ARDUINO
IL LIBRO DEI PROGETTI

ARDUINO - IL LIBRO DEI PROGETTI

AUTORI
I progetti e i testi sono di Scott Fitzgerald e Michael Shiloh
Con il contributo di Tom Igoe

DESIGN E ART DIRECTION
TODO
Giorgio Olivero, Mario Ciardulli, Vanessa Poli, Michelle Nebiolo
todo.to.it

FABBRICAZIONE DIGITALE E PROJECT MANAGEMENT
Officine Arduino Torino
Katia De Coi, Enrico Bassi

ADVISOR E SUPPORTER
Massimo Banzi, Gianluca Martino, Smart Projects

TEST DEI PROGETTI E REVISIONI DEL TESTO
Michael Shiloh, Michelle Nebiolo, Katia De Coi, Alessandro Buat,
Federico Vanzati, David Mellis

TRADUZIONE E IMPAGINAZIONE DELLA VERSIONE ITALIANA
Luisa Castiglioni, press-office.co

GRAZIE
Un grande ringraziamento va all'intera comunità di Arduino per i continui contributi, il supporto e il feedback.
Un grazie speciale va al team di Fritzing: alcune illustrazioni di componenti elettronici usate nel libro sono state prese o modificate dal progetto open-source Fritzing (www.fritzing.org).
Un ringraziamento di cuore va a Paul Badger per la libreria CapacitiveSensor usata nel progetto 13.

Progettato, stampato e rilegato a Torino, Italia
Giugno 2013

INDICE

00

Tutti, ogni giorno, utilizzano la tecnologia. La maggior parte di noi lascia la programmazione agli ingegneri perché pensa che codice e elettronica siano complicati e difficili; in realtà, possono essere divertenti ed emozionanti. Grazie ad Arduino designer, artisti, hobbisti e studenti di tutte le età stanno imparando a creare cose che si accendono, si muovono, rispondono a persone, animali, piante e al resto del mondo.

Nel corso degli anni Arduino è stato utilizzato come "cervello" in migliaia di progetti, uno più creativo dell'altro. Una comunità mondiale di maker si è raccolta intorno a questa piattaforma open source, passando dal personal computer alla personal fabrication, e contribuendo a un nuovo mondo fatto di partecipazione, cooperazione e condivisione.

Arduino è aperto e semplice. È fondato su lezioni che abbiamo imparato insegnando ai nostri studenti: basta partire dal presupposto che imparare a utilizzare le tecnologie digitali è semplice e accessibile. Così l'elettronica e il codice diventano strumenti creativi che chiunque può utilizzare – come pennelli e colori. Questo libro ti guida attraverso le basi in maniera pratica attraverso progetti creativi per imparare facendo. Una volta apprese le basi, avrai una gamma di software e circuiti da utilizzare per creare qualcosa di bello e per inventare qualcosa di divertente.

COSE CHE TI SERVIRANNO

Batteria da 9V

Piccola sorgente di luce come una torcia elettrica

Materiale conduttivo come un foglio di alluminio o una maglia di rame

Carta colorata

Forbici

Un vecchio CD o DVD

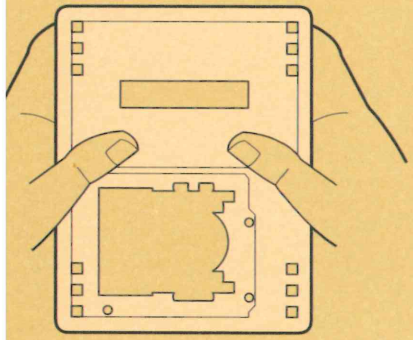
Scotch e colla

Una scatola in cui poter fare dei buchi

Attrezzi di base come un cacciavite

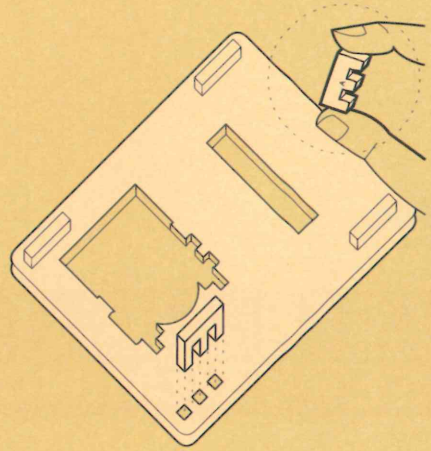
Dispositivo elettronico alimentato a 9V
Qualsiasi dispositivo elettronico alimentato a batteria con almeno un interruttore o pulsante su cui puoi smanettare.

Saldatore
(necessario solo nel Progetto 15)



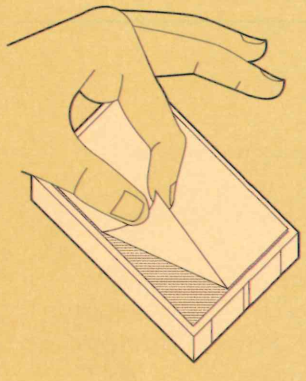
❷

Continua fino a che hai separato tutti i pezzi.



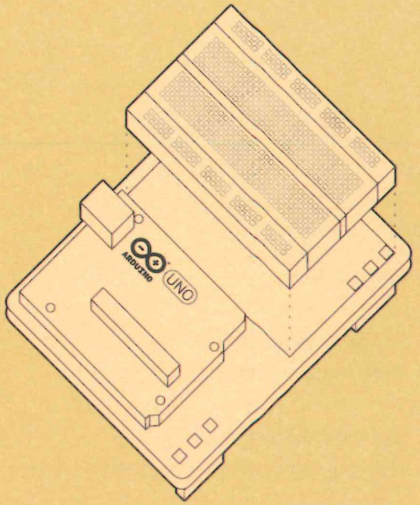
❸

Metti i pezzi marchiati con una "A" nei buchi negli angoli, per creare i piedini della base.



❺

Rimuovi accuratamente il foglio di protezione sul retro dalla breadboard.



❻

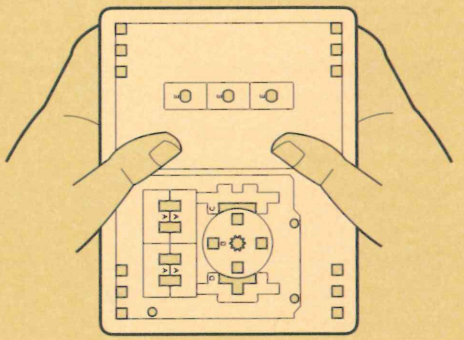
Posiziona la breadboard sulla base di legno, accanto all'Arduino Uno.

Nel tuo Starter Kit hai trovato una base di legno pretagliata e facile da assemblare che renderà ancora più semplice il lavoro sui tuoi progetti – che siano o no parte di questo libro.

Per costruirla, togli la base di legno dalla scatola e segui le istruzioni a destra.

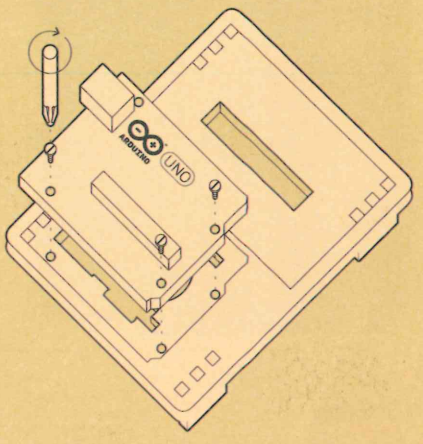
Fai attenzione a utilizzare solo le parti che vengono visualizzate, ma non perdere gli altri pezzi: ti serviranno per alcuni dei prossimi progetti.

Iniziamo!



❶

Prendi la base di legno e, con attenzione, separa i pezzi.



❹

Fissa l'Arduino Uno alla base con 3 viti. Fai attenzione a non stringerle troppo.

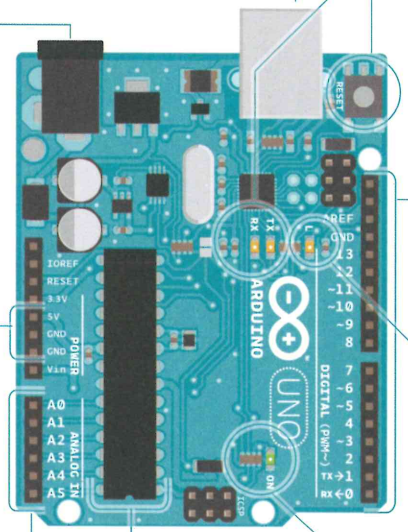
LA SCHEDA ARDUINO

Connettore di alimentazione
Serve per alimentare Arduino quando non è collegato a una porta USB. Vanno bene tensioni tra 7 e 12V.

Porta USB
Serve per alimentare l'Arduino Uno, caricare i tuoi sketch nella scheda e per comunicare con Arduino (via `Serial println()`).

Pulsante di reset
Fa ripartire il microcontrollore ATmega.

LED TX e RX
Indicano le comunicazioni tra Arduino e il computer. Aspettati uno sfarfallio durante il caricamento dello sketch, così come durante la comunicazione seriale. Utile per il debug.



Piedini digitali
Utilizza questi piedini con `digitalRead()`, `digitalWrite()` e `analogWrite()`. Quest'ultimo funziona solo sui piedini con il simbolo PWM.

LED del piedino 13
L'unico attuatore integrato ad Arduino Uno. Oltre a essere l'oggetto del tuo primo sketch di lampeggio, questo LED è molto utile per il debug.

Piedini GND e 5V
Usa questi piedini per fornire un'alimentazione di 5V e massa al tuo circuito.

Microcontrollore ATmega
Il cuore di Arduino Uno.

Analog in
Usa questi piedini con `analogRead()`.

Power LED
Indica che Arduino sta ricevendo alimentazione. Utile per il debug.

TAVOLA DEI SIMBOLI

FILI COLLEGATI

TRANSISTOR

PULSANTE

FILI NON COLLEGATI

MOSFET

MOTORE

INTERRUTTORE DI INCLINAZIONE

FOTORESISTENZA

POTENZIOMETRO

RESISTENZA

DIODO

PIEZO

LED

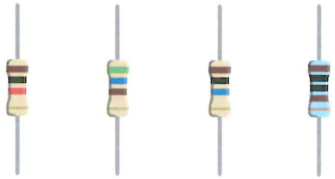
CONDENSATORE

BATTERIA

CONDENSATORE POLARIZZATO

MASSA

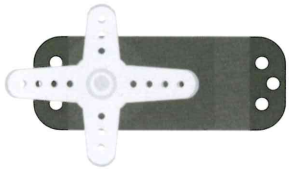
In questo libro ti mostreremo i circuiti sia con illustrazioni realistiche sia con schemi elettrici.
Le illustrazioni ti daranno un'idea di che aspetto avrà la breadboard in una possibile implementazione del progetto. Gli schemi, invece, utilizzano i simboli per catturare l'essenza dei circuiti: presentano i componenti e i modi in cui sono connessi in forma chiara, concisa e non ambigua, ma non la loro organizzazione fisica. Schemi e simboli sono i modi con cui vengono rappresentati i circuiti elettronici. Quando esplorerai il mondo dell'elettronica scoprirai che alcuni libri e siti web forniscono solo schemi, quindi imparare a leggere i circuiti in questo modo è una preziosa competenza.
Qui trovi i simboli che useremo in tutto il libro.



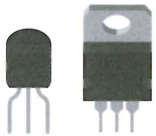
Resistenze - Resistono al flusso di energia elettrica in un circuito modificando di conseguenza la tensione e la corrente. I valori di resistenza sono misurati in Ohm (rappresentata dal carattere greco omega, Ω). Le strisce colorate delle resistenze indicano il loro valore (vedi tabella colori delle resistenze, a pagina 41).

Sensore di inclinazione - Un tipo di interruttore che si apre o si chiude a seconda del suo orientamento. Tipicamente sono cilindri cavi con all'interno una sfera di metallo che collega due contatti quando il sensore è in verticale.

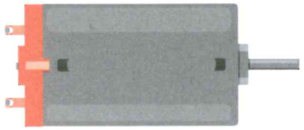
Sensore di temperatura - Cambia la sua tensione di uscita in funzione della temperatura del componente. I piedini esterni si collegano all'alimentazione e a massa. La tensione sul piedino centrale è proporzionale alla temperatura.



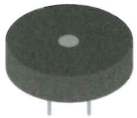
Servomotore - Un tipo di motore che può ruotare solo di 180 gradi. Si controlla inviando impulsi elettrici da Arduino. Questi impulsi dicono al motore in quale posizione spostarsi.



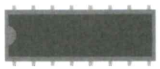
Transistor - Un dispositivo con tre piedini che può funzionare da interruttore elettronico. Utile per il controllo di componenti ad alta tensione/alta corrente, come i motori. Un piedino si collega a massa, un altro al componente da controllare e il terzo ad Arduino. Quando il transistor riceve tensione sul piedino collegato ad Arduino chiude il circuito tra la massa e il carico.



