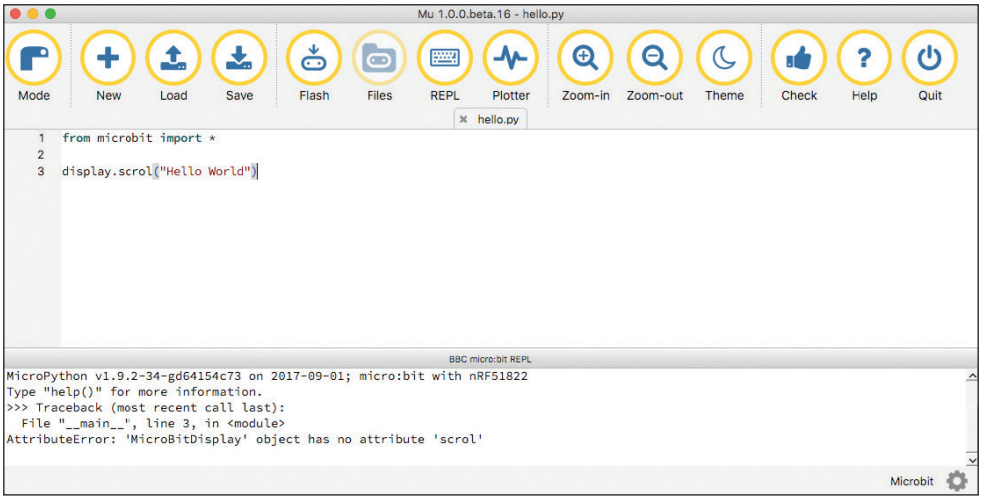


Рис. 1.18. Вывод сообщений об ошибках в инструменте REPL в редакторе Mu

Три угловые скобки >>> в окне REPL – это приглашение к вводу, то есть после этого приглашения можно ввести команду на языке Python, нажать клавишу Enter – и эта команда будет немедленно выполнена в micro:bit. Попробуйте ввести после приглашения >>> следующую команду и нажмите Enter (обратите внимание, что на этот раз команда `scroll` не содержит ошибки):

```
display.scroll("The REPL is useful")
```

Плата micro:bit должна немедленно вывести на светодиодный дисплей бегущую строку с указанным в команде текстом. Вам не нужно повторно вводить команду `import`, потому что устройство micro:bit уже выполнило ее – как раз перед тем, как наткнуться на ошибочную команду `scrol`.

ПРИМЕЧАНИЕ *При вводе команд в программе Hello World вы могли заметить, что редактор Mu пытается помочь, угадывая набираемый текст. Например, если ввести слово* `display`*, а затем приостановиться, то появится список вариантов продолжения (*`clear`*,* `get_pixel`*,* `is_on`*,* `off`*,* `on`*,* `scroll`*,* `set_pixel` *и* `show`*). Вы можете выбрать нужный, чтобы не вводить его вручную.*

Добавление графики

Теперь добавим в программу немного графики, как мы делали это в Blocks. Нам снова понадобится определить момент нажатия кнопки A и затем отобразить графический узор. Сделать это на языке MicroPython немного сложнее, потому что в отличие от Blocks этот язык не имеет понятия события. Вместо этого нужно написать цикл, который повторяет содержащиеся в нем команды, пока не получит приказ остановиться. В нашей программе эти команды в цикле будут проверять нажатие кнопки A и, когда это произойдет, выполнят необходимое действие. Другими словами, вместо ожидания события нажатия кнопки A программа должна снова и снова проверять, нажата ли эта кнопка. Вот эта программа:

```
from microbit import *

display.scroll("Hello World")

while True:
    if button_a.was_pressed():
        display.show(Image.CHESSBOARD)
```

Строка `while True` отмечает начало цикла, который будет продолжать выполняться, пока что-то его не прервет. Этим «что-то» может быть выключение платы micro:bit, нажатие кнопки сброса или комбинации клавиш Ctrl-C в окне инструмента REPL. Строки внутри цикла должны иметь дополнительный отступ, по сравнению со строкой начала цикла. К счастью, Mu распознает начало цикла и услужливо добавляет отступ к следующей строке.

