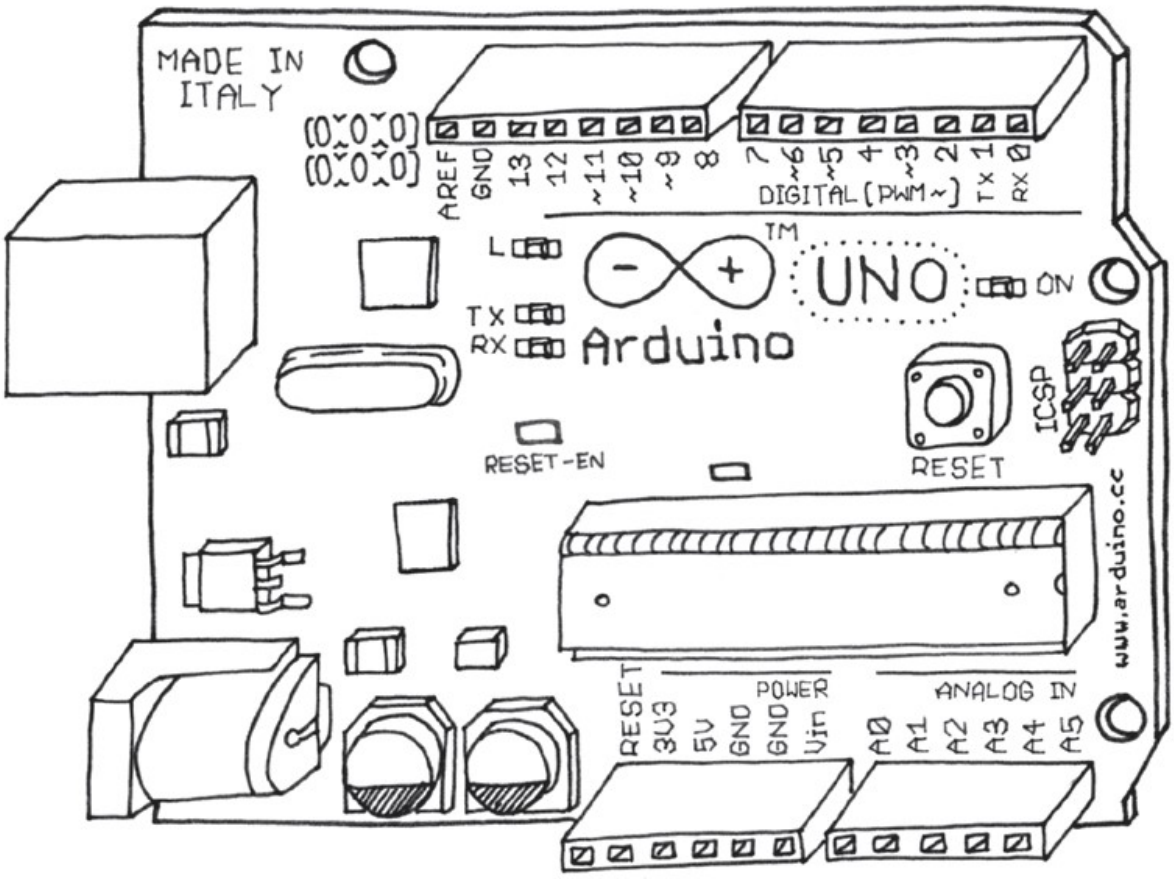
Make:

Wprowadzenie do Arduino

Wydanie II



Platforma prototypowania elektronicznego open source

Massimo Banzi współzałożyciel Arduino
i Michael Shiloh

Massimo Banzi

Michael Shiloh

Wprowadzenie do Arduino

Wydanie 2

Przekład: Maria Chaniewska
Marek Włodarz

APN Promise 2022

Wprowadzenie do Arduino, wydanie 2

Authorized Polish translation of English edition of **Getting Started with Arduino, 4th edition**, ISBN: 978-1-680-45693-6

APN PROMISE SA, ul. Domaniewska 44a, 02-672 Warszawa
tel. +48 22 35 51 600, fax +48 22 35 51 699
e-mail: *wydawnictwo@promise.pl*

Ważna informacja dla Czytelników: Czytelnicy sami odpowiadają za swoje bezpieczeństwo, w tym właściwe użycie sprzętu i elementów ochronnych, oraz decydują, czy mają odpowiednie umiejętności i doświadczenie. Elektryczność i elementy używane w tych projektach są niebezpieczne, o ile nie są właściwie używane i bez właściwych środków ostrożności. Niektóre ilustracje nie przedstawiają elementów zabezpieczających ani stosownych przyrządów, aby bardziej czytelnie przedstawić kroki projektu.
Te projekty nie są przeznaczone dla dzieci. Instrukcji i sugestii z książki Wprowadzenie do Arduino należy używać na własne ryzyko. Wydawnictwo O'Reilly Media, Inc. i autor nie ponoszą żadnej odpowiedzialności za żadne skutki, w tym zniszczenia, uszkodzenia zdrowia i koszty. Do Czytelników należy odpowiedzialność za zgodność ich działalności z odpowiednimi prawami, w tym prawami autorskimi.

ISBN: 978-83-7541-492-9 (druk), 978-83-7541-493-6 (ebook)

Ilustracje: Judy Aime' Castro

Przekład: Maria Chaniewska, Marek Włodarz
Korekta: Ewa Swędrowska
Skład i łamanie: MAWart Marek Włodarz

Spis treści

Przedmowa

Massimo i ja z radością przystąpiliśmy do wprowadzania licznych zmian pojawiających się w gwałtownie rozwijającym się świecie prototypowania elektronicznego do nowego wydania *Wprowadzenia do Arduino*.

Wydanie to zawiera szereg nowych rozdziałów. Rozdział 7 przedstawia chmurę Arduino, w tym IoT Cloud oraz Project Hub. Rozdział 8 to rozbudowany i praktyczny przykład systemu nawadniania ogrodu. Rozdział 9 przedstawia nową rodzinę płytek Arduino z 32-bitowym procesorem ARM, a rozdział 10 to nowy projekt wykorzystujacy sieć, oparty na Arduino ARM: Internet Fistbump.

Zostały wprowadzone również inne uaktualnienia:

- Wydanie to uwzględnia IDE w wersji 2.0.
- Instalowanie IDE jest dużo łatwiejsze i dołączone zostały instrukcje dla systemu Linux.
- Dodatek zawiera obecnie przegląd wszystkich rodzin Arduino, płytek z ich wymiarami oraz przewodnik ułatwiający wybór.
- Dokonaliśmy pewnych zmian w nomenklaturze:
  - Nazwy sygnałów SPI są obecnie zgodne z rezolucją Open Source Hardware dostępną pod adresem *oshwa.org/a-resolution-to-redefine-spi-signal-names/*
  - Rodzaje łączówek to obecnie albo pin, albo gniazdko.

W kolejnych wydaniach ilustracje się zmieniały i dodawanych było wiele nowych. Autorzy uznają wkład Elisy Canducci, która sporządziła ilustracje do pierwszego i drugiego wydania oraz Judy Aime' Castro, która zmodyfikowała niektóre istniejące ilustracje i dodała wiele nowych w wydaniu trzecim i bieżącym.

— Michael

Wprowadzenie

Parę lat temu stanąłem przed bardzo interesującym wyzwaniem: miałem nauczyć projektantów technicznych absolutnego minimum elektroniki, aby mogli budować interaktywne prototypy obiektów, które projektują.

Na początku zacząłem, kierując się podświadomym instynktem, uczyć elektroniki w taki sam sposób, w jaki byłem jej uczony w szkole. Później stwierdziłem, że nie przynosi to zamierzonych efektów, i zacząłem przypominać sobie, jak piekielnie znudzony siedziałem w klasie, zalewany całą tą teorią bez wzmianki o jakimkolwiek praktycznym zastosowaniu.

W rzeczywistości, gdy byłem w szkole, znałem już elektronikę w bardzo empiryczny sposób: niewiele teorii, ale mnóstwo przydatnych doświadczeń.