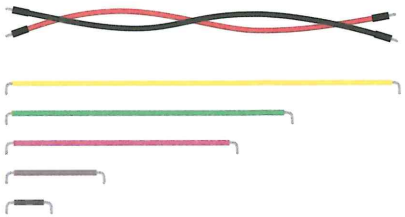
Motore a corrente continua - Converte l'energia elettrica in energia meccanica quando la corrente elettrica viene applicata ai suoi terminali. Gli avvolgimenti all'interno del motore vengono magnetizzati quando vi fluisce la corrente. Questi campi magnetici attraggono e respingono magneti permanenti facendo ruotare l'albero. Se la direzione della corrente viene invertita, il motore gira in senso opposto.



Piezo - Un componente elettrico per rilevare le vibrazioni e produrre suoni.



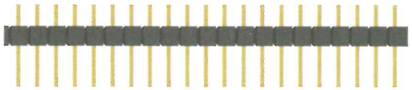
Ponte H - Un circuito che permette di controllare la polarità della tensione applicata a un carico, in genere un motore. Il ponte H nel kit è un circuito integrato, ma potrebbe anche essere costruito con un certo numero di componenti separati.



Ponticelli - Da utilizzare per collegare tra loro i componenti sulla breadboard e ad Arduino.

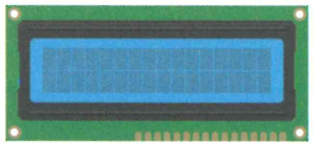
Potenziometro - Una resistenza variabile con tre piedini. Due dei piedini sono connessi alle estremità di una resistenza fissa. Il piedino centrale, o cursore, si muove lungo la resistenza, dividendola in due metà. Quando i piedini esterni del potenziometro sono collegati all'alimentazione e a massa, il piedino centrale fornisce una tensione proporzionale alla posizione della manopola.

Pulsanti - Interruttori che chiudono un circuito fintanto che vengono premuti. Si inseriscono facilmente nelle breadboard. Sono adatti per rilevare segnali acceso/spento.



Connettori a pettine - Questi piedini si infilano nei connettori femmina, come quelli sulla breadboard. Facilitano le connessioni.

Diodo - Assicura che l'elettricità scorra solo in una direzione. Utile quando nel circuito si ha un motore o un carico ad alta corrente/tensione. I diodi sono polarizzati: significa che è importante la direzione in cui sono inseriti nel circuito. Collocati in un modo, permettono il passaggio di corrente attraverso di essi. Inseriti nella direzione opposta, la bloccano. Il lato anodo si collega generalmente al punto di tensione più elevata nel circuito. Il catodo si collega al punto di minore tensione o a massa. Il catodo è solitamente contrassegnato da una striscia su un lato del componente.

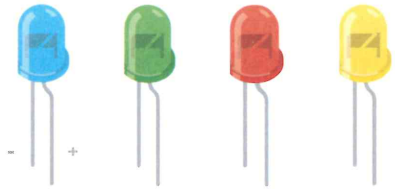


Display a cristalli liquidi (LCD) - Display alfanumerico o grafico a cristalli liquidi. Gli LCD sono disponibili in molte dimensioni, forme e stili. Nel kit ce n'è uno da 2 righe di 16 caratteri.

Filtri (rosso, verde, blu) - Filtrano diverse lunghezze d'onda della luce. Quando sono utilizzati in combinazione con le fotoresistenze, fanno reagire il sensore solo alla quantità di luce del colore filtrato.

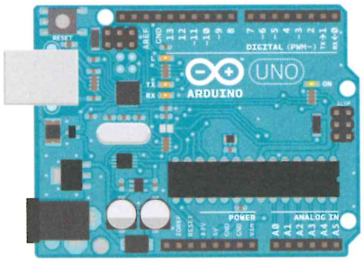
Fotoaccoppiatore - Consente di collegare due circuiti che non hanno l'alimentazione in comune. Internamente c'è un piccolo LED che, quando viene illuminato, attiva una fotocellula. Quando si applica tensione al piedino +, il LED si accende e l'interruttore interno si chiude. Le due uscite fungono da interruttore nel secondo circuito.

Fotoresistenza - (chiamata anche fotocellula). Componente la cui resistenza cambia in base alla quantità di luce che lo colpisce.

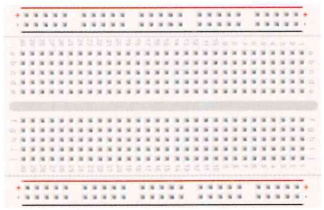


LED (Diodo Emettitore di Luce) - Un tipo di diodo che si illumina quando l'elettricità lo percorre. Come in tutti i diodi, l'elettricità scorre in una sola direzione attraverso questi componenti. Probabilmente li conosci come spie su molti dispositivi elettronici. L'anodo, che si collega all'alimentazione, è di solito il terminale più lungo, e il catodo il più corto.

PARTI NEL KIT

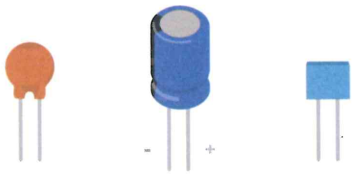


Arduino Uno - Il microcontrollore sarà al centro dei tuoi progetti. Si tratta di un semplice computer, ma uno con il quale non hai ancora avuto a che fare. Costruirai i circuiti e le interfacce per l'interazione e dirai al microcontrollore come comportarsi con gli altri componenti.

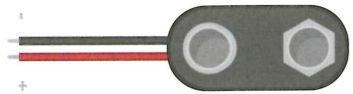


Breadboard - Una base su cui costruire i tuoi circuiti elettronici. È un pannello di connessione, con file di fori che consentono di collegare insieme cavi e componenti. Oltre il modello senza saldatura del kit, sono disponibili versioni che richiedono saldature (basette millefori).

Cavo USB - Permette di collegare Arduino Uno al computer per la programmazione. Inoltre fornisce l'alimentazione ad Arduino per la maggior parte dei progetti presenti nel kit.



Condensatori - Questi componenti immagazzinano e rilasciano energia elettrica in un circuito. Quando la tensione del circuito è superiore di quella che viene immagazzinata nel condensatore, la corrente fluisce, dando una carica al condensatore. Quando la tensione del circuito è inferiore, la carica immagazzinata viene rilasciata. Spesso sono posizionati tra l'alimentazione e la massa vicino a un sensore o a un motore per compensare le fluttuazioni di tensione.



Connettore per batteria - Collega una batteria da 9V ai cavi di alimentazione, può essere facilmente collegato a una breadboard o ad Arduino.

BENVENUTO NEL MONDO DI ARDUINO!

ARDUINO RENDE FACILE PROGRAMMARE PICCOLI COMPUTER CHIAMATI MICROCONTROLLORI, CHE SONO CIÒ CHE RENDE INTERATTIVI GLI OGGETTI

Ogni giorno ne sei circondato: sono all'interno di timer, termostati, giocattoli, telecomandi, forni a microonde e anche in alcuni spazzolini da denti. Semplicemente svolgono un compito specifico e se quasi non li si nota – come spesso avviene – è perché lo stanno facendo bene. Sono stati programmati per sentire e controllare attraverso sensori e attuatori.

I sensori ascoltano il mondo fisico. Convertono l'energia in segnali elettrici che si dà loro quando si premono pulsanti, si agitano le braccia o si grida. Pulsanti e manopole sono sensori che si toccano con le dita, ma ci sono molti altri tipi di sensori.

Gli attuatori agiscono nel mondo fisico. Convertono l'energia elettrica in energia fisica, come la luce, il calore e il movimento.

I microcontrollori ascoltano i sensori e parlano con gli attuatori. Decidono cosa fare in base al programma che scrivi.

I microcontrollori e l'elettronica sono solo lo scheletro dei tuoi progetti, però. Dovrai utilizzare competenze che probabilmente già possiedi per dare consistenza alle tue idee. Per esempio, in uno dei progetti che ti proponiamo, potrai costruire una freccia e collegarla a un motore e mettere entrambi in una scatola con una manopola per creare un indicatore con cui dire alla gente se sei occupato o no. In un altro, metterai alcune luci e un sensore di inclinazione su un cartoncino per fare una clessidra.

Arduino può rendere reattivo il tuo progetto, ma solo tu lo puoi fare bello. Ti forniremo alcuni suggerimenti su come farlo.

Arduino è stato progettato per ottenere risultati. Per realizzare questo obiettivo, abbiamo deciso di offrirti unicamente le nozioni di base su programmazione e elettronica necessarie per iniziare i progetti. Se desideri saperne di più su questi aspetti, ci sono molte buone guide disponibili. Forniremo un paio di riferimenti, altri si possono trovare online: arduino.cc/starterkit

IMPOSTAZIONI

PRIMA DI INIZIARE A CONTROLLARE IL MONDO INTORNO A TE, DEVI SCARICARE L'IDE PER PROGRAMMARE LA TUA SCHEDA

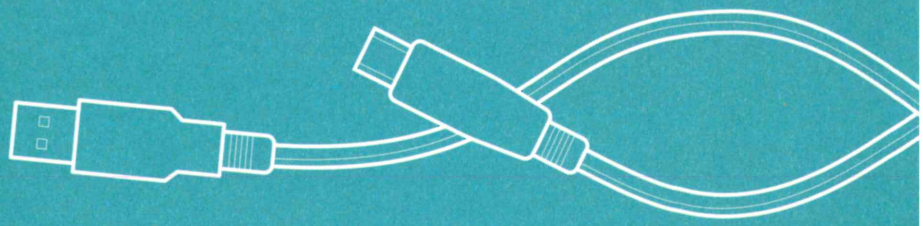
L'IDE di Arduino ti permette di scrivere programmi e caricarli in Arduino.

Scarica l'ultima versione dell'IDE da:
arduino.cc/download

Tieni la scheda Arduino e un cavo USB vicino al computer. Non collegarli ancora.

Segui le procedure indicate nelle prossime pagine per Windows, Mac OS X o Linux.

La versione online della guida è disponibile al link:
arduino.cc/guide



WINDOWS INSTALLAZIONE

Versione online
arduino.cc/windows

INSTRUZIONI PER: WINDOWS 7, VISTA, E XP

1. Terminato il download, fai doppio clic sul file "Install Arduino". Se appare una finestra di avviso di sicurezza, fai clic su "Esegui" o "Autorizza" e accetta il License Agreement. Clicca su "Next" per scegliere la cartella in cui installare l'IDE e su "Install". Alla fine del processo verranno creati i collegamenti all'IDE di Arduino sul Desktop e nel Menu Start.
2. Collega Arduino al computer con il cavo USB. La scheda viene automaticamente alimentata dalla connessione USB del computer e si accende il LED verde con la scritta ON. La scheda può essere anche alimentata da un alimentatore esterno.
3. Una volta collegata, Windows dovrebbe avviare automaticamente il processo di installazione del driver. Poiché il computer non è sempre in grado di trovare i driver autonomamente, dovrai indirizzarlo alla cartella corretta.
   Windows XP: Se ti viene chiesto di consentire a Windows Update la ricerca del software, seleziona "Si, solo in questa occasione" e poi "Installa da un elenco o percorso specifico";
   Windows Vista o 7: Se ti viene chiesto di installare il driver automaticamente o di cercarlo nel computer, su Windows 7, scegli la seconda opzione. Su Windows Vista, procedi direttamente al passaggio successivo selezionando la scelta raccomandata.
   Se l'installazione non dovesse avviarsi in automatico, clicca sul Menu Start e apri il pannello di controllo. Poi accedi alla gestione dispositivi seguendo questi percorsi:
   Windows XP: Passa alla visualizzazione classica -> Sistema -> Hardware -> Gestione Periferiche
   Windows Vista: Visualizzazione classica -> Gestione dispositivi
   Windows 7: Sistema e sicurezza -> Sistema -> Gestione dispositivi
4. Cerca il dispositivo Arduino sotto le categorie "Altre periferiche", "Altri dispositivi" o "Dispositivi sconosciuti" e, cliccando col tasto destro del mouse, seleziona "Aggiorna driver" o "Aggiornamento software driver".
5. Clicca su "Sfoglia", cerca la cartella di Arduino appena installata e seleziona la cartella "Drivers" (non la sottocartella "FTDI USB Drivers"). Premi "OK" e poi "Avanti" per continuare.
6. Windows ora installa il driver.
7. In "Gestione dispositivi", sotto la sezione "Porte (COM & LPT)", dovresti vedere una porta simile a "Arduino UNO (COM4)".

Congratulazioni! Hai installato l'IDE di Arduino nel tuo computer.