Первая строка внутри цикла – это условный оператор `if`. Он вызывает функцию `button_a.was_pressed`, чтобы проверить, была

ли нажата кнопка A с момента последней проверки вызовом `was_pressed`. Если да, то будут выполнены строки с отступом под оператором `if`. Обратите внимание, что следующая строка имеет еще больший отступ – это означает, что данная строка должна выполняться, только если команда в операторе `if` вернет значение `True` (то есть если кнопка действительно была нажата). Эта следующая строка сообщает, что устройство micro:bit должно показать на светодиодном дисплее указанный узор из библиотеки `Image`. Для этого примера я выбрал узор `CHESSBOARD` (шахматная доска). Более подробно о команде `if` мы поговорим ниже, в разделе «Условный оператор if».

В языке Python отступы очень важны, и необходимость строгого следования правилам оформления отступов может стать причиной множества ошибок при первом знакомстве с языком. Строки с отступами, например внутри команды `while` или `if`, должны иметь одинаковые отступы. В Mu используются отступы из четырех пробелов. По мере обретения опыта программирования на Python вы начнете оформлять отступы почти интуитивно, эта привычка превратится в вашу вторую натуру.

ОСНОВНЫЕ ПОНЯТИЯ В ПРОГРАММИРОВАНИИ

В этом разделе мы познакомимся с некоторыми ключевыми понятиями в программировании, которые необходимо понимать, чтобы уметь изменять программы, написанные другими программистами, или писать свои. Эти понятия не зависят от используемого языка – Blocks или MicroPython, – поэтому далее мы посмотрим, как каждое из понятий реализуется в обоих языках, сначала в Blocks, а потом в MicroPython.

Переменные

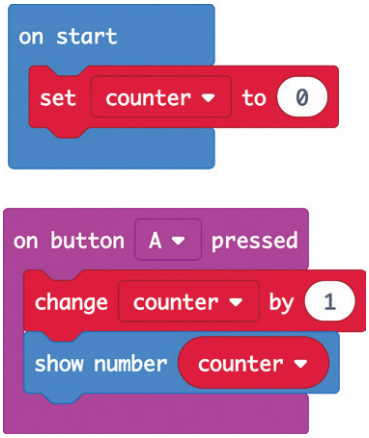
Переменная – это имя, связанное с одним или несколькими значениями. Чтобы использовать некоторое значение в программе, достаточно указать его имя. Проиллюстрируем эту идею на нескольких примерах.

Blocks

В числе основных категорий в редакторе Blocks имеется категория Variables (Переменные). Создайте новый проект, щелкнув на кнопке New Project (Новый проект) на главной странице редактора Blocks (*https://makecode.microbit.org*). Удалите блок forever (постоянно) – он нам не понадобится, – а затем в категории Variables (Переменные) выберите элемент Make a Variable... (Создать переменную...). В появившемся диалоге введите имя переменной counter. Перетащите блок set counter to 0 (задать для counter значение 0) в область программирования и вставьте его в блок on start (при начале).

Затем в категории Input (Ввод) найдите блок on button A pressed (кнопка A нажата) и перетащите его в область программирования. В категории Variables (Переменные) найдите блок change counter by 1 (изменить counter на 1) и перетащите его в блок on button A pressed (кнопка A нажата). Затем туда же перетащите блок show number (показать число) из категории Basic (Основное). Наконец, выберите блок counter в категории Variables (Переменные) и перетащите его в блок show number (показать число) на место числа 0, чтобы он заменил это число.

После всех этих действий программа в Blocks должна выглядеть, как показано ниже.



При желании можете опробовать эту программу на виртуальном устройстве micro:bit, изображенном слева от редактора, для этого щелкните на кнопке A. Вы должны увидеть, как

на дисплее появится число «1», и затем оно будет увеличиваться с каждым новым щелчком на A.

Давайте разберемся с тем, что происходит в программе. В блоке on start (при начале) создается переменная с именем counter, и ей присваивается начальное значение 0. Когда активируется блок on button A pressed (кнопка A нажата), он запускает сначала блок change counter by 1 (изменить counter на 1), который изменяет значение переменной counter, прибавляя 1, а затем show number (показать число), который выводит значение переменной на дисплей.