Zacząłem myśleć o tym, jak naprawdę uczyłem się elektroniki:

- Rozkładałem na części dowolne urządzenia elektroniczne, które wpadały mi w ręce.
- Powoli poznawałem wszystkie elementy.
- Zaczynałem z nimi majstrować, zmieniając pewne wewnętrzne połączenia i patrząc, co dzieje się z urządzeniem: zwykle coś między wybuchem a smugą dymu.
- Zaczynałem budować pewne zestawy z czasopism elektronicznych.
- Łączyłem rozpracowane urządzenia oraz zestawy i inne obwody, znalezione w czasopismach, zmieniając ich zastosowanie, aby powstawały nowe rzeczy.

Jako małe dziecko, zawsze byłem zafascynowany odkrywaniem, jak rzeczy działają. Dlatego miałem zwyczaj rozbierania ich na części. Ta pasja rozwijała się, a celem stawały się różne nieużywane sprzęty domowe, które następnie rozkładałem na najmniejsze kawałki. W końcu ludzie zaczęli przynosić mi wszelkiego rodzaju urządzenia do rozmontowania. Moim największym projektem w tym czasie były zmywarka i wczesny komputer, który pochodził z biura ubezpieczeniowego. Ten komputer miał dużą drukarkę, karty elektroniczne, magnetyczne czytniki kart i wiele innych rzeczy, które oczywiście były bardzo interesujące, a ich całkowite rozłożenie było wspaniałym wyzwaniem.

Po rozmontowaniu wielu urządzeń znałem elementy elektroniczne i z grubsza wiedziałem, do czego służą. Na dobitek mój dom był pełen starych czasopism elektronicznych, które mój ojciec kupił zapewne na początku lat siedemdziesiątych. Spędziłem wiele godzin, czytając te artykuły i przyglądając się schematom obwodów – bez większego zrozumienia.

Czytanie w kółko artykułów wraz ze wzrostem wiedzy zdobywanej dzięki rozkładaniu sprzętu na części tworzyły powoli efekt spirali.

Wspaniały przełom nastąpił w pewne Boże Narodzenie, gdy mój ojciec dał mi zestaw pozwalający nastolatkom uczyć się elektroniki. Każdy element był umieszczony w plastikowej kostce, która magnetycznie łączyła się z innymi kostkami, ustanawiając połączenie. Na wierzchu był umieszczony symbol elektroniczny. Nie zdawałem sobie wtedy sprawy, że ta zabawka, która powstała w latach sześćdziesiątych, była także charakterystyczna dla niemieckiego stylu projektowania, ponieważ jej projektantem był Dieter Rams.

Dzięki temu nowemu narzędziu mogłem szybko składać obwody i wypróbowywać ich działanie. Cykl prototypowania stawał się coraz krótszy.

Następnie budowałem radia, wzmacniacze, obwody wydające straszne hałasy i przyjemne dźwięki, czujniki deszczu i małe roboty.

Szukałem długo słowa, które podsumowuje sposób pracy bez specyficznego planu, zaczynający się od jednego pomysłu, a kończący się na całkowicie nieoczekiwanym wyniku. W końcu nazwałem to „majstrowaniem" (ang. tinkering). Odkryłem, że to słowo jest używane w wielu innych dziedzinach do opisania sposobu działania i określenia osób, które wytyczają drogę rozwoju. Na przykład pokolenie francuskich reżyserów, którzy powołali do życia „Nową falę", było nazywane mianem „tinkerers". Najlepsza definicja majstrowania, jaką kiedykolwiek spotkałem, pochodzi z wystawy w Exploratorium w San Francisco*:

> Majstrowanie polega na próbowaniu czegoś, gdy nie do końca wiemy, co chcemy osiągnąć. Kierunek wskazuje humor, wyobraźnia i zaciekawienie. Podczas majstrowania nie ma żadnych instrukcji – ale nie ma też błędów ani dobrych czy złych sposobów wykonania. Jest to rozpoznawanie, jak rzeczy działają i przerabianie ich. Przyrządy, maszyny, niedopasowane obiekty działające w harmonii – to istota majstrowania. Majstrowanie to zasadniczo proces łączący zabawę i dociekanie.

* *www.exploratorium.edu/tinkering*

Dzięki moim wczesnym eksperymentom wiedziałem, ile doświadczenia wystarcza do zbudowania obwodu działającego zgodnie z oczekiwaniami, gdy zaczynamy od podstawowych elementów.

Inny przełom przyszedł latem 1982 r., kiedy przyjechałem do Londynu z moimi rodzicami i spędziłem wiele czasu zwiedzając Muzeum Nauki. Właśnie otworzyli wtedy nowy dział komputerowy, gdzie dzięki seriom ćwiczeń z instruktorem nauczyłem się podstaw matematyki binarnej i programowania.

Wtedy odkryłem, że w wielu zastosowaniach inżynierowie nie muszą już budować obwodów z podstawowych elementów, ale zamiast tego mogą implementować znaczną część logiki przy użyciu mikroprocesorów. Oprogramowanie zastąpiło wiele godzin projektowania elektronicznego i pozwoliło skrócić cykl majstrowania.

Gdy wróciłem, zacząłem oszczędzać pieniądze, ponieważ chciałem kupić komputer i nauczyć się programować.

Moim pierwszym i najważniejszym projektem po tym wydarzeniu było użycie mojego fabrycznie nowego komputera ZX81 do sterowania spawarką. Wiem, że nie brzmi to jak bardzo ekscytujący projekt, ale był potrzebny i stanowił dla mnie wspaniałe wyzwanie, ponieważ właśnie nauczyłem się programować. W tym momencie stało się jasne, że pisanie wierszy kodu zabiera mniej czasu niż modyfikowanie złożonych obwodów.

Wiele parę lat później zacząłem myśleć, że to doświadczenie pozwala mi uczyć osoby, które niewiele pamiętają z lekcji matematyki, i zarazić je tym samym entuzjazmem i zdolnością do majstrowania, które miałem w młodości i zachowałem do tej pory.

—Massimo

Podziękowania od Massimo Banzi

Książkę tę dedykuję Ombretcie.

Podziękowania Michaela Shiloha

Dedykuję tę książkę memu bratu i rodzicom.

Nade wszystko chciałbym podziękować Massimo za zaproszenie mnie do pracy nad kolejnym wydaniem tej książki i za wprowadzenie mnie do społeczności Arduino. Uczestnictwo w tym projekcie to prawdziwy zaszczyt i radość.

Dziękuję Brianowi Jepsonowi za wskazówki, zachęty i wsparcie. Frank Teng zadbał, abym nie zbaczał z wytyczonego planu. Kim Cofer i Nicole Shelby wykonały świetną robotę przy redagowaniu tekstu i przygotowywaniu go do produkcji.

Dziękuję mojej córce Yasmine za tak dobre myśli, za nieustające wsparcie i dodawanie odwagi i za to, że nadal uważa za fajne to, że jestem jej ojcem. Niewiele bym zdziałał przy tej książce bez jej pomocy.

Jako ostatnią, ale zdecydowanie nie najmniej ważną muszę wymienić Judy Aime' Castro. Dziękuję za niekończące się godziny spędzone nad przekształcaniem moich niezdarnych szkiców w piękne ilustracje, za dyskusje o różnych aspektach książki i za nieograniczoną cierpliwość. Bez twojej pomocy to na pewno nie mogło się udać.

Konwencje zastosowane w tej książce

W tej książce użyliśmy następujących konwencji typograficznych:

Kursywa. Wskazuje nowe terminy, adresy URL i e-mail oraz nazwy i rozszerzenia plików.

`Czcionka stałopozycyjna`. Jest używana do listingów programów, a także wewnątrz akapitów, aby odwołać się do elementów programu, takich jak nazwy zmiennych lub funkcji, bazy danych, typy danych, zmienne środowiskowe, instrukcje i słowa kluczowe.

`Stała szerokość i pogrubienie`. Pokazuje polecenia lub inny tekst, który powinien być wpisany przez użytkownika dokładnie tak.

`Stała szerokość i kursywa`. Pokazuje tekst, który powinien być zastąpiony przez wartości podane przez użytkownika lub wynikające z kontekstu.

Ta ikona oznacza ogólną uwagę.

Ta ikona oznacza ostrzeżenie lub przestrogę.