MAC OS X INSTALLAZIONE

Versione online
arduino.cc/mac

NSTRUZIONI PER:
S X 10.5 E
UCCESSIVE

1. Quando è termintao il download dell'IDE, fai doppio clic sul file .zip: espanderà l'applicazione Arduino.
2. Copia l'applicazione Arduino nella cartella delle applicazioni o dove vuoi installare il software.
3. Collega la tua Arduino al computer con il cavo USB. La scheda verrà automaticamente alimentata dalla connessione USB del computer e si accenderà il led verde con la scritta ON.
4. Non è necessario installare alcun driver per far funzionare correttamente la scheda.
5. A seconda della versione di OS X che stai usando, potrebbe aprirsi una finestra di dialogo che ti chiede di configurare le "Preferenze di sistema". Clicca sul bottone "Network" e poi su "Applica".
6. Nonostante venga visualizzata come un dispositivo "Non configurato", la Uno funzionerà ugualmente in maniera corretta. Esci dalle preferenze di sistema e...

Congratulazioni! Hai installato Arduino e sei pronto a realizzare i tuoi progetti.

LINUX INSTALLAZIONE

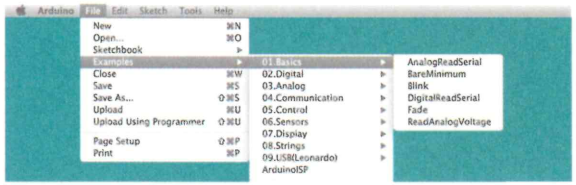
Se stai usando Linux, trovi le istruzioni a questo link:
arduino.cc/linux

COMUNICARE CON ARDUINO

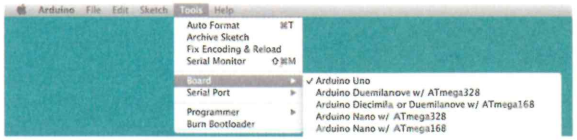
Ora che hai installato l'IDE di Arduino assicurati che il computer possa parlare con la scheda: è ora di verificare se puoi caricare un programma.

1. Fai doppio clic sull'applicazione di Arduino per aprirla. Se carichi l'IDE nella lingua sbagliata, puoi cambiarla nelle preferenze dell'applicazione. Cerca "Language Support" su questa pagina per i dettagli: ***arduino.cc/ide***

2. Vai allo sketch di esempio del LED che lampeggia ('sketch' è il modo con cui vengono chiamati i programmi di Arduino). Lo trovi sotto: ***FILE > EXAMPLES > 01.BASICS > BLINK***



3. Si dovrebbe aprire una finestra con del testo. Per ora lascia la finestra e seleziona la scheda sotto il menu: ***TOOLS > BOARD***



4. Scegli la porta seriale (COM) a cui è collegata Arduino dal menu ***TOOLS > SERIAL PORT***.

— *Su Windows.* Questa è probabilmente la COM con il numero più alto. Non succede niente se sbagli e se non funziona prova la successiva. Per scoprirlo, è possibile scollegare la scheda Arduino e riaprire il menu; la voce che scompare dovrebbe essere la scheda Arduino. Ricollega la scheda e seleziona la porta seriale.
— *Su Mac*. Dovrebbe essere qualcosa con dentro/dev/tty. usbmodem. Normalmente ce ne sono due; selezionane uno.

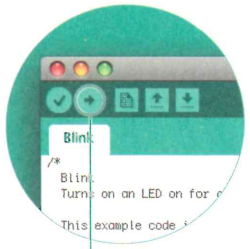


Fig. 1

5. Per caricare lo sketch Blink in Arduino, premi il pulsante ***UPLOAD*** nell'angolo in alto a sinistra della finestra. Vedi la Fig. 1.

6. Dovresti vedere una barra che indica il progresso dell'upload vicino all'angolo in basso a sinistra dell'IDE di Arduino e le luci TX e RX sulla scheda dovrebbero lampeggiare. Se l'upload è avvenuto con successo, l'IDE mostrerà il messaggio ***DONE UPLOADING***.

7. Qualche secondo dopo la fine dell'upload, dovresti vedere il LED giallo accanto alla **L** iniziare a lampaggiare. Vedi la Fig. 2.
Se funziona, congratulazioni! Hai programmato con successo l'Arduino per far lampeggiare il suo LED!

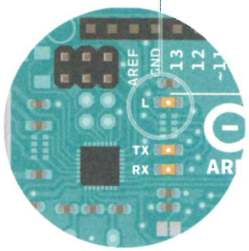


Fig. 2

Qualche volta la tua nuova Arduino è già programmata con lo sketch Blink, così non riesci a capire se sei veramente tu al comando. Se è questo il caso, cambia il tempo di `delay` mettendo 100 come numero tra parentesi e carica ancora lo sketch Blink. Ora il LED dovrebbe lampeggiare più velocemente.
Congratulazioni! Sei al comando! Ora vai al Progetto 01. (Non devi salvare nessun cambiamento che hai fatto.)

ALTRE INFORMAZIONI

Se hai problemi con i passaggi descritti in precedenza, guarda i suggerimenti di troubleshooting a questo link:
arduino.cc/trouble

Mentre ti stai preparando a costruire i tuoi progetti, puoi guardare questa pagina per ulteriori informazioni sull'ambiente di programmazione di Arduino:
arduino.cc/ide

Potresti aver voglia di guardare anche:

— gli esempi per usare vari sensori e attuatori
arduino.cc/tutorial

— la reference per il linguaggio di Arduino
arduino.cc/examples

INGREDIENTI

PULSANTE

LED

RESISTENZA DA 220 OHM

01

CONOSCI I TUOI STRUMENTI

COSTRUIRAI UN SEMPLICE CIRCUITO CON ALCUNI PULSANTI, UN LED E UNA RESISTENZA

Scopri: la teoria elettrica di base, il funzionamento di una breadboard, i componenti in serie e in parallelo

Tempo: **30 MINUTI**
Livello: ■■■■■

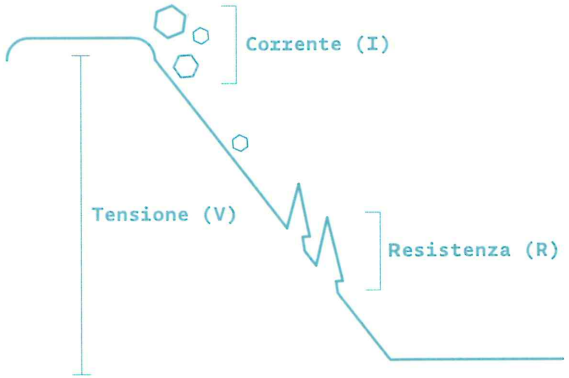
L'elettricità è un tipo di energia come il calore, la gravità o la luce. L'energia elettrica scorre lungo conduttori come i fili elettrici. Puoi convertire l'energia elettrica in altre forme di energia facendo cose interessanti, come accendere una luce o produrre dei suoni con un altoparlante.

I componenti che potresti usare per farlo, come gli altoparlanti o le lampadine, sono **trasduttori** elettrici. I trasduttori convertono gli altri tipi di energia in energia elettrica e viceversa. Ciò che converte le altre forme di energia in energia elettrica è spesso chiamato **sensore**, e ciò che converte l'energia elettrica in altre forme di energia è detto **attuatore**. Costruirai **circuiti** per far fluire l'elettricità attraverso vari componenti. I circuiti sono percorsi chiusi fatti da conduttori elettrici dotati di una fonte di energia (come una pila) e un carico che fa qualcosa di utile con l'energia.

In un circuito, l'elettricità scorre da un punto di maggior energia potenziale (solitamente chiamato alimentazione o +) a uno di minor energia potenziale. La massa (spesso rappresentata con un -) è generalmente il punto nel circuito con il potenziale elettrico più basso. Nei circuiti che costruirai, l'elettricità scorre in una sola direzione. Questo tipo di circuito si chiama a corrente continua (CC). Nei circuiti a corrente alternata (CA) l'elettricità cambia la sua direzione 50 o 60 volte al secondo (dipende dal luogo in cui vivi). Questo è il tipo di elettricità che arriva da una presa elettrica di casa.

Ci sono alcuni termini da conoscere per lavorare con i circuiti elettrici. La **corrente** (misurata in Ampere, il cui simbolo è **A**) è la quantità di carica elettrica che attraversa un punto specifico di un circuito. La **tensione** (misurata in Volt, il cui simbolo è **V**) è la differenza in energia potenziale tra un punto e l'altro di un circuito. E infine, la **resistenza** (misurata in Ohm, il cui simbolo è Ω) rappresenta quanto un componente resiste al flusso di energia elettrica.

Per immaginare il circuito elettrico puoi pensare alla caduta di una roccia da una rupe come mostrato nella Fig. 1. Più alta è la rupe, più energia avranno le rocce quando colpiranno il fondo. L'altezza della rupe è come la tensione in un circuito: maggiore è la tensione della fonte di energia, più energia dovrai usare. Più rocce hai, più energia sarà trasportata giù dalla rupe. Il numero di rocce è come la corrente in un circuito elettrico. Le rocce attraversano cespugli lungo la discesa, perdendo energia; l'energia si esaurisce scontrandosi contro i cespugli. I cespugli sono come le resistenze in un circuito: si oppongono al flusso elettrico convertendolo in altre forme di energia.



La caduta di una roccia come metafora del flusso di corrente
Fig. 1

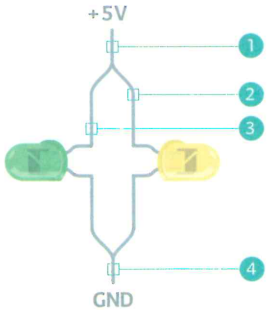
UN PAIO DI COSE SUI CIRCUITI

— In un circuito ci deve essere un percorso completo dalla fonte di energia (alimentazione) fino al punto di minor energia (massa). Se non c'è un percorso per far fluire l'energia, il circuito non funziona.

— Tutta l'energia elettrica viene utilizzata in un circuito dai suoi componenti. Ogni componente converte parte dell'energia in un'altra forma di energia. In ogni circuito, tutta la tensione viene convertita in un'altra forma di energia (luce, calore, suono, ecc).

— Il flusso di corrente in un punto specifico di un circuito è sempre lo stesso in arrivo e in uscita.

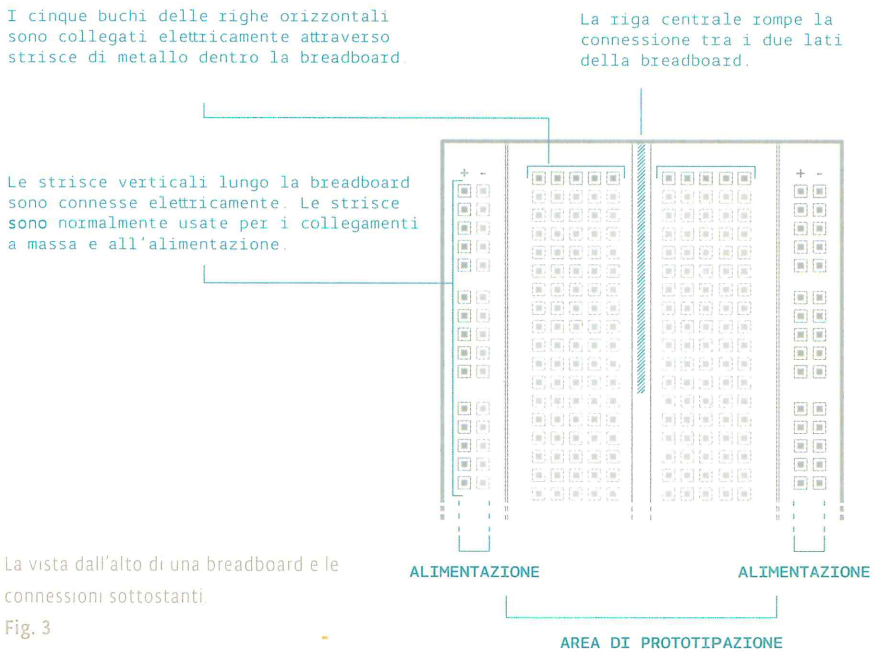
— La corrente elettrica cerca il percorso di minor resistenza verso terra. Dati due percorsi possibili, la maggior parte della corrente elettrica passa lungo il sentiero con meno resistenza. Se si ha una connessione che collega la potenza e la massa senza resistenza, si provoca un cortocircuito, e la corrente cerca di seguire questa strada. In un cortocircuito, la sorgente di alimentazione e i fili convertono l'energia elettrica in calore; possono provocare anche esplosioni. Se hai mai cortocircuitato una batteria e visto delle scintille, sai quanto possa essere pericoloso un cortocircuito.