В данном случае наша переменная counter содержит число, но вообще переменные могут хранить текст и даже коллекции данных.

MicroPython

На языке MicroPython аналогичная программа выглядит так:

```
from microbit import *

counter = 0

while True:
    if button_a.was_pressed():
        counter += 1
        display.scroll(str(counter))
```

Первая строка, как обычно, импортирует библиотеку micro:bit. Следующая строка создает переменную counter и присваивает ей начальное значение 0. Затем объявляется цикл while, который проверяет факт нажатия кнопки A и добавляет 1 к переменной counter. Чтобы добавить 1, используется оператор +=, который эквивалентен блоку change counter (изменить counter) в Blocks.

Чтобы вывести новое значение на дисплей, нужно с помощью команды str(counter) преобразовать число в текстовую строку, потому что команда display умеет выводить только текстовые строки.

Обратите внимание, что использование одних команд внутри других – это нормальное явление в программировании на

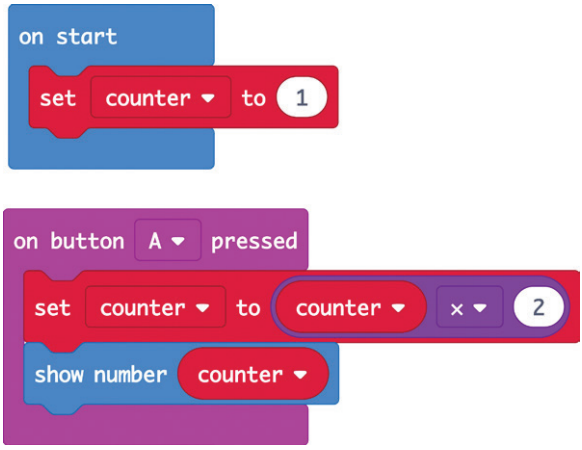
Python. Здесь мы использовали команду `str` внутри команды `display.scroll`, заключив ее в круглые скобки. В таких случаях первой выполняется самая внутренняя команда (в данном случае `str`), и ее результат передается следующей команде (в данном случае команде `display.scroll`).

Арифметика

В предыдущем примере вы увидели, как прибавить число к переменной. Кроме сложения с переменными, можно использовать все обычные арифметические операции, включая вычитание, умножение и деление.

Blocks

Допустим, мы решили удваивать число в переменной вместо увеличения его на единицу. Для этого нужно предыдущую программу изменить так, чтобы переменная `counter` инициализировалась начальным числом 1, а блок on button A pressed (кнопка A нажата) умножал `counter` на 2, как показано ниже.



Теперь вместо блока change counter by 1 (изменить counter на 1) нужно использовать блок set counter to (задать для counter значение), а внутри него число 0 заменить блоком умножения (×) из категории Math (Математика). На место первого операнда в блок умножения нужно поместить переменную `counter`, а во втором операнде ввести число 2. Теперь при каждом нажатии кнопки A значение переменной `counter` будет удваиваться.

В Blocks трудно манипулировать математическими операциями, потому что приходится вкладывать одни блоки внутрь других. Поэтому если программа должна выполнять много арифметических операций, то, вероятно, лучше написать ее на MicroPython.

MicroPython

В языке MicroPython арифметические операции выполняются с помощью символов +, -, * (умножение) и / (деление). Для изменения порядка выполнения операций также можно использовать круглые скобки, как это принято в математике. Вот как выглядит аналогичная программа удвоения на MicroPython:

```
from microbit import *

counter = 1

while True:
    if button_a.was_pressed():
        counter = counter * 2
        display.scroll(str(counter))
```

Ключевая строка здесь:

```
counter = counter * 2
```

Символ = после имени переменной означает: «результат выражения справа от символа = присвоить этой переменной». В данном случае используется выражение `counter * 2` (значение переменной `counter` умножается на 2). Также можно использовать более короткую форму записи `counter *= 2`, как мы сделали это для операции сложения в предыдущей программе.