Używanie przykładów kodu

Ta książka ma pomóc programistom w osiąganiu własnych celów. Zasadniczo jeśli książka zawiera przykładowy kod, można użyć go we własnych programach oraz dokumentacji. Nie ma potrzeby kontaktowania się w celu uzyskania zezwolenia, o ile nie planuje się reprodukcji znaczącej części kodu. Na przykład, napisanie programu wykorzystującego kilka fragmentów kodu z książki nie wymaga zezwolenia. Natomiast sprzedaż lub dystrybucja przykładów z książek wydawnictwa O'Reilly wymaga zezwolenia. Udzielenie odpowiedzi poprzez zacytowanie tej książki i przykładowego kodu nie wymaga zezwolenia. Umieszczenie znaczącej części przykładów kodu z tej książki w dokumentacji własnego produktu wymaga zezwolenia.

Będziemy wdzięczni za wskazanie źródła, choć nie jest to wymagane. Odwołanie do źródła zwykle zawiera nazwisko autora, tytuł, wydawnictwo oraz numer ISBN. Na przykład: „Getting Started With Arduino, Fourth Edition, by Massimo Banzi and Michael Shiloh (Make Community LLC). Copyright 2022 Massimo Banzi and Michael Shiloh, 978-1-6804-5693-6".

W przypadku wątpliwości, czy planowane zastosowanie przykładowego kodu wykracza poza przedstawione powyżej zezwolenia, prosimy o kontakt za pośrednictwem adresu e-mail: *books@make.co*.

1 Wstęp

Arduino to platforma do budowania interaktywnych fizycznych urządzeń o otwartych źródłach (open source), działających samodzielnie lub połączonych z Internetem.

Arduino zostało początkowo zaprojektowane dla artystów, projektantów i inne osoby, które chciały włączyć przetwarzanie fizyczne do swoich projektów, bez konieczności wcześniejszego ukończenia studiów inżynierskich. Później stało się platformą dla literalnie milionów ludzi, którzy pragnęli być innowacyjni przy użyciu technologii cyfrowych.

Sprzęt i oprogramowanie Arduino jest open source (o otwartych źródłach). Filozofia open source sprzyja wychowywaniu społeczności w ogólnym dzieleniu się wiedzą. Jest to wspaniała rzecz dla początkujących, gdyż pomoc często jest dostępna w pobliżu (w sensie geograficznym), a zawsze online, na wielu różnych poziomach umiejętności i w oszałamiającej gamie tematów. Przedstawiane przykładowe projekty nie są po prostu zdjęciami ukończonej pracy, ale zawierają instrukcje zbudowania ich samodzielnie albo użycia jako punktu wyjścia dla własnego, pochodnego lub powiązanego projektu.

Oprogramowanie Arduino, znane jako zintegrowane środowisko programowania (Integrated Development Environment – IDE), jest darmowe. Można je pobrać ze strony www.arduino.cc. Arduino IDE opiera się na języku Processing (*http://www.processing.org/*), który został opracowany w celu ułatwienia artystom tworzenia sztuki komputerowej bez konieczności zostania najpierw inżynierem oprogramowania. Arduino IDE może działać w systemach Windows, Mac i Linux.

Płytka Arduino UNO jest niedroga (około 23 dolarów) i dość tolerancyjna, jeśli chodzi o typowe błędy nowicjuszy. Jeśli komuś jednak uda się jakoś zniszczyć główny komponent Arduino Uno, można go wymienić za cenę 4 dolarów.

Projekt Arduino został opracowany w środowisku szkolnym i jest bardzo popularnym narzędziem edukacyjnym. Ta sama filozofia open source, która stworzyła społeczność ogólnie dzielącą się informacjami, odpowiedziami i projektami, udostępnia też metody kształcenia, programy nauczania i inne informacje.

Jako że sprzęt i oprogramowanie Arduino są open source, można pobrać projekt sprzętowy Arduino i zbudować je samodzielnie, aby wykorzystać jako punkt wyjścia dla własnego projektu opartego (lub wykorzystującego) Arduino w swoich ramach, albo po prostu po to, by zrozumieć, jak działa Arduino. To samo można zrobić z oprogramowaniem.

Arduino jest zaprojektowane z myślą o łatwości użycia, a niniejsza książka ma za zadanie pomóc początkującym bez wcześniejszego doświadczenia w rozpoczęciu korzystania z Arduino.

Niniejsza książka ma za zadanie pomóc początkującym poznać korzyści, jakie odniosą z nauki korzystania z platformy Arduino i działania zgodnie z jej koncepcją.

Grupa docelowa

Ta książka została napisana dla „pierwotnych" użytkowników Arduino: projektantów i artystów. Dlatego sposób wytłumaczenia pewnych rzeczy może doprowadzić niektórych inżynierów do szału. Faktycznie jeden z nich nazwał wstępne rozdziały mojego pierwszego brudnopisu „banałami". To oddaje istotę rzeczy. Zmierzmy się z tym: większość inżynierów nie potrafi wyjaśnić, co robią, innym inżynierom, a co dopiero zwykłym ludziom. Dlatego teraz zagłębimy się w banały.

Książka ta nie ma być podręcznikiem do nauki elektroniki lub programowania, ale w trakcie lektury, niejako „po drodze", nauczysz się co nieco z obu tych dziedzin.

> Gdy Arduino stawało się popularne, zorientowałem się, że eksperymentatorzy, hobbyści i hakerzy wszelakiej maści zaczęli używać go do tworzenia zarówno cudownych, jak i szalonych rzeczy. Pojąłem, że wszyscy jesteśmy artystami i projektantami na swoich własnych zasadach, zatem ta książka jest również dla was.
>
> —Massimo

Arduino powstało na podstawie pracy dyplomowej Hernanda Barragana na platformie Wiring w szkole IDII w Ivrea. Pracą oprócz mnie kierował Casey Reas.

Czym jest projektowanie interaktywne?

Arduino powstało, aby można było nauczać projektowania interaktywnego, dyscypliny projektowej, która umieszcza prototypowanie w centrum swojej metodologii. Istnieje wiele definicji projektowania interaktywnego, ale najbardziej podoba mi się następująca:

> *Projektowanie interaktywne to projektowanie dowolnych doświadczeń interaktywnych.*

W dzisiejszym świecie projektowanie interaktywne skupia się na tworzeniu znaczących doświadczeń między nami (ludźmi) i obiektami. Jest dobrą drogą, która pozwala zagłębić się w tworzenie pięknych – a może nawet kontrowersyjnych – doświadczeń między nami a technologią. Projektowanie interaktywne stymuluje proces tworzenia dzięki interaktywnemu procesowi opartemu na prototypach o coraz większej dokładności. To podejście – również stanowiące część pewnego rodzaju konwencjonalnego projektowania – może być rozszerzone, aby obejmowało prototypowanie przy użyciu technologii. W szczególności – prototypowanie z wykorzystaniem elektroniki.

Szczególnym polem projektowania interaktywnego, dotyczącym Arduino, jest *programowanie urządzeń* (lub *projektowanie interakcji z urządzeniami*).

Czym jest programowanie urządzeń?

Programowanie urządzeń korzysta z elektroniki do prototypowania nowych materiałów dla projektantów i artystów. Obejmuje projektowanie interaktywnych obiektów, które mogą się komunikować z ludźmi przy użyciu czujników i elementów aktywnych sterowanych oprogramowaniem działającym w *mikrokontrolerze* (małym komputerze zawartym w jednym układzie scalonym).

W przeszłości korzystanie z elektroniki oznaczało konieczność nieustannej współpracy z inżynierami oraz tworzenie obwodów po jednym małym elemencie na raz. Te problemy powodowały, że kreatywne osoby trzymały się z dala od bezpośredniej zabawy z medium. Większość narzędzi była przeznaczona dla inżynierów i wymagała dużej wiedzy.

W ostatnich latach mikrokontrolery stały się tańsze i łatwiejsze w użyciu, co pozwala na tworzenie lepszych narzędzi. Jednocześnie komputery stały się szybsze i wydajniejsze, umożliwiając tworzenie lepszych (i łatwiejszych w użyciu) narzędzi programistycznych.

Dzięki Arduino dokonaliśmy postępu, który polegał na przybliżeniu tych narzędzi do nowicjuszy. Teraz tworzenie własnych projektów jest możliwe po zaledwie dwóch czy trzech dniach warsztatów – albo po przeczytaniu tej książki!

Korzystając z Arduino, projektant lub artysta może bardzo szybko poznać podstawy elektroniki i czujników, a następnie zacząć budować prototypy przy bardzo małym nakładzie finansowym.