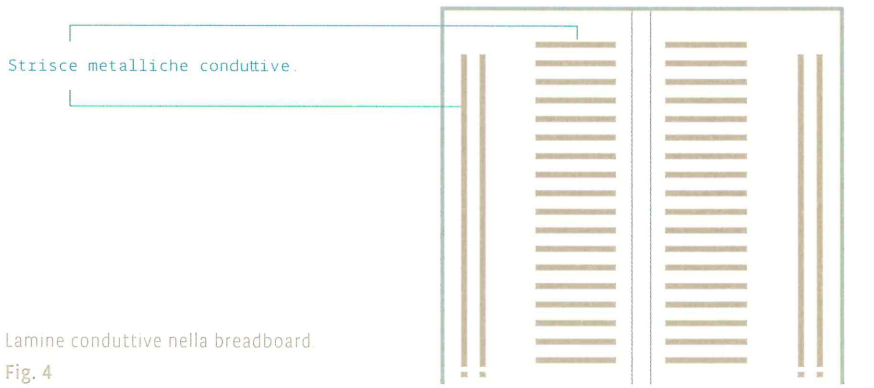
La corrente (1) = corrente (2) + corrente (3) = corrente (4).
Fig. 2

COS'È UNA BREADBOARD?

La breadboard è il luogo dove costruirai i circuiti. Quella contenuta nel kit è senza saldature, perché non c'è bisogno di saldare nulla, è una specie di LEGO in forma elettronica. Le righe orizzontali e verticali della breadboard, come mostrato in Fig. 3, portano l'elettricità attraverso sottili connettori metallici sotto la plastica forata.



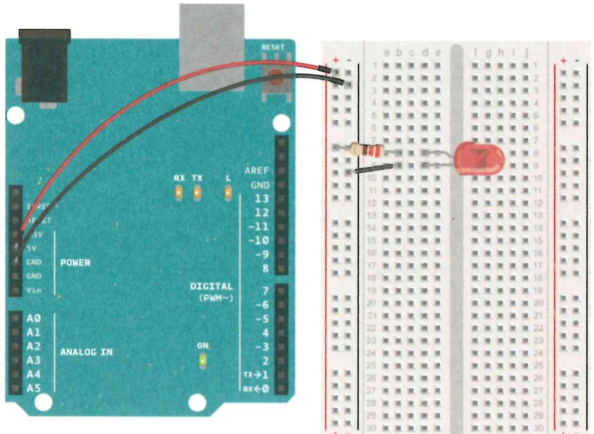
La vista dall'alto di una breadboard e le connessioni sottostanti.
Fig. 3



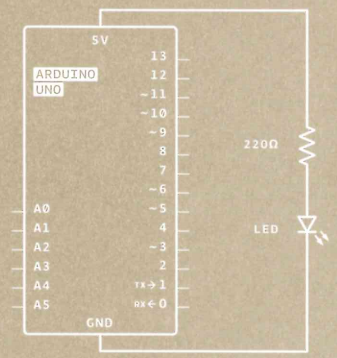
Lamine conduttive nella breadboard.
Fig. 4

SCHEMI ELETTRICI

Nel corso di questi progetti, avrai a che fare con due modi di rappresentare i circuiti: uno schema di montaggio (come nella Fig. 5), che rappresenta gli elementi del kit. L'altra è uno schema elettrico (come nella Fig. 6): un modo più astratto che mostra le relazioni tra i componenti di un circuito. Gli schemi elettrici non sempre mostrano dove sono posizionati i componenti l'uno rispetto all'altro, ma mostrano come sono collegati.

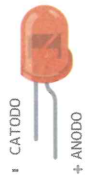


Schema di un circuito.
Fig. 5



Schema elettrico.
Fig. 6

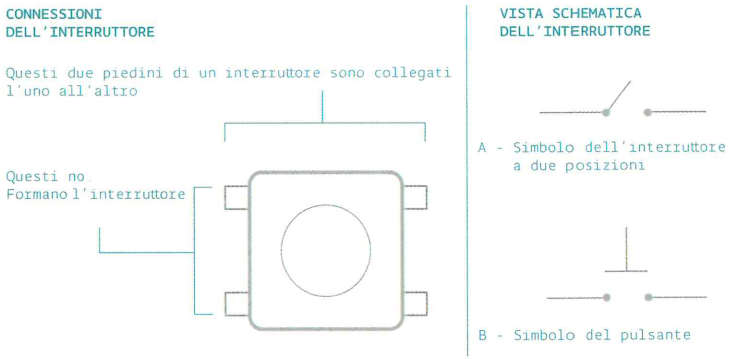
I TUOI PRIMI COMPONENTI



Un ***LED***, diodo emettitore di luce, è un componente che converte l'energia elettrica in luce. I LED sono componenti polarizzati, il che significa che permettono all'energia elettrica di fluire attraverso di loro in una sola direzione. Il piede più lungo sul LED, chiamato anodo, si collega all'alimentazione. Il piedino corto, il catodo, si collega a massa. Quando la tensione viene applicata all'anodo del LED, e il catodo è collegato a massa, il LED emette luce.

Una ***resistenza*** è un componente che resiste al flusso di energia elettrica (vedi l'elenco dei componenti per una spiegazione sulle strisce colorate sul lato). Converte parte dell'energia elettrica in calore. Se si mette una resistenza in serie con un componente come un LED, la resistenza consuma parte dell'energia elettrica e il LED riceve meno energia. Ciò consente di alimentare componenti con la quantità di energia di cui hanno bisogno. Si usa una resistenza in serie con il LED per evitare che riceva troppa tensione. Senza la resistenza, il LED è più luminoso per qualche istante, ma si brucia rapidamente.

Un ***interruttore*** interrompe il flusso di energia elettrica, spezzando il circuito quando è aperto. Quando un interruttore viene chiuso, completa il circuito. Ci sono molti tipi di interruttori. Quelli nel kit sono pulsanti perché sono chiusi solo quando è applicata una pressione.



L'interruttore.
Fig. 7

COSTRUISCI IL CIRCUITO

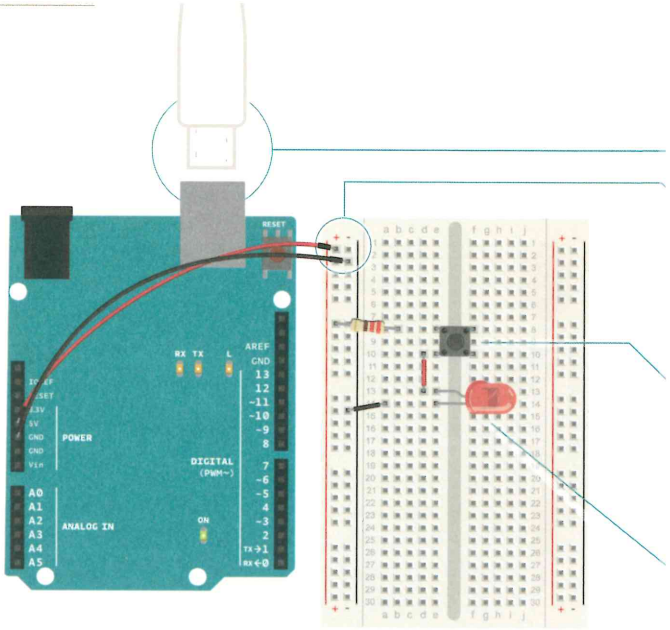


Fig. 8

Il tuo primo circuito interattivo, usando un interruttore, una resistenza e un LED. Arduino è solo la fonte di alimentazione per questo circuito; nei prossimi progetti, potrai collegare i suoi piedini di ingresso e di uscita per il controllo di circuiti più complessi.

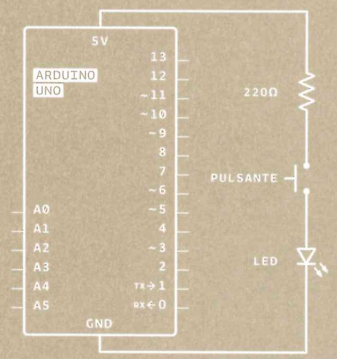


Fig. 9

Stai per utilizzare Arduino in questo progetto, ma solo come fonte di energia. Quando la colleghi a una porta USB o a una batteria da 9 volt, Arduino fornisce un'alimentazione di 5 volt tra il suo piedino 5V e il piedino GND. 5V = 5 volt: lo vedrai scritto in questo modo molte volte.

1. Se Arduino è collegato a una batteria o a un computer via USB, scollegalo prima di costruire il circuito!

2. Collega un filo rosso al piedino 5V sulla scheda Arduino e metti l'altra estremità in una delle linee lungo la tua breadboard. Collega la massa di Arduino alla linea adiacente al filo nero. È utile mantenere il colore del filo coerente (rosso per alimentazione, nero per massa) in tutto il circuito.

3. Ora che hai alimentazione sulla breadboard, monta l'interruttore a cavallo della fessura al centro della breadboard.

4. Utilizza una resistenza da 220 ohm per collegare l'alimentazione a un lato dell'interruttore. Le illustrazioni di questo libro utilizzano 4 fasce. Il kit può avere resistenze a 4 e 5 fasce. Utilizza la figura a lato per controllare quella giusta per questo progetto. Guarda pagina 41 per una spiegazione dettagliata dei codici colore delle resistenze.
   Sull'altro lato del pulsante, collega l'anodo (piedino lungo) del LED. Con un filo collega il catodo (piedino corto) del LED a massa. Quando sei pronto, collega il cavo USB ad Arduino.

USALO

Una volta che è tutto pronto, premi il pulsante. Si dovrebbe accendere il LED. Congratulazioni, hai appena costruito un circuito! Quando sarai stanco di premere il pulsante per accendere la luce, sarà il momento di movimentare le cose e aggiungere un secondo pulsante.

Metterai i componenti sulla breadboard in serie e in parallelo.
I componenti in serie sono posizionati uno dopo l'altro.
I componenti in parallelo sono uno di fianco all'altro.

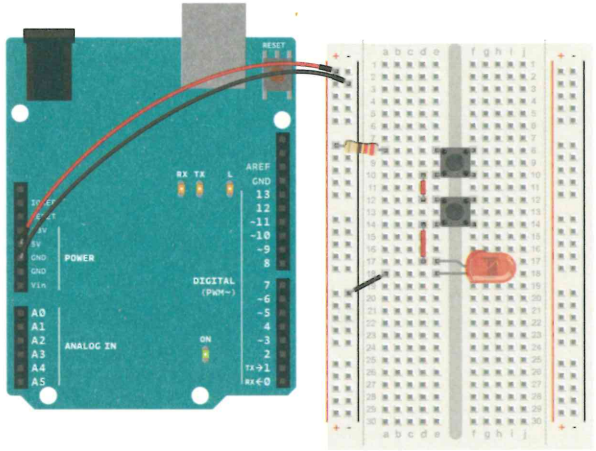
Circuito in serie

I COMPONENTI IN SERIE SONO UNO DOPO L'ALTRO

Una volta tolta l'alimentazione aggiungi un pulsante accanto all'altro già sulla breadboard. Collegali in serie come mostrato in Fig. 10. Connetti l'anodo (piedino lungo) del LED al secondo interruttore. Connetti il catodo del LED a massa. Alimenta ancora l'Arduino: ora, per accendere il LED, devi schiacciare entrambi i pulsanti. Essendo in serie, hanno bisogno di essere chiusi entrambi per completare il circuito.