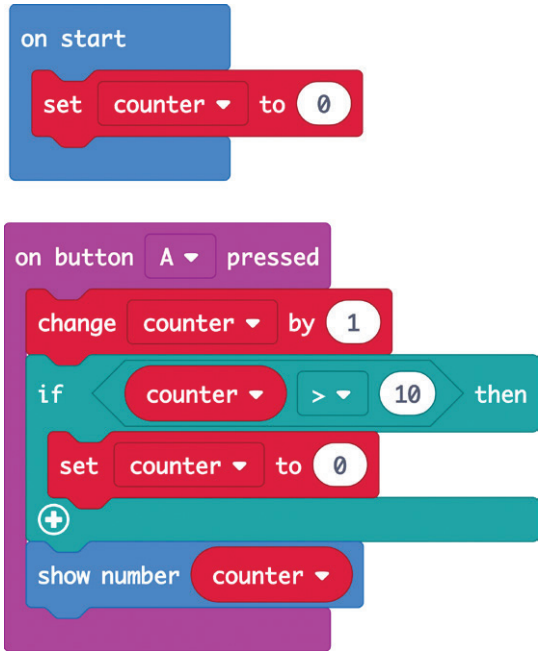
Условный оператор if

Программу можно рассматривать как последовательность шагов, которые выполняет micro:bit. Иногда требуется, чтобы программа принимала решения и выполняла определенные шаги на основе этих решений. Блоки в категории Logic (Логика) позволяют программам принимать решения.

Давайте перепишем наш пример, увеличивающий переменную counter на 1, так чтобы она продолжала счет до 10, а затем возвращалась к 0, чтобы начать счет сначала.

Blocks

Попробуйте сконструировать в Blocks следующую программу.



Сначала добавьте блок if (если) после блока change counter by 1 (изменить counter на 1). В блок if true then (если истина то) на место true (истина) подставьте блок сравнения, который сравнивает значение counter с числом 10. Если значение счетчика больше (>) 10, то программа запустит блоки внутри блока if (если). В данном случае блок if содержит единственный блок, присваивающий переменной counter число 0. Если счетчик не больше 10, то программа просто выведет число на дисплей.

Существует и другой вариант блока if (если), который позволяет выполнить одно действие, если условие истинно, и другое действие, если условие ложно. Больше об этом вы узнаете в следующих главах.

MicroPython

Мы уже использовали условный оператор `if` в MicroPython, когда проверяли, была ли нажата кнопка. Но теперь мы не просто проверяем истинность или ложность условия, а сравниваем переменную `counter` с числом 10. Вот как выглядит та же версия программы на MicroPython:

```
from microbit import *

counter = 0

while True:
    if button_a.was_pressed():
        counter += 1
        if counter > 10:
            counter = 0
        display.scroll(str(counter))
```

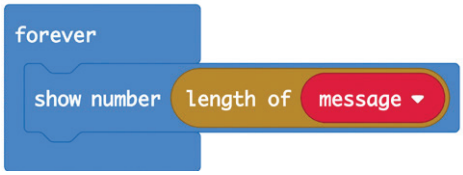
Особое внимание обратите на отступы в этой программе: здесь один условный оператор `if` вложен внутрь другого оператора `if`, который сам находится внутри цикла `while`. Проверьте себя – сможете ли вы понять, как работает этот код, опираясь на объяснение в описании версии для Blocks.

Строки

Как уже говорилось выше, строка – это последовательность символов (цифр, букв или знаков препинания), часто составляющих слова и предложения. Мы уже использовали строку в самой первой программе *Hello World* на рис. 1.8, чтобы вывести текст «Hello World».

Blocks

Большинство блоков со строками в Blocks находятся в категории Text (Строки). Самый простой из них – блок с парой кавычек и пробелом между ними, вместо которого можно ввести свой текст и получить строку. Этот блок можно использовать, чтобы запомнить строку в переменной. Следующая программа записывает строку в переменную и выводит ее длину.



Мы используем блок on start (при начале), внутри которого присваиваем переменной message некоторый текст. В блок show number (показать число) нужно добавить блок length of (длина) и внутрь этого блока вставить переменную message. Блок length of (длина) определяет количество символов в указанной строковой переменной message, которое затем отображается на дисплее блоком show number (показать число).