2 Droga Arduino

Koncepcja Arduino jest oparta na tworzeniu rzeczy, zamiast ich omawiania. Jest to nieustanne szukanie szybszych i potężniejszych sposobów budowania lepszych prototypów. Zbadaliśmy wiele technik prototypowania i wynaleźliśmy *myślenie przy użyciu rąk*.

Klasyczna inżynieria polega na prostym procesie przechodzenia z A do B. Droga Arduino zachwyca możliwością zagubienia się na trasie i znalezienia zamiast tego C.

Jest to proces majstrowania, który tak lubimy – zabawa ze środkami w otwarty sposób i znajdowanie nieoczekiwanego. W tym szukaniu sposobów budowania lepszych prototypów wybraliśmy także pewną liczbę pakietów oprogramowania, które umożliwiają proces stałego zmieniania środków programistycznych i sprzętowych.

W następnych punktach zaprezentujemy pewne koncepcje, zdarzenia i osoby, które zainspirowały drogę Arduino.

Prototypowanie

Prototypowanie to najważniejsza koncepcja drogi Arduino: wykonujemy rzeczy i budujemy obiekty, które oddziałują z innymi obiektami, osobami i sieciami. Staramy się znaleźć prosty i szybki sposób prototypowania, a równocześnie możliwie najtańszy.

Wiele osób początkujących podchodzących do elektroniki po raz pierwszy myśli, że będzie się uczyć, jak budować wszystko od podstaw. Jest to strata energii: chcemy bardzo szybko sprawdzić, że coś działa, aby zmotywować się do podjęcia następnego kroku, a nawet zmotywować kogoś innego, aby nam za to zapłacił.

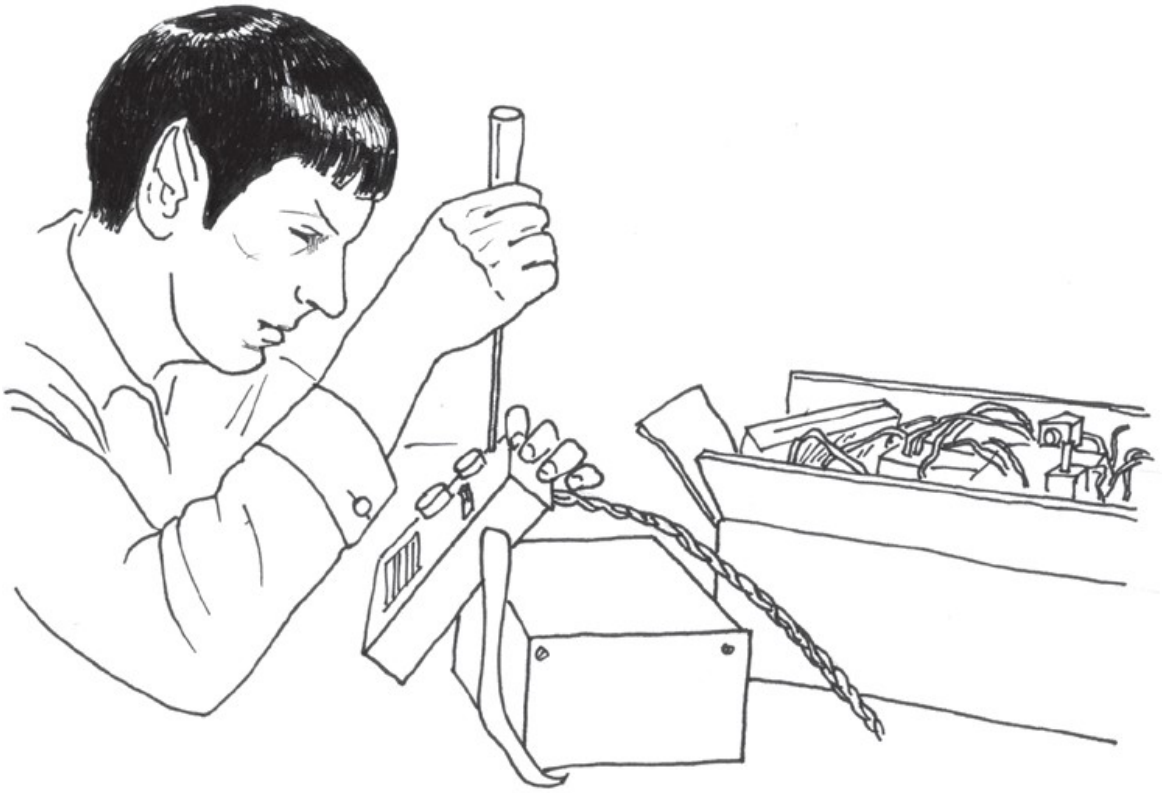
Dlatego wynaleźliśmy *prototypowanie oportunistyczne*: po co tracić czas i energię na budowanie od podstaw, proces wymagający czasu i głębokiej wiedzy technicznej, jeżeli możemy wziąć gotowe urządzenia

i rozpracować je, aby zbadać ciężką pracę wykonaną przez duże firmy i dobrych inżynierów?

Majstrowanie

Wierzymy, że jest ono istotą działania z technologią, badania różnych możliwości bezpośrednio sprzętowych i programistycznych – czasami bez dobrze zdefiniowanego celu.

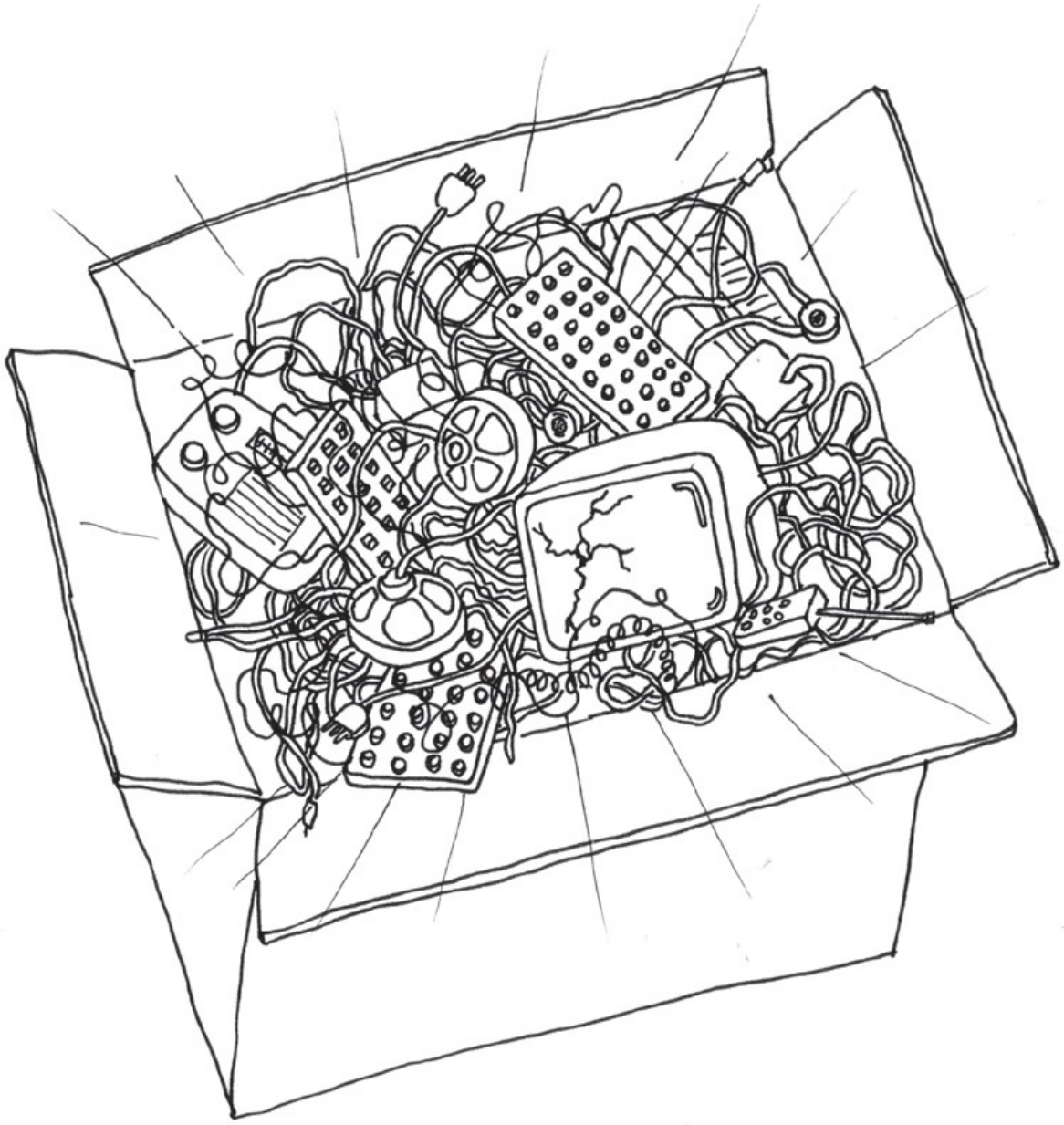
Ponowne używanie istniejącej technologii jest jednym z najlepszych sposobów majstrowania. Rozpracowywanie tanich zabawek lub starego, wyrzuconego sprzętu i używanie ich do czegoś innego jest jednym z najlepszych sposobów osiągnięcia wspaniałych wyników.