Questi due elementi sono in serie

RIMUOVI SEMPRE L'ALIMENTAZIONE PRIMA DI INSERIRE QUALCOSA NEL TUO CIRCUITO



I due pulsanti sono in serie. Significa che sono percorsi dalla stessa corrrente elettrica; entrambi devono essere premuti per accendere il LED.
Fig. 10

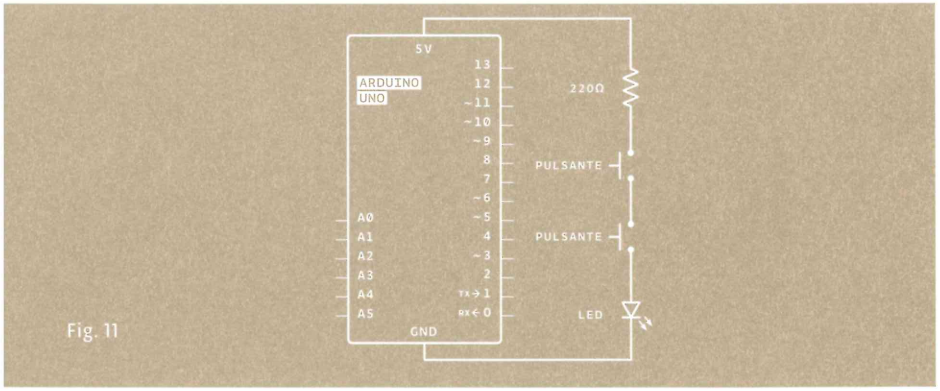
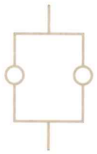


Fig. 11

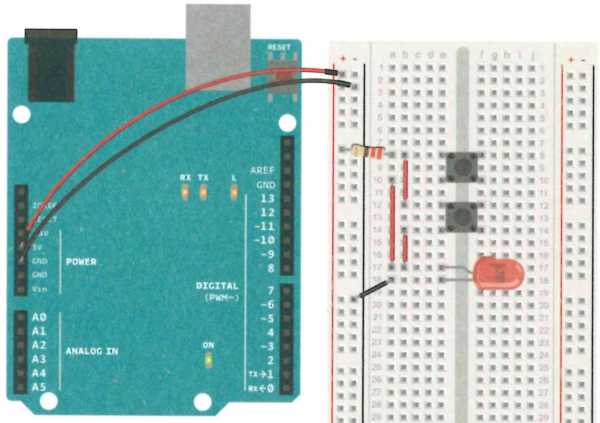


Questi due elementi sono in parallelo

Circuito in parallelo

I COMPONENTI IN PARALLELO SONO UNO DI FIANCO ALL'ALTRO

Ora che sai tutto sui circuiti in serie, è tempo di dedicarsi a quelli in parallelo. Lascia dove sono l'interruttore e il LED, ma rimuovi il filo tra i due pulsanti. Collega entrambi i pulsanti alla resistenza. Connetti le altre estremità dei pulsanti al LED, come mostrato nella Fig. 12. Ora, premendo uno dei due pulsanti, il circuito è completato e la luce si accende.



Questi due interruttori sono in parallelo. Significa che la corrente elettrica si divide tra di loro. Se uno dei pulsanti viene premuto, il LED si accende

Fig. 12

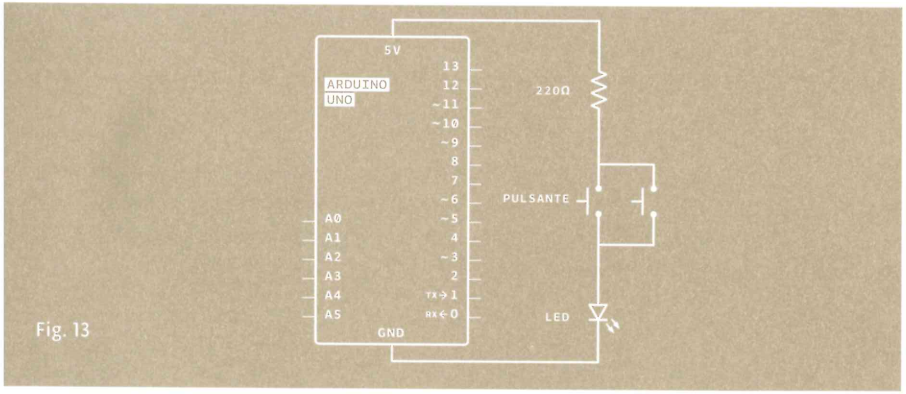
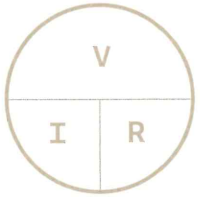


Fig. 13

CAPIRE LA LEGGE DI OHM



Puoi usare questo cerchio per ricordarti il rapporto fra tensione, corrente e resistenza. Metti il dito sopra uno dei tre, e vedi come si relaziona agli altri due.

V = I * R

I = V / R

R = V / I

Corrente, tensione e resistenza sono collegati. Cambiando uno di questi valori in un circuito, si condizionano gli altri. Il rapporto tra di loro è conosciuto come Legge di Ohm, dal nome dello scienziato che la scoprì, Georg Simon Ohm.

TENSIONE (V) = CORRENTE (I) * RESISTENZA (R)

Quando misuri l'amperaggio (corrente) in un circuito che stai costruendo, i valori sono nell'ordine dei milliampere, che sono millesimi di un ampere.

Nel circuito della Fig. 5, stai fornendo 5 volt. Il resistore ha 220 ohm di resistenza. Per trovare la corrente usata dal LED, sostituisci i valori nell'equazione. Dovresti avere 5=I*220. Dividendo entrambe le parti dell'equazione per 220, dovrebbe risultare che I = .023 cioè 23 millesimi di un ampere o 23 milliampere (23 mA) usati dal LED. Questo valore è il massimo che puoi usare in sicurezza con questi LED, ecco perché stai usando una resistenza da 220 ohm.

Puoi espandere il tuo progetto in molti modi, sia creando il tuo pulsante (due pezzi di alluminio con del filo funzionano bene), sia creando una combinazione di pulsanti e LED in parallelo e in serie. Cosa succede quando metti tre o quattro LED in serie? Cosa succede quando sono in parallelo? Perché si comportano in questo modo?

Un ***multimetro*** è uno strumento che misura la quantità di resistenza, corrente e tensione nel circuito. Sebbene non necessario in questi progetti, rappresenta uno strumento utile. Online trovi una buona descrizione su come usarlo all'indirizzo ***arduino.cc/multimeter***

Hai imparato le caratteristiche elettriche della tensione, della corrente e della resistenza costruendo un circuito sulla breadboard. Con alcuni componenti come i LED, le resistenze e i pulsanti, puoi creare semplici sistemi interattivi: se si preme un pulsante, la luce si accende. A questi fondamenti dell'elettronica si farà riferimento – e saranno ampliati – nei prossimi progetti.

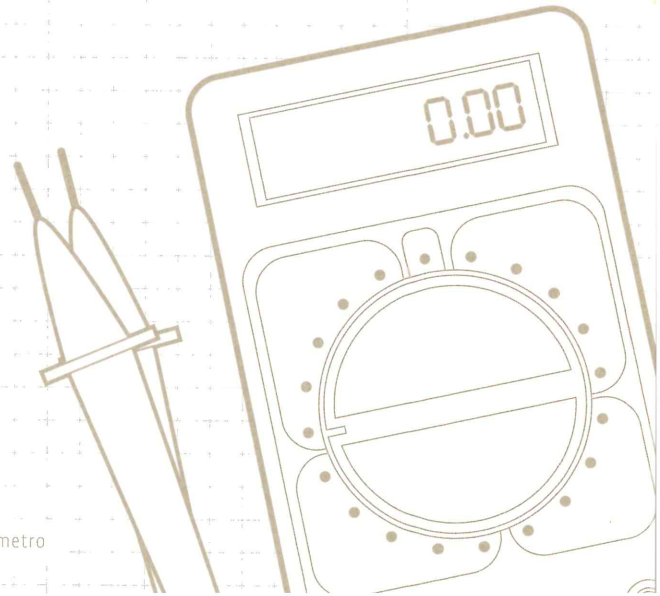


Fig. 14 - Un multimetro

INGREDIENTI

PULSANTE

LED

RESISTENZA DA 220 OHM

RESISTENZA DA 10 KILO OHM

02

INTERFACCIA PER ASTRONAVE

ARDUINO STA PER DIVENTARE IL PROTAGONISTA DI UN FILM DI FANTASCIENZA

Scopri: input e output digitali, il tuo primo programma, le variabili

Tempo: **45 MINUTI**
Livello: ■□□□□

Basato sul progetto: **1**

Ora che hai acquisito le basi dell'elettronica, è tempo di passare a controllare cose e oggetti con Arduino. In questo progetto, costruirai qualcosa che potrebbe essere un'interfaccia di una astronave di un film di fantascienza degli anni Settanta. Realizzerai un pannello di controllo con un pulsante e delle luci che si accenderanno quando schiaccerai il pulsante. Puoi decidere se le luci significano "Inserire l'hyperdrive" o "Spara il laser!". Un LED verde sarà acceso finché non premerai un pulsante. Quando Arduino riceverà un segnale da un pulsante, la luce verde si spegnerà e le altre due luci inizieranno a lampeggiare.

I piedini digitali di Arduino possono leggere solo due stati: quando c'è o non c'è la tensione su un piedino di ingresso. Questo tipo di ingresso è chiamato digitale (o qualche volta binario, per via dei due stati). Questi stati vengono definiti come ALTO (`HIGH`) e BASSO (`LOW`). `HIGH` è come dire "c'è tensione qui!" e `LOW` significa "non c'è tensione su questo piedino!". Quando metti un piedino di `OUTPUT` a `HIGH`, usando un comando chiamatao `digitalWrite()`, lo stai accendendo. Se misuri la tensione tra il piedino e la massa, ottieni 5 volt. Quando metti un piedino di `OUTPUT` a `LOW`, lo stai spegnendo. Il piedino digitale di Arduino funziona sia come ingresso sia come uscita. Nel tuo codice, dovrai configurarli a seconda della funzione che vorrai dare loro. Quando i piedini sono output (uscita), puoi accendere componenti come i LED. Se configuri i piedini come input (ingresso), puoi verificare se è stato premuto o no un pulsante. Dato che i piedini 0 e 1 sono usati per comunicare con il computer, è meglio iniziare con il piedino 2.

COSTRUISCI IL CIRCUITO

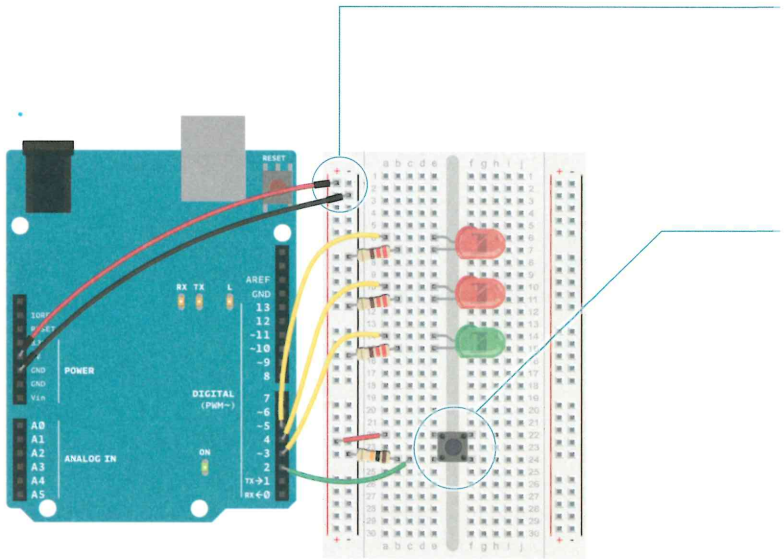


Fig. 1

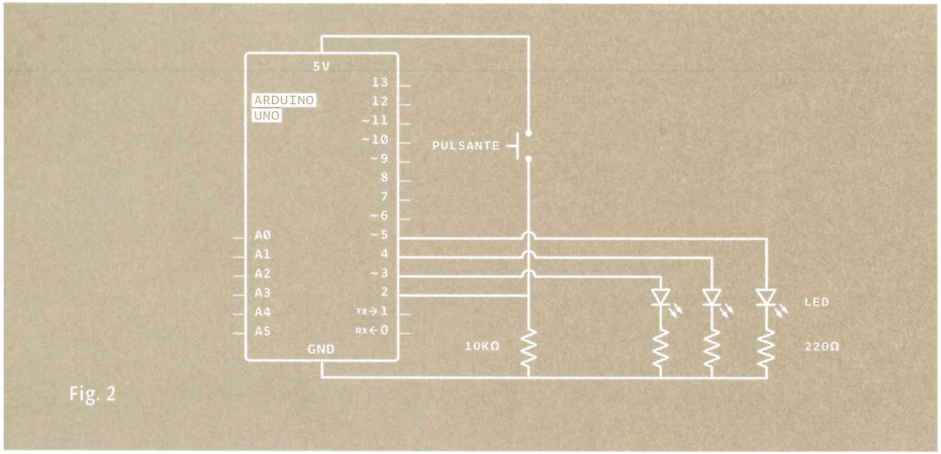
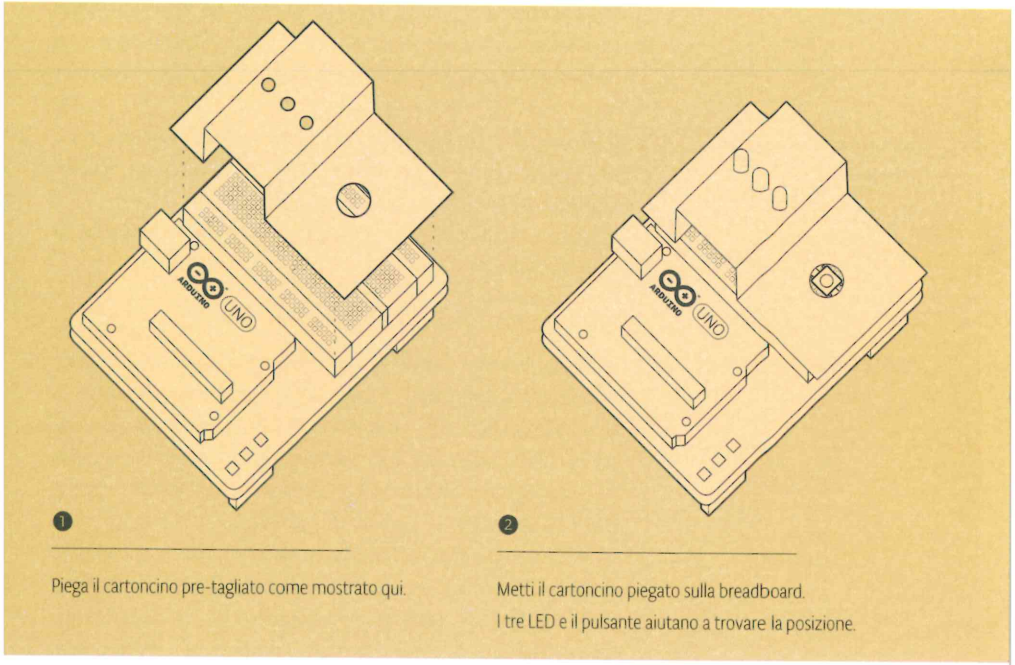


Fig. 2

1. Collega la breadboard alle connessioni 5V e GND (massa) di Arduino, come nel progetto precedente. Posiziona due LED rossi e uno verde sulla breadboard. Connetti il catodo (piedino corto) di ciascun LED a massa attraverso una resistenza da 220 ohm. Connetti l'anodo (piedino lungo) del LED verde al piedino 3. Connetti gli anodi dei LED rossi rispettivamente ai piedini 4 e 5.