В категории Text (Строки) имеются также другие блоки, позволяющие выполнять такие действия, как объединение двух строк, вырезание части строки и преобразование строки в число.

MicroPython

Строки в MicroPython отличаются от команд тем, что заключены в двойные кавычки, в точности как в Blocks. Вот как выглядит предыдущая программа, написанная на MicroPython:

```
from microbit import *

message = "This text is in a variable."

while True:
    display.scroll(str(len(message)))
```

Она работает точно так же, как программа на Blocks. Обратите внимание, что здесь используется команда str для преобразования числа (то есть длины строки, которую возвращает len) в строку, чтобы ее можно было вывести на дисплей.

Массивы и списки

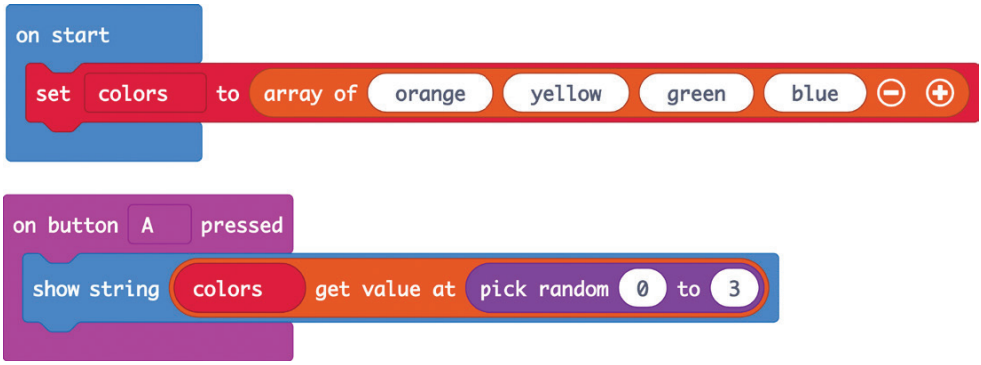
До сих пор мы использовали переменные, хранящие только один элемент данных – число или строку. Но иногда требуется сохранить в переменной сразу несколько значений и обращаться к ним по очереди. Примером такого набора значений может служить последовательность нот, составляющих мелодию (как будет показано в главе 2).

В Blocks такие коллекции значений называются *массивами*, а в MicroPython – *списками*. Однако суть от названия не меняется.

Blocks

В редакторе Blocks есть категория Arrays (Массивы), содержащая блоки для работы с массивами. Наиболее важными из них являются set list to (задать для список значение), set text list to (задать для список значение) и get value at (получить значение по индексу). Первые два позволяют создать список, содержащий числа или строки соответственно, а последний – получить значение определенного элемента в массиве.

Следующая небольшая программа создает переменную с именем `colors` и присваивает ей массив из четырех строк. Затем выбирает случайный элемент из этого массива и выводит его на дисплей. Обратите внимание, что во вложенном блоке array of (массив) есть кнопки + и –, которые позволяют настроить количество элементов в массиве при его создании.



Мы добавили блок on button A pressed (кнопка A нажата). Внутрь этого блока вставили блок show string (показать строку),

внутри которого вставили блок get value at (получить значение по индексу). Затем мы добавили блок pick random (выбрать случайно) из категории Math (Математика), чтобы получить случайное число от 0 до 3. Все вместе это означает следующее: когда программа обнаружит, что была нажата кнопка A, блок pick random (выбрать случайно) выберет случайное число от 0 до 3, блок get value at (получить значение по индексу) извлечет из массива строку с индексом, равным этому случайному числу, а затем блок show string (показать строку) выведет полученную строку на дисплей. Обратите внимание, что независимо от языка программирования – Blocks или MicroPython – нумерация элементов в массивах начинается с 0, а не с 1, поэтому если блок pick random (выбрать случайно) выберет число 0, то на дисплее появится первый элемент массива.