Uwielbiamy śmieci!

Obecnie ludzie wyrzucają mnóstwo elementów technicznych: stare drukarki, komputery, dziwne urządzenia biurowe, sprzęt techniczny, a nawet mnóstwo militariów. Zawsze istniał duży rynek obejmujący tę nadwyżkę technologii, szczególnie wśród młodych i/lub biednych majsterkowiczów oraz tych, którzy dopiero zaczynają. Ten rynek stał się oczywisty w Ivrea,

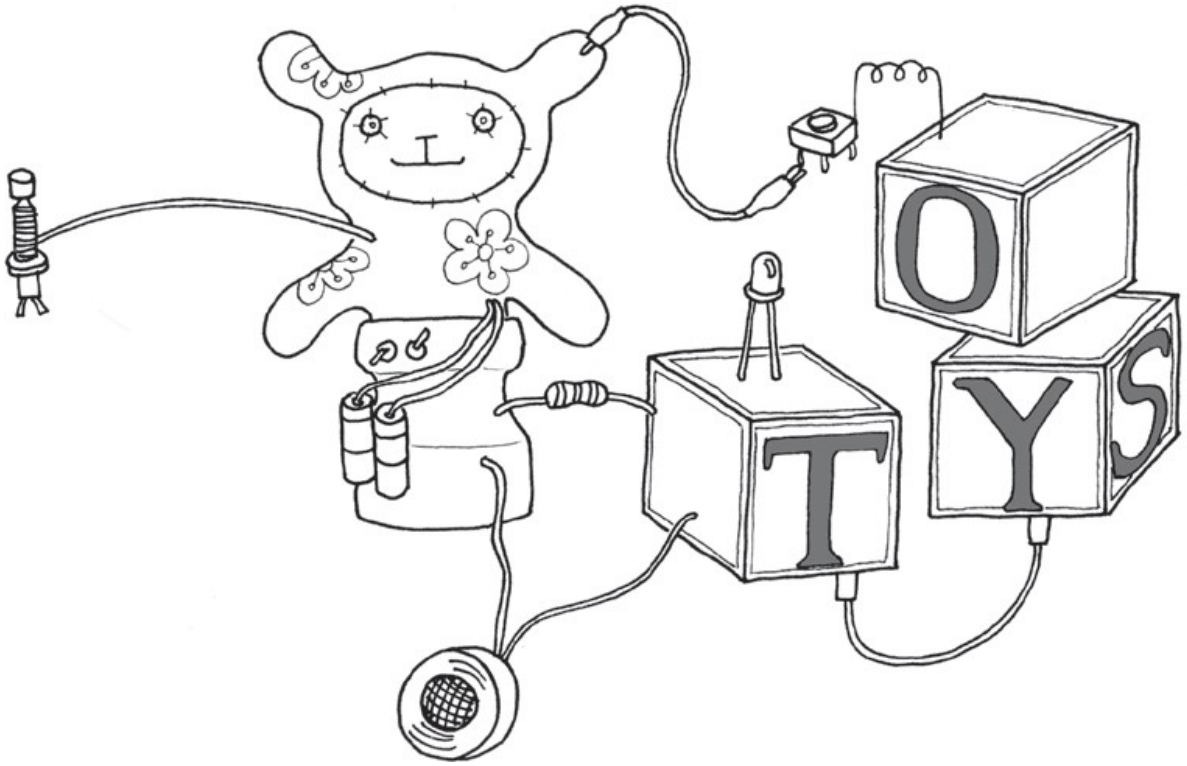
gdzie wynaleźliśmy Arduino. To miasto było siedzibą firmy Olivetti. Firma ta produkowała komputery od lat 60 XX w. W środku lat dziewięćdziesiątych wyrzucali wszystko na okoliczne złomowiska. Były one pełne części komputerowych, elementów elektronicznych i dziwnych urządzeń wszelkiego rodzaju. Spędzaliśmy tam niezliczone godziny, kupując najrozmaitsze ustrojstwa za grosze i rozpracowując je, aby tworzyć nasze prototypy. Kiedy można kupić tysiąc głośników za bardzo małe pieniądze, niewątpliwie ktoś wpadnie w końcu na jakiś pomysł, co z nimi zrobić. Warto zbierać śmieci i korzystać z nich, zanim zaczniemy coś robić od podstaw.



Rozpracowywanie zabawek

Zabawki są cudownym źródłem taniej technologii, którą można rozpracować i użyć ponownie, o czym świadczy praktyka łączenia obwodów wspomniana wcześniej. Przy obecnym napływie tysięcy bardzo tanich zabawek technologicznych z Chin, możemy szybko realizować pomysły przy użyciu kilku hałaśliwych kotów i paru mieczy świetlnych.

> Robiłem to przez parę lat, aby przekonać moich studentów, że technika nie jest straszna ani trudna do opanowania. Jednym z moich ulubionych źródeł jest broszura „Low Tech Sensors and Actuators" Usmana Haque i Adama Somlai-Fischera (*lowtech.propositions.org.uk*). Uważam ich opis tej techniki za doskonały i chętnie z niego korzystam.
>
> —Massimo



Współpraca

Współpraca między użytkownikami jest jedną z kluczowych zasad w świecie Arduino – korzystając z forum pod adresem *forum.arduino.cc* osoby z różnych części świata pomagają sobie nawzajem poznawać i uczyć się tej platformy. Zespół Arduino zachęca ludzi do współpracy na poziomie lokalnym, pomagając im tworzyć grupy użytkowników w każdym mieście, które odwiedza. Utworzyliśmy też witrynę o nazwie „Project Hub" (*https://create.arduino.cc/projecthub*), gdzie użytkownicy dokumentują swoje projekty i udostępniają je innym. Zdumiewające jest widzieć, jak dużo wiedzy przekazują te osoby do Internetu, aby każdy mógł z niej korzystać. Ta kultura udostępniania i pomagania sobie nawzajem jest jedną z rzeczy, z których jestem najbardziej dumny, gdy chodzi o Arduino.

3 Platforma Arduino

Arduino składa się z dwóch głównych komponentów: płytki Arduino, czyli sprzętu, na którym budujemy swoje obiekty, oraz środowiska Arduino IDE, czyli oprogramowania, które działa na naszym komputerze. Środowisko IDE służy do napisania *szkicu* (ang. *sketch*) – małego programu komputerowego, który ładujemy do płytki Arduino. Szkic zawiera instrukcje działania płytki.

Nie tak dawno praca ze sprzętem elektronicznym oznaczała tworzenie obwodów od podstaw, przy użyciu setek różnych elementów o dziwnych nazwach, takich jak opornik (rezystor), kondensator, induktor, tranzystor itd. Każdy obwód był „związany" z jednym konkretnym zastosowaniem, a wprowadzanie zmian wymagało cięcia przewodów, lutowania złączy itp.

Wraz z pojawieniem się technologii cyfrowych i mikroprocesorów funkcje, które były implementowane przez połączenia, zostały zastąpione programami. Oprogramowanie jest łatwiejsze do modyfikacji niż sprzęt. Wystarcza użyć klawiatury, aby drastycznie zmienić logikę urządzenia i wypróbować dwie lub trzy wersje w takim samym czasie, jaki zajęłoby przylutowanie paru oporników.