2. Posiziona l'interruttore sulla breadboard come nel progetto precedente. Connetti un lato all'alimentazione e l'altro lato al piedino digitale 2 di Arduino. Hai anche bisogno di aggiungere una resistenza da 10 kilo ohm da massa al piedino dell'interruttore che è collegato ad Arduino. Questa resistenza di 'pull-down' connette il piedino a massa quando il pulsante è aperto, così si legge `LOW` quando nessuna tensione proviene dal pulsante.

Puoi coprire la breadboard con il cartoncino del kit. O puoi decorarlo per realizzare il tuo personale sistema di lancio. Se le luci si accendono e si spengono non significa nulla di per sé, ma, quando le metti in un pannello di controllo e dai loro un senso, acquistano un significato maggiore. Cosa vuoi che significhi il LED verde? E il lampeggio del LED rosso? Sei libero di deciderlo!



1. Piega il cartoncino pre-tagliato come mostrato qui.

2. Metti il cartoncino piegato sulla breadboard. I tre LED e il pulsante aiutano a trovare la posizione.

IL CODICE

Alcune note prima di iniziare

Ogni programma Arduino ha due funzioni principali. Le funzioni sono parti di un programma del computer che eseguono comandi specifici. Le funzioni hanno nomi univoci e sono 'chiamate' quando servono. Le funzioni necessarie in un programma Arduino sono chiamate `setup()` e `loop()`. Queste funzioni devono essere dichiarate: significa che si deve dire ad Arduino cosa faranno. `setup()` e `loop()` sono indicate come vedi sulla destra.
In questo programma, stai per creare una variabile prima di entrare nella parte principale del programma. Le variabili sono nomi che dai a delle aree di memoria di Arduino così puoi tenere traccia di ciò che sta accadendo. Questi valori possono cambiare a seconda delle istruzioni del tuo programma. I nomi delle variabili dovrebbero essere descrittivi dei valori che stanno immagazzinando. Per esempio, una variabile chiamata `switchState` spiega cosa sta immagazzinando: lo stato di un pulsante. Invece, una variabile chiamata '`x`' non spiega nulla su ciò che contiene.

Iniziamo a programmare

Per creare una variabile, hai bisogno di dichiararne il *tipo*. Il tipo di dato `int` contiene un numero intero (chiamato anche un intero); che è un numero qualsiasi senza punti decimali. Quando dichiari una variabile, normalmente gli dai anche un valore iniziale. La dichiarazione della variabile deve finire con un punto e virgola (;).

Configura la funzionalità dei piedini

Il `setup()` viene eseguito una volta soltanto all'accensione dell'Arduino. Qui è dove configuri i piedini digitali per renderli ingressi o uscite usando la funzione `pinMode()`.
I piedini connessi ai LED sono uscite (OUTPUT) e il pulsante un ingresso (INPUT).

Crea la funzione loop

Il `loop()` viene eseguito continuamente dopo che è stato completato il `setup()`. Il `loop()` è il posto dove verifichi la tensione sugli ingressi e controlli le uscite. Per controllare la tensione su un input digitale, usa la funzione `digitalRead()` che rileva se c'è tensione sul piedino scelto. Per sapere quale piedino leggere, `digitalRead()` si aspetta un *parametro*. I parametri sono informazioni che dai alle funzioni dicendo loro come dovrebbero svolgere il loro lavoro. Per esempio, `digitalRead()` ha bisogno di un parametro: quale piedino leggere. Nel tuo programma, `digitalRead()` sta verificando lo stato del piedino 2 e

```
void setup(){
}

void loop(){
}
```

{ Parentesi graffe }

Il codice scritto all'interno di parentesi graffe è eseguito quando viene chiamata la funzione.

```
int switchState = 0;
```

```
void setup(){
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(2,INPUT);
}
```

Maiuscole, minuscole

Nel codice stai attento a distinguere le maiuscole e le minuscole. Per esempio, `pinMode` è il nome di un comando, ma pinmode produce un errore.

```
void loop(){
  switchState = digitalRead(2);
  // questo è un commento
```

Commenti

I commenti sono note che scrivi per te stesso, che il computer ignora. Per scrivere un commento, aggiungi due barre //. Il computer ignora tutto il contenuto presente dopo le due barre.

immagazzinando il valore nella variabile `switchState`. Se c'è tensione sul piedino quando viene chiamata `digitalRead()`, la variabile switchState ha il valore `HIGH` (o 1). Se non c'è tensione sul piedino, switchState prende il valore `LOW` (o 0).

L'istruzione if

Un comando `if()` nella programmazione confronta due cose e determina se il confronto è vero o falso ed esegue le azioni che gli indichi. Quando confronti due cose nella programmazione, devi usare due simboli dell'uguale ==. Se usi un segno solo, imposti il valore della variabile invece di confrontarlo.

Costruisci la tua astronave

`digitalWrite()` è il comando che ti permette di impostare un'uscita a 5V o 0V. `digitalWrite()` richiede due parametri: quale piedino controllare e a quale valore impostare il piedino, `HIGH` o `LOW`. Se vuoi accendere i LED rossi e spegnere il LED verde nella tua istruzione `if()`, il tuo codice sarà così.

Se ora esegui il tuo programma, le luci cambiano quando premi il pulsante. Questo è piuttosto chiaro, ma puoi aggiungere un tocco di complessità al programma per una reazione più interessante.

Hai detto ad Arduino cosa fare quando l'interruttore è aperto. Ora definisci cosa accade quando è chiuso. L'istruzione `if()` ha un componente opzionale `else` (in italiano oppure) che fa succedere qualcosa se la condizione originale non si è verificata. In questo caso, dato che hai verificato che il pulsante è `LOW`, scrivi il codice per la condizione `HIGH` dopo l'istruzione `else`.

Per far lampeggiare i LED rossi quando è premuto il pulsante, devi accendere e spegnere le luci nell'istruzione `else` che hai appena scritto.

Ora il tuo programma fa lampeggiare il LED rosso quando il pulsante è premuto.

Dopo aver impostato i LED in un determinato stato, devi mettere in pausa l'Arduino prima di riportarli allo stato precedente. Se non aspetti, le luci vanno avanti e indietro così velocemente che appaiono leggermente meno luminose ma non accese o spente. Questo perché l'Arduino esegue il suo `loop()` migliaia di volte ogni secondo e il LED si accende e spegne più velocemente di quanto riusciamo a percepirlo. La funzione `delay()` impedisce ad Arduino di eseguire istruzioni per un certo periodo di tempo. `delay()` richiede un parametro che determina il numero di millisecondi prima di eseguire le istruzioni successive. Ci sono 1000 millisecondi in un secondo. `delay(250)` fa pausa per un quarto di secondo.

```
11   if (switchState == LOW) {
12   // il pulsante non è premuto
13     digitalWrite(3, HIGH); // LED verde
14     digitalWrite(4, LOW);  // LED rosso
15     digitalWrite(5, LOW);  // LED rosso
16   }
17   else {  // il pulsante è premuto
18     digitalWrite(3, LOW);
19     digitalWrite(4, LOW);
20     digitalWrite(5, HIGH);
21     delay(250);  // aspetta un quarto di secondo
22     // cambia gli stati dei LED
23     digitalWrite(4, HIGH);
24     digitalWrite(5, LOW);
25     delay(250); // aspetta un quarto di secondo
26   }
27 } // torna indietro all'inizio del loop
```

Può essere utile trascrivere il flusso del programma in pseudocodice: un modo per descrivere cosa vuoi che il programma faccia in un linguaggio semplice, ma strutturato in modo che renda facile scrivere un vero programma. In questo caso stai determinando se `switchState` è `HIGH` (significa che il pulsante è premuto) o no. Se il pulsante è premuto, spegnerai il LED verde e accenderai quello rosso. Nel pseudocodice, il comando potrebbe essere così

```
se switchState è LOW:
  accendi il LED verde
  spegni il LED rosso

se switchState è HIGH:
  spegni il LED verde
  accendi il LED rosso
```

USALA

Una volta che la tua Arduino è programmata, dovresti vedere accendersi la luce verde. Quando premi il pulsante, le luci rosse iniziano a lampeggiare e la luce verde si spegne. Prova a cambiare il tempo delle due funzioni `delay()`; nota cosa accade alle luci e come la risposta del sistema cambia in base alla velocità del lampeggio. Quando esegui un `delay()` nel programma, questo ferma tutte le altre funzionalità. Non avviene nessuna lettura di sensori fino a che questo periodo non è passato. Sebbene i ritardi siano spesso utili, quando realizzi il tuo progetto assicurati che non interferiscano inutilmente con l'interfaccia.

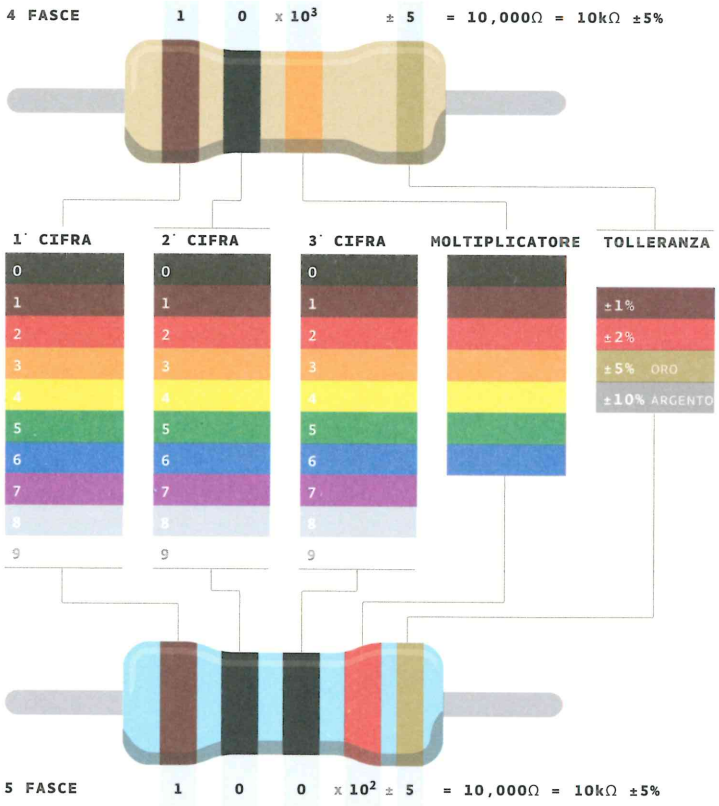
Come potresti far lampeggiare i LED rossi quando si avvia il programma? Come potresti creare un'interfaccia più grande o più complessa con LED e pulsanti per le tue avventure interstellari?

Quando inizi a creare un'interfaccia per il tuo progetto, pensa alle aspettative delle persone che lo useranno. Quando premono un pulsante, vorranno un feedback immediato? Dovrebbe esserci un ritardo tra le loro azioni e ciò che fa Arduino? Prova a metterti nei panni di un altro quando stai progettando e verifica che le tue aspettative siano adeguate alla realtà del progetto.

In questo progetto, hai creato il tuo primo programma con Arduino per controllare con un pulsante il comportamento di alcuni LED. Hai usato delle variabili, un'istruzione if()..., e funzioni per leggere lo stato di un ingresso e controllare delle uscite.