MicroPython

Вот как выглядит та же программа на языке MicroPython:

```
from microbit import *
import random

numbers = ["orange", "yellow", "green", "blue"]

while True:
    if button_a.was_pressed():
        display.scroll(numbers[random.randint(0, 3)])
```

Чтобы в MicroPython определить список, нужно напечатать квадратные скобки [и] и внутри этих скобок перечислить элементы списка через запятую. Квадратные скобки также используются для получения значения определенного элемента списка, но в этом случае внутри квадратных скобок указывается индекс – позиция элемента в списке. В данном примере в квадратные скобки заключена команда `random.randint`, которая возвращает случайное число от 0 до 3.

В заключение о программировании

На этом мы заканчиваем краткое знакомство с приемами программирования micro:bit. Весь код, который приводится в этой книге, можно скачать из интернета, поэтому вам не нужно осваивать программирование, чтобы начать экспериментировать и создавать свои проекты. В процессе чтения этой книги вы познакомитесь с новыми блоками и функциями языка MicroPython, работу которых я буду объяснять по мере их появления.

Дополнительную информацию о программировании на MicroPython для micro:bit можно найти на сайте *https://microbit-micropython.readthedocs.io*. Если вы только начинаете знакомиться с языком Python, то полезным дополнением к этой книге вам послужит другая моя книга: «Programming micro:bit: Getting Started with MicroPython» (McGraw-Hill, 2018).

СКАЧИВАНИЕ ПРОГРАММ

Некоторые программы в этой книге длинные и сложные, и, возможно, вам не захочется вводить их вручную. Если вы не хотите разрабатывать программы самостоятельно, то можете просто скачать их и прошить в свое устройство micro:bit.

Скачивание программ на языке Blocks

Все программы на языке Blocks доступны на сайте GitHub по адресу: *https://github.com/simonmonk/mbms/*. Когда вы выберете понравившийся вам проект и щелкнете на его ссылке, он откроется в вашем браузере.

Прокрутите страницу GitHub вниз, пока не увидите текст, похожий на изображение на рис. 1.19. Вам нужен список ссылок на все программы на языке Blocks.

Чтобы открыть выбранную программу, просто щелкните левой кнопкой мыши на соответствующей ссылке. Например, на рис. 1.20 показано, что получится, если щелкнуть на ссылке проекта «Musical Doorbell» (музыкальный дверной звонок).