Sprzęt Arduino

Płytka Arduino to mała płytka mikrokontrolera, czyli mały obwód, zawierający cały komputer w jednym układzie scalonym (mikrokontroler).

> Ten komputer jest przynajmniej tysiąc razy słabszy niż MacBook, którego używam do napisania tej książki, ale też znacznie tańszy i bardzo przydaje się do budowania interesujących urządzeń.
>
> —Massimo

Przyjrzyjmy się płytce Arduino: widzimy czarny układ scalony z 28 „nóżkami” (a być może po prostu czarny kwadratowy kawałek plastyku, jeśli twoja wersja to wydanie SMD) – ten układ to ATmega328, serce płytki.

W istocie jest wiele różnych płytek Arduino, ale najbardziej popularna (o wiele bardziej od wszystkich pozostałych łącznie) jest Arduino Uno, którą opisujemy w tym rozdziale. W rozdziale 9 przedstawimy skrótowy przegląd całej rodziny Arduino, w tym najnowsze odmiany z procesorami ARM, co bardzo odróżnia je od płytek AVR.

Umieściliśmy (my, czyli zespół Arduino) na tej płytce wszystkie elementy konieczne do właściwej pracy mikrokontrolera i jego komunikacji z komputerem. Wersja płytki, której najczęściej będziemy używać w tej książce, to Arduino Uno, najprostsza w użyciu i najlepsza do nauki. Jednak podane instrukcje dotyczą także innych wersji płytki, zarówno tych starszych, jak i najnowszych.

Na rysunku 3-1 widzimy, że Arduino ma dwa rzędy pasków u górny i dołu, z wieloma etykietami. Te paski czy też słupki to złącza (styki, nazywane zwykle pinami), które umożliwiają przyłączanie czujników i aktuatorów. Czujnik jest czymś, co odczytuje jakąś cechę rzeczywistego świata i przekształca ją na sygnał, który komputer może zrozumieć. Aktuator (element wykonawczy) przekształca sygnał z komputera w jakieś rzeczywiste działanie. W dalszej części książki dowiemy się znacznie więcej na temat czujników i aktuatorów.

Początkowo wszystkie te złącza mogą być trochę niejasne. Oto objaśnienie pinów wejściowych i wyjściowych, których używania będziemy się uczyć w tej książce. Nie należy się martwić, jeśli na razie wszystko będzie wydawało się nieco przytłaczające – w tej książce jest wiele nowych koncepcji, do których przyzwyczajenie się może wymagać trochę czasu. Będziemy powtarzać te objaśnienia kilkukrotnie przy różnych okazjach, a tak naprawdę wszystko to zacznie nabierać sensu, gdy zaczniesz budować własne obwody i eksperymentować z wynikami.

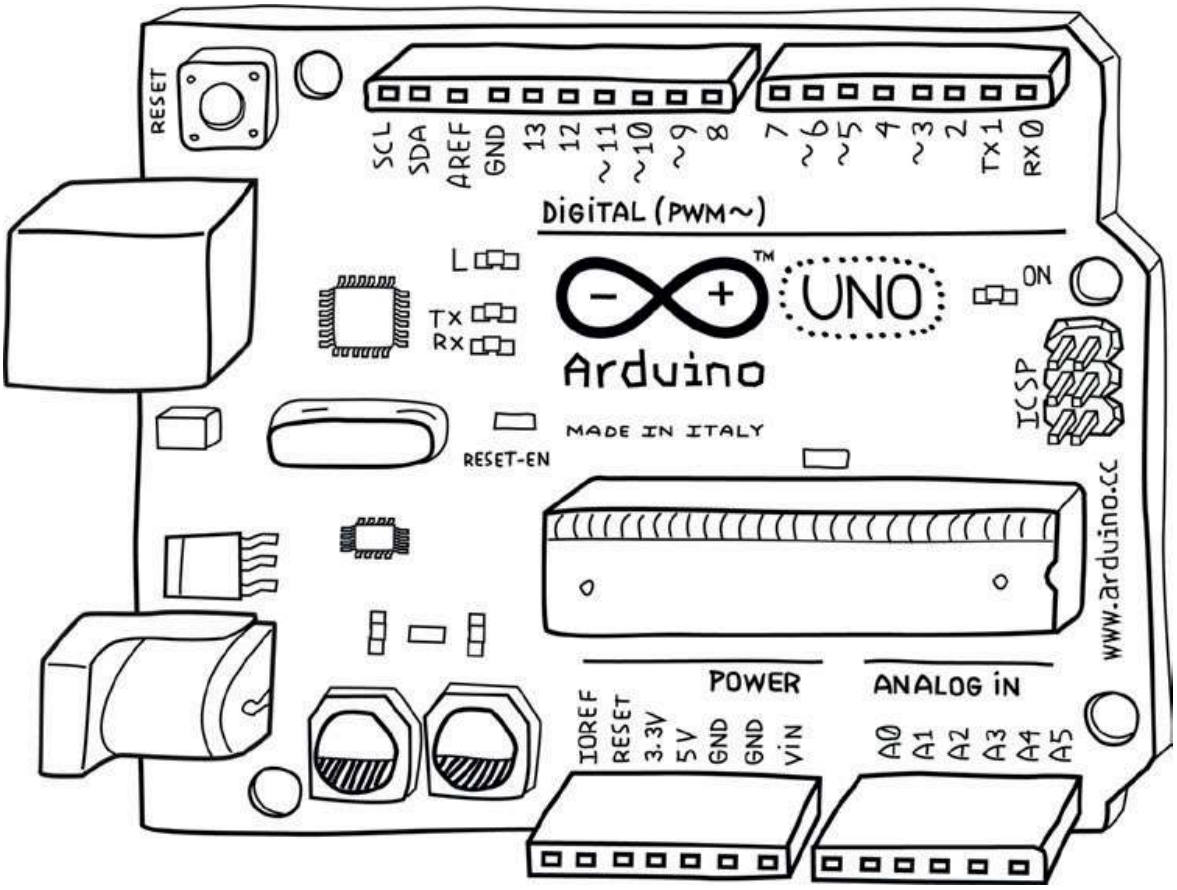
14 cyfrowych pinów IO (piny 0–13)

Te piny mogą działać jako *wejścia* lub *wyjścia*. Wejścia służą do odczytywania informacji z czujników, podczas gdy wyjścia pozwalają kontrolować aktuatory. Kierunek (do środka lub na zewnątrz) specyfikujemy

w szkicu utworzonym przy użyciu środowiska IDE. Cyfrowe wejścia mogą odczytywać tylko jedną z dwóch możliwych wartości, zaś cyfrowe wyjścia mogą dawać tylko jedną z dwóch wartości: HIGH (wysoki) oraz LOW (niski), zwykle interpretowanych jako 1 lub 0, odpowiednio.

6 analogowych pinów wejściowych (piny 0–5)
: Analogowe piny wejściowe służą do odczytu napięcia z czujników analogowych. W przeciwieństwie do wejść cyfrowych, które rozróżniają tylko dwa poziomy (HIGH i LOW), wejścia analogowe potrafią zmierzyć 1024 różne poziomy napięcia.

6 analogowych pinów wyjściowych (piny 3, 5, 6, 9, 10 i 11)
: Jest to w rzeczywistości sześć spośród pinów cyfrowych, które mogą wykonywać trzecią funkcję: zapewniać analogowe wyjście. Podobnie jak w przypadku pinów cyfrowych, za pomocą instrukcji w szkicu określamy, co pin powinien robić.



Rysunek 3-1. Arduino Uno

Płytka może być zasilana z portu USB komputera, większości ładowarek USB lub zasilacza (zalecane 9 V, końcówka jack 2,1 mm, środek dodatni).

Jeżeli nie ma żadnego źródła, podłączonego do gniazda zasilania, Arduino użyje prądu ze złącza USB, ale gdy tylko podłączymy źródło zasilania, płytka użyje go automatycznie. Uzyskiwanie zasilanie zarówno z zasilacza, jak i ze złącza USB jest bezpieczne i nie grozi uszkodzeniem płytki.