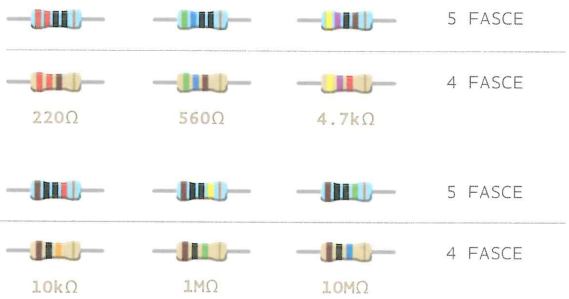
COME LEGGERE IL CODICE COLORI DELLE RESISTENZE

I valori delle resistenze sono espressi da anelli colorati secondo una norma sviluppata negli anni Venti quando era troppo complesso scrivere numeri su oggetti tanto piccoli. Ogni colore corrisponde a un numero, come vedi nella tabella sotto. Ogni resistenza ha 4 o 5 anelli. Nel tipo con 4 anelli, i primi due anelli indicano le prime due cifre del valore mentre il terzo indica il numero degli zero seguenti (tecnicamente rappresenta la potenza di dieci). L'ultimo anello specifica la tolleranza: nell'esempio qui sotto, l'oro indica che il valore della resistenza può essere 10 kilo ohm più o meno 5%.



RESISTENZE CONTENUTE NELLO STARTER KIT

Troverai versioni con 4 e 5 fasce.



03

LED

RESISTENZA DA 220 OHM

SENSORE DI TEMPERATURA

INGREDIENTI

AMOROMETRO

TRASFORMA ARDUINO IN UNA MACCHINA DELL'AMORE. USANDO UN INGRESSO ANALOGICO, POTRAI MISURARE QUANTO SEI CALDO VERAMENTE!

Scopri: ingresso analogico, monitor seriale

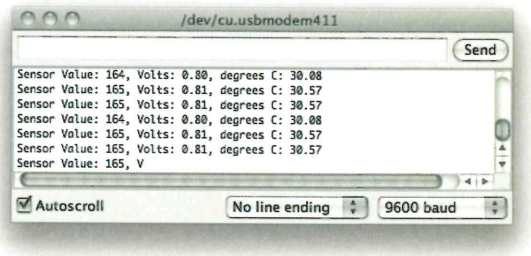
Tempo: **45 MINUTI**
Livello: ■□□□□

Basato sui progetti: **1, 2**

Pulsanti e interruttori sono una gran cosa, ma il mondo fisico è molto di più che non acceso e spento. Anche se Arduino è uno strumento digitale, può leggere informazioni da sensori analogici per misurare grandezze come la temperatura o la luce. Per farlo, utilizzerai il convertitore analogico-digitale (ADC) integrato nell'Arduino. I piedini di ingresso analogico A0-A5 restituiscono un valore compreso tra 0 e 1023 quando leggono delle tensioni comprese tra 0 e 5 volt.

Userai un ***sensore di temperatura*** per misurare quanto è calda la tua pelle. Questo componente produce una tensione che varia in base alla temperatura percepita. Ha tre piedini: uno che si connette a massa, uno che si connette all'alimentazione e il terzo che produce una tensione variabile per Arduino. Nello sketch di questo progetto, leggerai il valore del sensore e lo userai per accendere e spegnere i LED, indicando quanto sei caldo. Esistono molti modelli di sensori di temperatura. Questo, il TMP36, è comodo perché produce una tensione in uscita che è direttamente proporzionale alla temperatura in gradi Celsius.

L'ambiente di sviluppo integrato (IDE) di Arduino è dotato di uno strumento chiamato ***monitor seriale*** che ti consente di visualizzare dati provenienti dal microcontrollore. Usando il monitor seriale, è possibile ottenere informazioni sullo stato dei sensori e avere un'idea di ciò che sta accadendo nel circuito e nel codice.



Monitor seriale.
Fig. 1

COSTRUISCI IL CIRCUITO

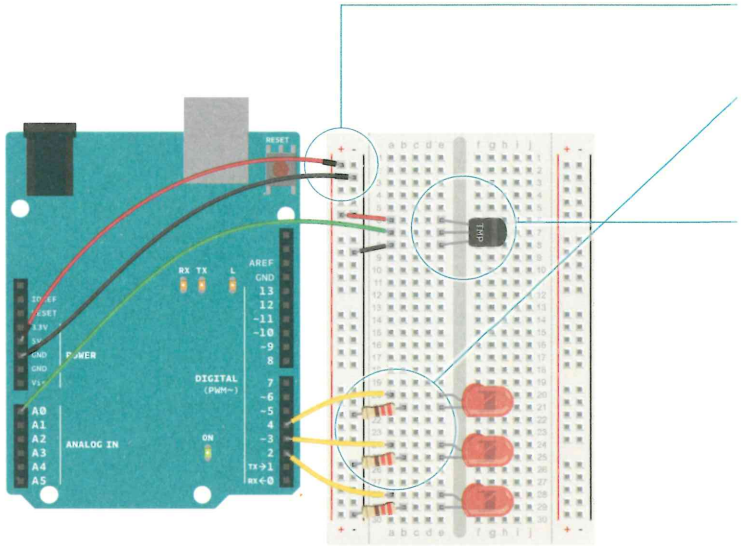


Fig. 2

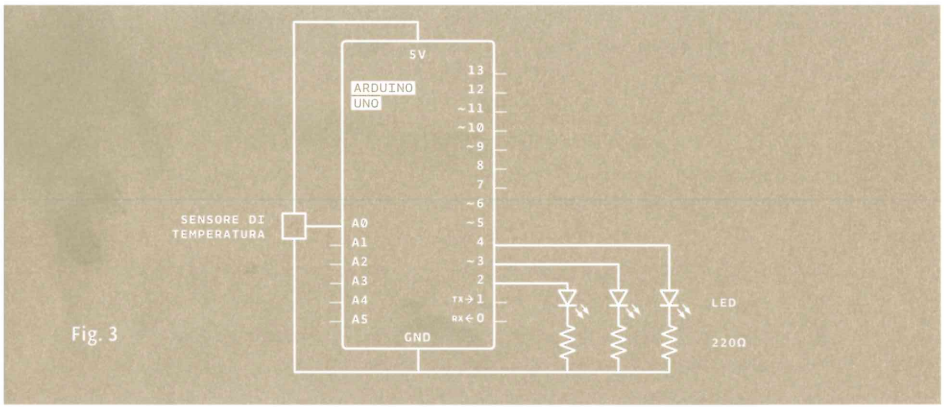


Fig. 3

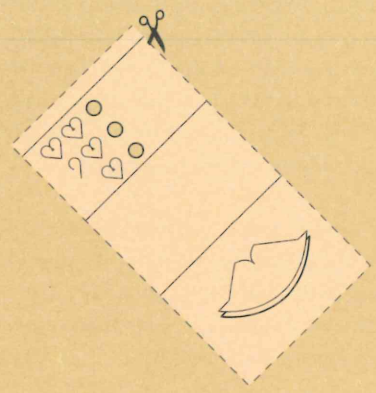
In questo progetto, prima di procedere devi misurare la temperatura della stanza. Lo si può fare anche attraverso calibrazione. È possibile usare un pulsante per impostare la temperatura di riferimento o prendendo un campione con Arduino prima di iniziare il `loop()` e usarlo come punto di riferimento. Il Progetto 06 scenderà nei dettagli di questo tema; puoi intanto guardare l'esempio di calibrazione che viene dato con il software di Arduino: *arduino.cc/calibration*

1. Proprio come hai fatto nei progetti precedenti, collega la tua breadboard in modo da avere alimentazione e massa.

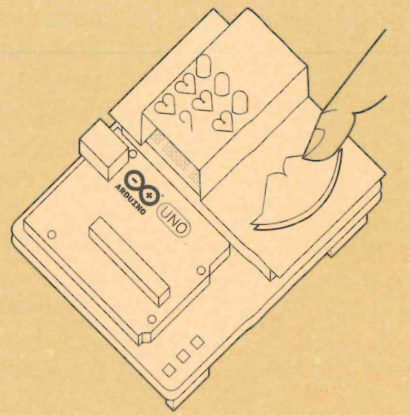
2. Collega a massa il catodo (il piedino corto) di ogni LED che stai usando attraverso una resistenza da 220 ohm. Collega gli anodi dei LED ai piedini dal 2 al 4. Questi saranno gli indicatori per il progetto.

3. Posiziona il TMP36 sulla breadboard con la parte piatta verso Arduino (l'ordine dei piedini è importante!) come mostrato in Fig. 2. Collega il piedino di sinistra della parte piatta all'alimentazione e il piedino destro a massa. Collega il piedino centrale al piedino A0 su Arduino. Questo è il piedino di ingresso analogico 0.

Crea un'interfaccia per il tuo sensore che permetta alle persone di interagirvi. Un ritaglio di carta a forma di mano è un buon indicatore. Se ti senti fortunato, crea un paio di labbra da baciare, guarda come si accendono le luci! Potresti etichettare i LED per dar loro un significato. Un LED potrebbe significare che sei un asociale, due LED che sei caldo e amichevole e tre che sei troppo caldo da gestire!



1. Ritaglia un pezzo di carta da posizionare sulla breadboard. Disegna delle labbra dove sarà messo il sensore e taglia alcuni cerchi per farci passare i LED.



2. Posiziona il ritaglio sulla breadboard in modo che le labbra coprano il sensore e che i LED fuoriescano dai buchi. Premi le labbra per vedere quanto sei caldo!

IL CODICE

Un paio di costanti utili

Le costanti sono simili alle variabili: ti permettono di dare un nome univoco agli elementi presenti nel programma, ma, al contrario delle variabili, non possono cambiare. Dai un nome all'ingresso analogico per riconoscerlo facilmente e crea un'altra costante per la temperatura di base. Per ogni 2 gradi al di sopra di questo riferimento, si accende un LED. Hai già visto i tipi di dati int, usali qui per identificare su quale piedino è collegato il sensore. La temperatura è immagazzinata come un *float*, cioè un numero decimale.

Avvia la porta seriale alla velocità desiderata

Nel `setup()` usa un nuovo comando, `Serial.begin()`, che apre una connessione tra l'Arduino e il computer, così puoi vedere il valore proveniente dall'ingresso analogico sullo schermo del computer.
Il parametro `9600` è la velocità alla quale comunica l'Arduino: 9600 bit al secondo. Usa il monitor seriale dell'IDE di Arduino per visualizzare le informazioni che invii dal tuo microcontrollore. Quando apri il monitor seriale dell'IDE verifica che la velocità di trasmissione sia 9600.

Imposta le direzioni dei piedini digitali e spegnili

Il prossimo è un ciclo `for()` per impostare alcuni piedini come uscite. Sono i piedini che in precedenza hai collegato ai LED. Invece di dare loro nomi unici e digitare la funzione `pinMode()` per ognuno di essi, puoi usare un ciclo `for()` per impostarli velocemente. Questo è un trucco comodo se hai un gran numero di cose simili che vuoi ripetere in un programma. Di' al ciclo `for()` di andare dal piedino 2 al 4 in modo sequenziale.

Leggi il sensore di temperatura

Nel `loop()`, usa una variabile locale chiamata `sensorVal` per immagazzinare le letture del sensore. Per ottenere il valore dal sensore, chiama `analogRead()` che richiede come parametro il numero del piedino dove leggere la tensione. Il valore, compreso tra 0 e 1023, rappresenta la tensione sul piedino.

Invia i valori del sensore di temperatura al computer

La funzione `Serial.print()` manda informazioni dall'Arduino al computer collegato. Puoi vedere questa informazione con il monitor seriale. Se dai al `Serial.print()` come parametro un testo tra virgolette, questo invia il testo che hai digitato. Se dai una variabile come parametro, invia il valore di quella variabile.

```
1 const int sensorPin = A0;
2 const float baselineTemp = 20.0;
```

```
3 void setup(){
4   Serial.begin(9600); // apri una porta seriale
```

```
5   for(int pinNumber = 2; pinNumber<5; pinNumber++){
6     pinMode(pinNumber,OUTPUT);
7     digitalWrite(pinNumber, LOW);
8   }
9 }
```

tutorial per il ciclo for()

arduino.cc/for

```
10 void loop(){
11   int sensorVal = analogRead(sensorPin);
```

```
12   Serial.print("Sensor Value: ");
13   Serial.print(sensorVal);
```

Converti la lettura del sensore in tensione

Anche con poche conoscenze di matematica, è possibile determinare la reale tensione sul piedino. La tensione è tra 0 e 5 volt e potrebbe essere decimale (per esempio, potrebbe essere 2,5 volt), perciò hai bisogno di immagazzinarla in un **float**. Crea una variabile chiamata `voltage` per memorizzare questo numero. Dividi `sensorVal` per 1024.0 e moltiplicalo per 5.0. Il nuovo numero rappresenta la tensione sul piedino.