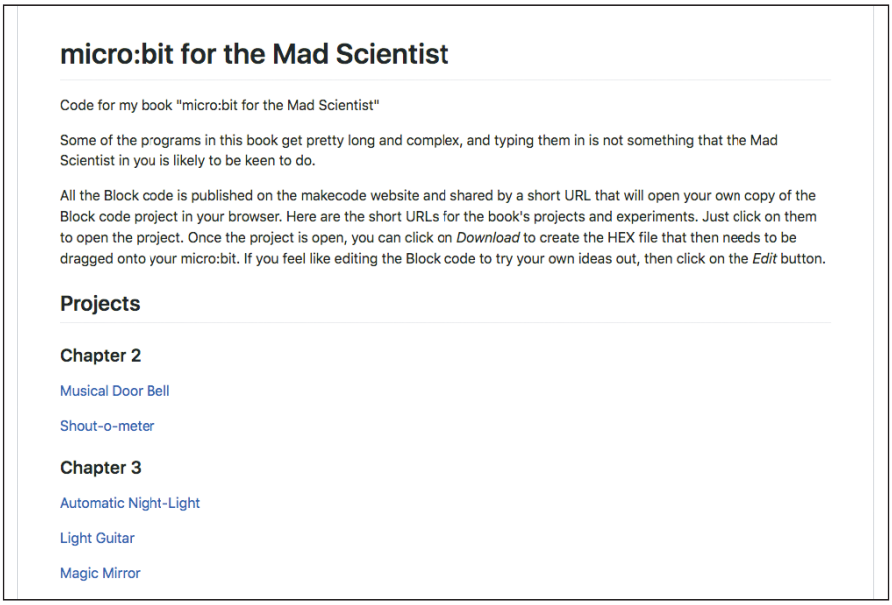


Рис. 1.19. Ссылки на все программы на языке Blocks из этой книги

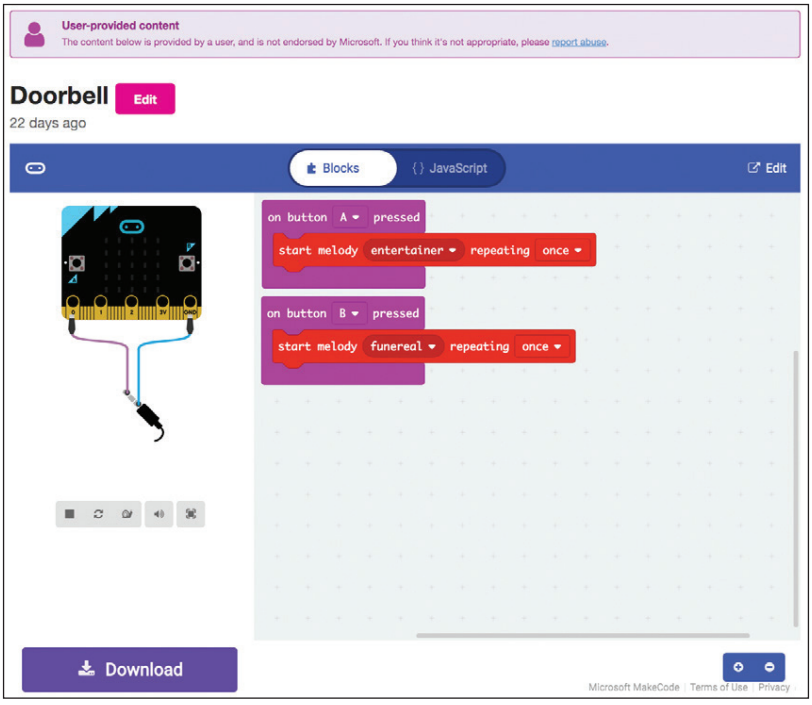


Рис. 1.20. После щелчка на ссылке «Musical Doorbell» откроется проект музыкального дверного звонка

Обратите внимание, что рис. 1.20 не похож на обычное окно редактора Blocks. Причина в том, что проекты открываются в режиме, в котором можно только просмотреть код и прошить его в micro:bit. Если у вас появится желание изменить программу или просто увидеть ее в более привычном редакторе, щелкните на кнопке Edit (Правка) в правом верхнем углу. После этого для вас будет создана отдельная копия программы, которая откроется в привычном редакторе и будет доступна для изменения.

Скачивание программ на языке MicroPython

Программы на MicroPython доступны на сайте GitHub по адресу: *https://github.com/simonmonk/mbms/*.

Если у вас есть опыт использования программного обеспечения Git, то можете клонировать весь репозиторий на свой компьютер. Для тех, кто не имеет такого опыта, ниже приводится пошаговое руководство, следуя которому, можно скачать все программы.

1. Откройте в браузере страницу с адресом *https://github.com/simonmonk/mbms/*, щелкните на зеленой кнопке Clone or download (Клонировать или скачать)[1] и в открывшемся меню выберите пункт Download ZIP (Скачать файл ZIP), как показано на рис. 1.21.

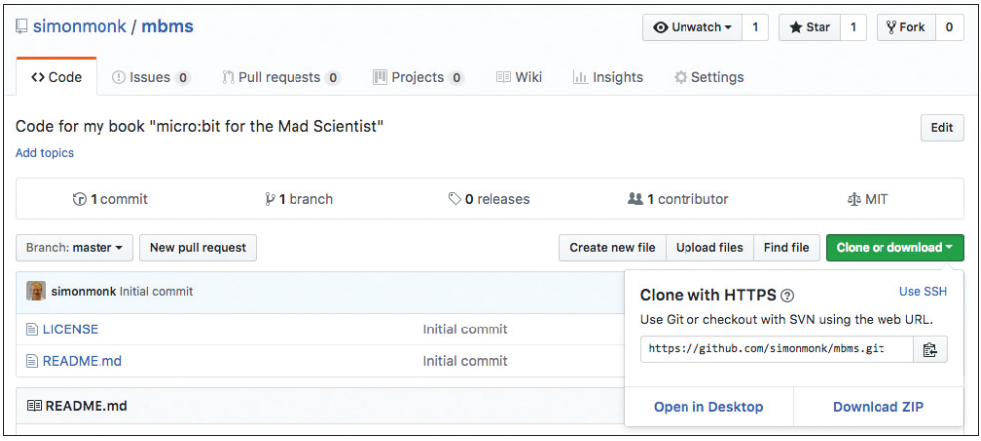


Рис. 1.21. Скачивание кода с программами из книги

[1] Пока эта книга готовилась к печати, разработчики сайта изменили надпись на кнопке, теперь на ней написано **Code** (Код). – *Прим. перев.*

2. Найдите файл ZIP в папке, куда вы его скачали (*mbms-master.zip*), и распакуйте его.