Oprogramowanie (IDE)

Zintegrowane środowisko oprogramowania, czyli IDE (Integrated Development Environment) to specjalny program działający na komputerze, który pozwala pisać szkice dla płytki Arduino w prostym języku opartym na języku Processing (*www.processing.org*). Magia odbywa się, gdy naciskamy przycisk, który przekazuje szkic do płytki: napisany kod jest tłumaczony na język C (którego nauczenie się zwykle sprawia trudności osobom początkującym), a następnie przekazywany do kompilatora avr-gcc – ważnej części otwartego oprogramowania, która dokonuje końcowego tłumaczenia na język zrozumiały dla mikrokontrolera. Ten ostatni krok jest dość ważny, ponieważ wtedy właśnie Arduino ułatwia nasze życie, ukrywając możliwie najwięcej złożoności programowania mikrokontrolerów.

Cykl programowania Arduino jest zasadniczo następujący:

1. Podłączamy płytkę do portu USB komputera.
2. Piszemy szkic, który ożywi płytkę.
3. Przekazujemy szkic na płytkę przy użyciu połączenia USB i czekamy parę sekund, aż płytka się uruchomi.
4. Obserwujemy, jak płytka wykonuje napisany szkic.

Instalacja Arduino na komputerze

W celu programowania płytki Arduino musimy najpierw pobrać środowisko projektowe (IDE) ze strony *www.arduino.cc/en/software*. Należy wybrać wersję odpowiednią dla danego systemu operacyjnego. W przypadku Windows należy wybrać opcję *Win 7 and Newer* (Win 7 i nowsze). Na kolejnej stronie witryny można zdecydować się na dokonanie wpłaty na rzecz rozwoju Arduino IDE, ale można też kliknąć przycisk z napisem *JUST DOWNLOAD* (po prostu pobierz). Po zapisaniu pliku należy go uruchomić, po czym wykonać wskazówki z kolejnych podrozdziałów.

Instalowanie IDE: MacOS

Gdy zakończy się pobieranie pliku, zależnie od ustawień przeglądarki zostanie on rozpakowany automatycznie, ale może być konieczne jego ręczne rozpakowanie, zazwyczaj poprzez podwójne kliknięcie.

Wypakowaną aplikację Arduino przeciągnij do swojego folderu Applications (Aplikacje).

Konfigurowanie sterowników

Arduino Uno używa sterownika dostarczanego z systemem operacyjnym MacOS, zatem niczego nie trzeba instalować.

Teraz, gdy IDE jest już zainstalowane, podłącz Arduino Uno do swojego Maca za pomocą kabla USB.

Zielona dioda LED na płytce, oznaczona PWR, powinna się zapalić, zaś żółta dioda opisana literą L powinna zacząć mrugać.

Na ekranie komputera możesz zauważyć wyskakujące okno informujące, że został wykryty nowy interfejs sieciowy.

Jeśli tak się stanie, kliknij Network Preferences, a gdy się otworzy, kliknij Apply. Urządzenie Uno pojawi się jako Not Configured, ale działa poprawnie. Zamknij System Preferences.

Teraz, gdy masz już skonfigurowane oprogramowanie, musisz wybrać odpowiedni port do komunikacji z Arduino Uno.

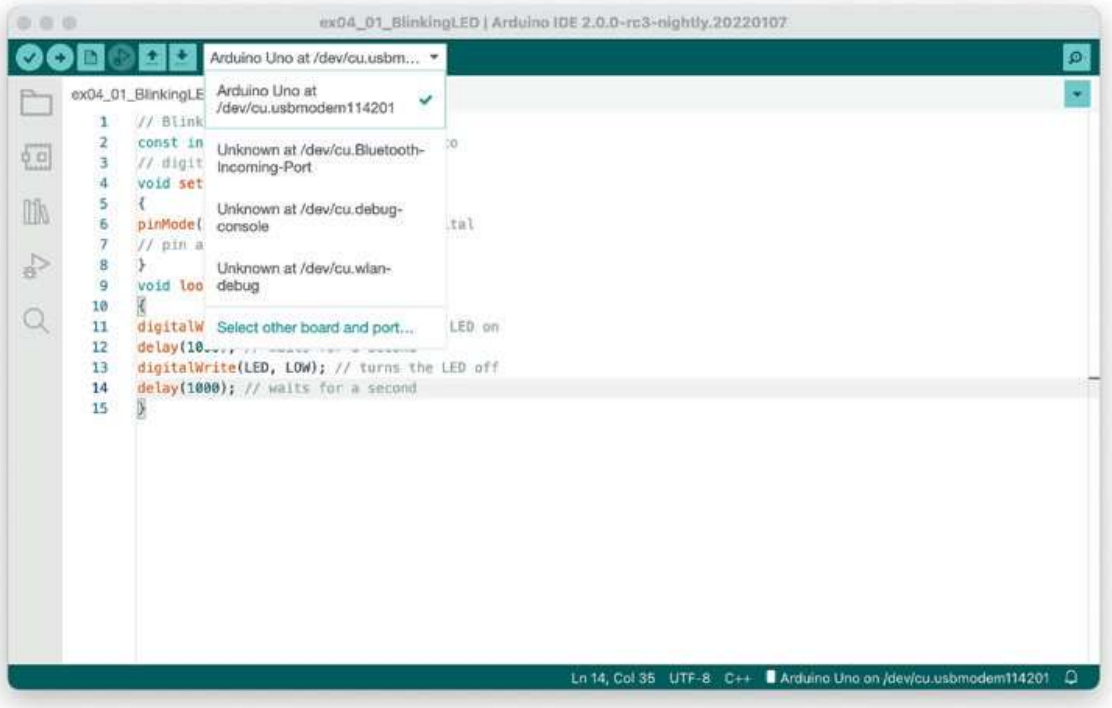
Identyfikacja portu

Uruchom Arduino IDE, albo poprzez folder Applications, albo przy użyciu Spotlight.

W menu Tools programu Arduino IDE wybierz Serial Port, a następnie wybierz port, który zaczyna się od /dev/cu.usbmodem lub /dev/tty .usbmodem. Będą one zapewne również zawierać wpis Arduino/ Genuino Uno po nazwie portu. Obydwa z tych portów odnoszą się do twojej płytki Arduino i nie ma różnicy, którą z nich wybierzesz.

Rysunek 3-2 pokazuje listę portów.

Prawie gotowe! Ostatnią rzeczą, którą trzeba sprawdzić, jest upewnienie się, że Arduino IDE jest skonfigurowane dla typu płytki, której używasz.



Rysunek 3-2. Lista portów szeregowych w Arduino IDE na Macu

W menu Tools programu Arduino IDE wybierz Board, po czym zaznacz Arduino Uno. Jeśli masz inną płytkę, musisz wybrać ten typ (nazwa płytki jest wydrukowana obok symbolu Arduino).

Gratulacje! Twoje oprogramowanie Arduino jest zainstalowane, skonfigurowane i gotowe do użycia. Możesz teraz przejść do rozdziału 4.

Jeśli pojawią się jakieś trudności na którymkolwiek z tych kroków, zajrzyj do rozdziału 11, „Rozwiązywanie problemów”.

Instalowanie IDE: Windows

Po zakończeniu pobierania pliku kliknij go podwójnie, aby uruchomić instalator.

Zostanie wyświetlona licencja użytkownika. Przeczytaj ją, a jeśli się z nią zgadzasz, kliknij przycisk I Agree.

Pojawi się lista komponentów do zainstalowania i domyślnie wszystkie są wybrane. Pozostaw je bez zmian i kliknij Next.

Zostaniesz poproszony o wybór folderu instalacji i instalator zaproponuje domyślny do tego celu. O ile nie masz dobrego powodu, aby tego nie robić, zaakceptuj domyślną lokalizację i kliknij Install.

Instalator będzie wyświetlać postęp w miarę rozpakowywania i instalowania plików.

Po zainstalowaniu wszystkich plików pojawi się okno z pytaniem o zezwolenie na instalację sterowników. Kliknij Install.

Gdy instalator zakończy pracę, kliknij Close.

Konfigurowanie sterowników

Po zainstalowaniu IDE podłącz Arduino Uno do komputera za pomocą kabla USB.

Zielona dioda LED na płytce, oznaczona PWR, powinna się zapalić, zaś żółta dioda opisana literą L powinna zacząć mrugać.

Pojawi się okno Found New Hardware Wizard (Znaleziono nowy sprzęt) i system Windows powinien sam odszukać potrzebne sterowniki.

Jeśli pojawią się jakieś trudności na którymkolwiek z tych kroków, zajrzyj do rozdziału 11, „Rozwiązywanie problemów”, do podrozdziału „Problemy z instalowaniem sterowników w Windows”.

Teraz, gdy masz już skonfigurowane oprogramowanie, musisz wybrać odpowiedni port do komunikacji z Arduino Uno.

Identyfikacja portu

Uruchom Arduino IDE, albo przy użyciu skrótu na pulpicie, albo z menu Start.

W menu Tools programu IDE wybierz Serial Port. Zobaczysz jeden lub więcej portów COM z różnymi numerami. Jeden z tych portów zapewne będzie miał wpis Arduino/Genuino Uno po nazwie portu. Ten właśnie należy zaznaczyć.

Jeśli żaden z portów nie ma wpisu Arduino/Genuino Uno po nazwie portu, dostępny jest alternatywny sposób zidentyfikowania właściwego portu:

1. Zanotuj dostępne numery portów.
2. Odłącz Arduino od komputera, sprawdź ponownie listę portów i zobacz, który port COM zniknął. Podłącz Arduino z powrotem i wybierz ten port, który się pojawił.
   (Zniknięcie portu może wymagać paru chwil i konieczne może być opuszczenie menu Tools i jego ponowne otwarcie, aby odświeżyć listę portów).

Jeśli pojawią się jakieś trudności na którymkolwiek z tych kroków, zajrzyj do rozdziału 11, „Rozwiązywanie problemów", do podrozdziału „Identyfikowanie portu COM w Windows".

Po ustaleniu właściwego przypisania portu COM trzeba wybrać ten port w menu Tools→Serial Port w Arduino IDE.

Prawie gotowe! Ostatnią rzeczą, którą trzeba sprawdzić, jest upewnienie się, że Arduino IDE jest skonfigurowane dla typu płytki, której używasz.

W menu Tools programu Arduino IDE wybierz Board, po czym zaznacz Arduino Uno. Jeśli masz inną płytkę, musisz wybrać ten typ (nazwa płytki jest wydrukowana obok symbolu Arduino).

Gratulacje! Twoje oprogramowanie Arduino jest zainstalowane, skonfigurowane i gotowe do użycia. Możesz teraz przejść do rozdziału 4.

Instalowanie IDE: Linux

Po zakończeniu pobierania pliku przejdź do folderu, w którym plik został zapisany. Typowo będzie to folder ~/Downloads. Rozpakuj plik, wpisując polecenie:

```
tar xf arduino-ide_2.0.0-rc3_Linux_64bit.tar.xz
```

(w razie potrzeby zmieniając nazwę zgodnie z pobranym plikiem). Zajmie to kilka sekund i w tym czasie nic nie jest wyświetlane. Gdy polecenie zakończy pracę, znajdziesz nowy folder:

```
arduino-ide_2.0.0-rc3_Linux_64bit
```

Folder ten można przenieść w wygodniejsze miejsce, na przykład do swojego folderu domowego, wpisując:

```
mv arduino-ide_2.0.0-rc3_Linux_64bit ~
```

Konfigurowanie sterowników

Arduino Uno używa sterownika dostarczanego przez system operacyjny Linux, zatem nic nie trzeba instalować.

Przyznawanie uprawnień do portów szeregowych

Porty szeregowe, których używa Arduino, normalnie są zastrzeżone tylko dla administratorów, zatem trzeba będzie przyznać sobie uprawnienie do używania tych portów. Można to zrobić, dodając siebie (czyli swoje konto użytkownika) do grupy dial out, wpisując poniższe polecenie:

```
sudo usermod -a -G dialout $USER
```

Pojawi się polecenie wpisania hasła w celu uwierzytelnienia. Po wpisaniu hasła polecenie kończy działanie, ale nie da efektów do chwili zrestartowania sesji, zatem trzeba albo się wylogować i zalogować ponownie, albo wykonać restart komputera.

Teraz, gdy masz już zainstalowane oprogramowanie, musisz wybrać odpowiedni port do komunikacji z Arduino Uno.

Identyfikacja portu

Uruchom Arduino IDE, wpisując:

```
~/arduino-ide_2.0.0-rc3_Linux_64bit/arduino
```

W menu Tools programu Arduino IDE wybierz Serial Port. Zobaczysz jeden lub więcej portów szeregowych o nazwach podobnych do /dev/tty. Jeden z tych portów będzie miał wpis Arduino/Genuino Uno po nazwie portu. Ten właśnie należy wybrać.

Po ustaleniu właściwego przypisania portu COM trzeba wybrać ten port w menu Tools→Serial Port w Arduino IDE.

Prawie gotowe! Ostatnią rzeczą, którą trzeba sprawdzić, jest upewnienie się, że Arduino IDE jest skonfigurowane dla typu płytki, której używasz.

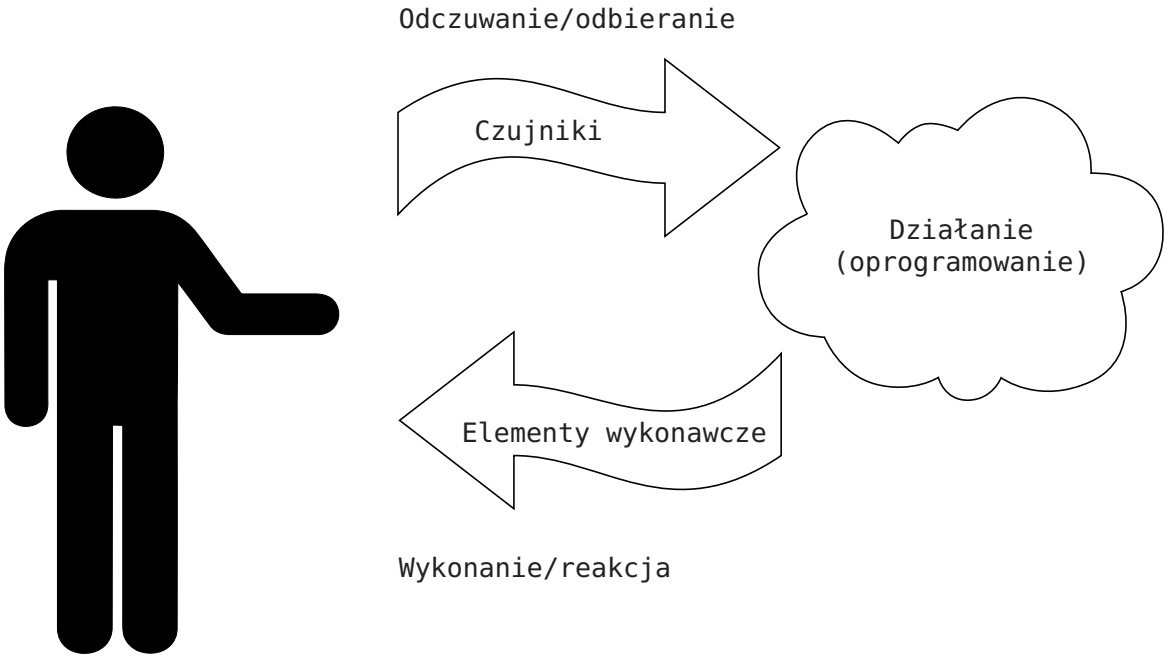
W menu Tools programu Arduino IDE wybierz Board, po czym zaznacz Arduino Uno. Jeśli masz inną płytkę, musisz wybrać ten typ (nazwa płytki jest wydrukowana obok symbolu Arduino).

Gratulacje! Twoje oprogramowanie Arduino jest zainstalowane, skonfigurowane i gotowe do użycia. Możesz teraz przejść do rozdziału 4.

4 Prawdziwe wprowadzenie do Arduino

Teraz możemy zacząć uczyć się budować i programować pierwsze interaktywne urządzenia.

Anatomia urządzeń interaktywnych



Rysunek 4-1. Urządzenie interaktywne

Wszystkie obiekty budowane przy użyciu Arduino są zgodne z prostym wzorcem, który nazywamy „urządzeniem interaktywnym". Urządzenie interaktywne to obwód elektroniczny, który ma zdolność odbierania „wrażeń" ze środowiska przy użyciu czujników (elementów elektronicznych, które konwertują pomiary świata rzeczywistego na sygnały elektryczne). Urządzenie przetwarza informacje odbierane z czujników zgodnie z działaniem zaimplementowanym w formie oprogramowania. Następnie