   Процесс распаковки зависит от вашей операционной системы – Windows, macOS или Linux. В macOS и в большинстве дистрибутивов Linux ZIP-файлы автоматически распаковываются при попытке открыть их. Если вы пользуетесь Windows, то имейте в виду, что Windows позволяет заглядывать внутрь ZIP-файлов, не распаковывая их, поэтому вы не сможете использовать файлы программ, находящиеся внутри ZIP-файла, пока не извлечете их. Чтобы извлечь файлы в Windows, щелкните правой кнопкой мыши на ZIP-файле в проводнике и в появившемся контекстном меню выберите пункт Extract All... (Извлечь все...), как показано на рис. 1.22.

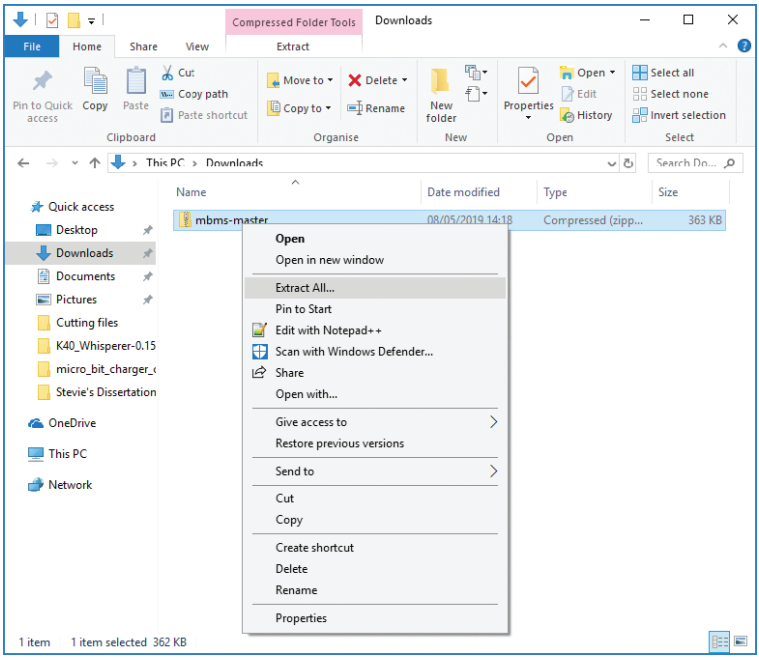


Рис. 1.22. Распаковывание ZIP-файла в Windows

Извлеченные файлы будут сохранены в папке *mbms-master*. Внутри этой папки вы найдете еще одну папку, с именем *python*, а внутри нее – программы на языке MicroPython в виде файлов с расширением *.py*.

3. К сожалению, эти программы нельзя открыть в редакторе Mu простым двойным щелчком. Чтобы открыть программу, сначала запустите редактор Mu, щелкните на кнопке Load (Загрузить), в открывшемся диалоге найдите нужную вам программу на MicroPython и щелкните на кнопке Open (Открыть). Чтобы сократить время на поиск нужной программы, я советую переместить все программы на MicroPython из папки *python*, которую вы только что скачали, в папку, где обычно Mu ожидает найти свои программы. По умолчанию это папка *mu_code* в вашей домашней папке. Теперь, когда вы щелкнете на кнопке Load (Загрузить) в редакторе Mu, вы сразу увидите все программы на MicroPython.

ИТОГИ

Итак, познакомившись с некоторыми основами micro:bit, можно приступать к экспериментам и работе над проектами. Мы начнем с создания и улавливания звуков с помощью micro:bit.