Come con il valore del sensore, invialo al monitor seriale.

Converti la tensione in temperatura e manda il valore al computer

Se esamini la documentazione (*datasheet*) del sensore, trovi informazioni sui valori della tensione in uscita. I datasheet sono come manuali per i componenti elettronici. Sono stati scritti da ingegneri per altri ingegneri. Il datasheet di questo sensore spiega che una variazione di 10 millivolt del sensore è equivalente a un cambio di temperatura di 1 grado Celsius. Indica anche che il sensore può leggere temperature sotto gli 0 gradi. Per questo, devi creare una compensazione (*offset*) per i valori sotto gli 0 gradi. Se prendi la tensione, sottrai 0.5 e moltiplichi per 100, hai la temperatura accurata in gradi Celsius. Memorizza questo numero in una variabile **float** chiamata `temperature`.

Ora che hai la temperatura vera, manda il valore al monitor seriale. Visto che la variabile `temperature` è l'ultima cosa che stampi in questo ciclo, puoi usare un comando leggermente differente: `Serial.println()`.
Questo comando manda a capo il monitor seriale dopo l'invio del valore, aiutando a rendere più leggibili i dati visualizzati.

Spegni i LED con una temperatura bassa

Con la temperatura vera puoi usare un'istruzione `if()...else` per accendere i LED. Usando la temperatura di riferimento come punto di partenza, accendi un LED per ogni aumento di temperatura di 2 gradi. Cerca diversi intervalli di valori mentre ti muovi sulla scala delle temperature.

```
14  // converti la lettura ADC in tensione
15  float voltage = (sensorVal/1024.0) * 5.0;
```

```
16  Serial.print(", Volts: ");
17  Serial.print(voltage);
```

```
18  Serial.print(", degrees C: ");
19  // converti la tensione in temperatura
20  float temperature = (voltage - .5) * 100;
21  Serial.println(temperature);
```

```
22  if(temperature < baselineTemp){
23    digitalWrite(2, LOW);
24    digitalWrite(3, LOW);
25    digitalWrite(4, LOW);
```

Starter Kit
dei datasheet
arduino.cc/kitdatasheets

Accendi un LED a una bassa temperatura

L'operatore && significa "`e`" (and), in senso logico. Puoi verificare più condizioni come se la temperatura è di 2 gradi più alta del riferimento e meno di 4 gradi sopra di essa.

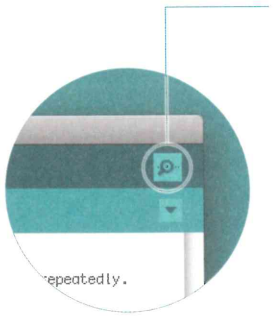
Accendi due LED a una temperatura media

Se la temperatura è tra 2 e 4 gradi sopra il riferimento, questo blocco di codice accende anche il LED sul piedino 3.

Accendi tre LED a un'alta temperatura

Il convertitore digitale-analogico non può leggere velocissimamente così dovresti inserire un leggero ritardo alla fine del tuo `loop()`. Se lo leggi troppo di frequente, i tuoi valori appaiono irregolari.

USALO



Con il codice caricato su Arduino, clicca l'icona del monitor seriale. Dovresti vedere un flusso di valori così formattati: `Sensor: 200, Volts: .70, degrees C: 17`

Metti le dita intorno al sensore quando è collegato alla breadboard e guarda cosa accade ai valori nel monitor seriale. Prendi nota della temperatura quando il sensore è lasciato all'aria aperta.

Chiudi il monitor seriale e imposta la costante del riferimento di temperatura al valore della temperatura ambiente. Carica ancora il codice e prova a tenere il sensore tra le dita. Mentre la temperatura aumenta, dovresti vedere accendersi i LED uno a uno. Congratulazioni!

```
26    }else if(temperature >= baselineTemp+2 &&
        temperature < baselineTemp+4){
27      digitalWrite(2, HIGH);
28      digitalWrite(3, LOW);
29      digitalWrite(4, LOW);

30    }else if(temperature >= baselineTemp+4 &&
        temperature < baselineTemp+6){
31      digitalWrite(2, HIGH);
32      digitalWrite(3, HIGH);
33      digitalWrite(4, LOW);

34    }else if(temperature >= baselineTemp+6){
35      digitalWrite(2, HIGH);
36      digitalWrite(3, HIGH);
37      digitalWrite(4, HIGH);

38    }
39    delay(1);
40  }
```

Crea un'interfaccia per due persone per testare la loro compatibilità reciproca. Decidi cosa significa compatibilità e come la leggerai. Forse si sono tenuti la mano e hanno generato calore? Forse si sono abbracciati? Cosa ne pensi?

Espandendo i tipi di ingressi che puoi leggere, hai usato analogRead() e il monitor seriale per tenere traccia dei cambiamenti dentro Arduino. Ora è possibile leggere un gran numero di sensori analogici.

04

LED

RESISTENZA DA 220 OHM

RESISTENZA DA 10 KILO OHM

FOTORESISTENZA

FILTRO

INGREDIENTI

LAMPADA MISCELA COLORI

USANDO UN LED A TRE COLORI E TRE FOTORESISTENZE, POTRAI CREARE UNA LAMPADA CHE CAMBIA COLORE IN BASE ALLE CONDIZIONI ESTERNE DI LUCE

Scopri: uscite analogiche, mappare valori

Tempo: **45 MINUTI**
Livello: ■■□□□

Basato sui progetti: **1, 2, 3**

Far lampeggiare i LED può essere divertente, ma che ne pensi di farli sfumare o miscelarne i colori? Ti potresti aspettare che si tratti solo di fornire meno tensione a un LED per fargli cambiare luminosità.

Arduino non può variare la tensione d'uscita dei suoi piedini che può essere solo di 5V. Quindi avrai bisogno di una tecnica chiamata modulazione di larghezza di impulso – in inglese ***Pulse Width Modulation (PWM)*** – per attenuare i LED. La modulazione di larghezza di impulso accende e spegne rapidamente i piedini di uscita in un determinato intervallo di tempo. Il cambiamento avviene più velocemente di quanto l'occhio umano possa percepire. È simile al modo in cui funziona il cinema: mostrare velocemente una serie di immagini fisse per creare l'illusione del movimento.

Quando fai passare velocemente il piedino da `HIGH` a `LOW`, è come se stessi cambiando la tensione. La percentuale di tempo in cui un piedino è `HIGH` in un periodo è chiamata *duty cycle* o ciclo di lavoro utile. Quando il piedino è `HIGH` per metà di un periodo e `LOW` per l'altra metà, il ciclo di lavoro è 50%. Un ciclo di lavoro più basso produce un LED più attenuato che un ciclo di lavoro più alto.

Arduino Uno ha sei piedini riservati al PWM ***(piedini digitali 3, 5, 6, 9, 10 e 11)***; sono identificati da ~ accanto al loro numero sulla scheda.

In questo progetto come ingressi userai delle ***fotoresistenze*** (sensori che cambiano la loro resistenza secondo la quantità di luce che li colpisce; sono conosciuti anche come fotocellule o fotorivelatori). Se ne colleghi una estremità ad Arduino, puoi misurare il cambiamento di resistenza controllando la tensione sul piedino.

COSTRUISCI IL CIRCUITO

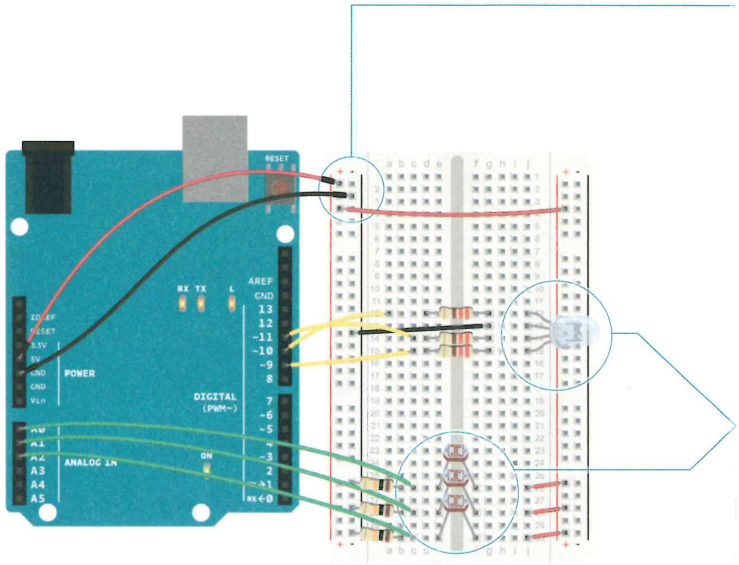


Fig. 1

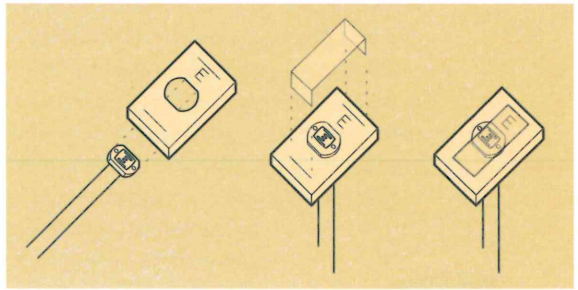


Fig. 2

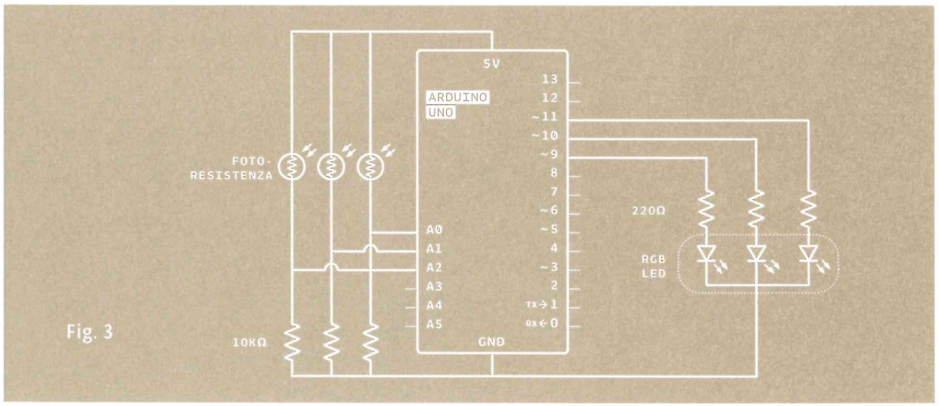


Fig. 3

1. Collega la breadboard così hai alimentazione e massa su entrambi i lati, come nei progetti precedenti.

2. Metti tre fotoresistenze sulla breadboard a cavallo del separatore centrale come in Fig. 1. Collega una estremità di ogni fotoresistenza all'alimentazione. Sull'altro lato, collega a massa una resistenza da 10 kilo ohm. Questa resistenza è in serie con la fotoresistenza, e insieme formano un partitore di tensione. La tensione nel punto in cui si incontrano è proporzionale al rapporto delle loro resistenze, secondo la legge di Ohm (vedi il Progetto 01 per maggiori dettagli sulla legge di Ohm). Al variare della luce che colpisce la fotoresistenza cambia la tensione nel punto di contatto. Sullo stesso lato della resistenza, collega le fotoresistenze agli ingressi analogici 0, 1 e 2 con un ponticello.

3. Prendi tre filtri colorati e posizionali ciascuno su una fotoresistenza. Metti il filtro rosso sulla fotoresistenza collegata all'A0, il verde su quella collegata all'A1 e il blu su quella collegata all'A2. Ognuno di questi filtri consente solo alla luce di una specifica lunghezza d'onda di arrivare al sensore. Il filtro rosso fa passare solo luce rossa, il filtro verde solo luce verde e il filtro blu solo luce blu. Questo permette di rilevare i livelli di colore nella luce che colpisce i sensori.

4. Il LED con 4 gambe è un LED RGB a catodo comune. Il LED contiene tre elementi separati – rosso, verde e blu – e una massa comune (il catodo). Creando una differenza di tensione tra il catodo e la tensione d'uscita dei piedini PWM dell'Arduino (che sono connesi agli anodi attraverso resistenze da 220 ohm), il LED sfuma nei tre colori. Prendi nota di quale sia il piedino più lungo sul LED, posizionalo sulla breadboard e collegalo a massa. Collega gli altri tre piedini alle uscite digitali 9, 10 e 11, tramite le resistenze da 220 ohm. Assicurati di collegare ogni LED al piedino PWM corretto, seguendo l'immagine a sinistra.

IL CODICE

Costanti utili

Imposta le costanti per i piedini che stai usando come ingressi e uscite, così puoi tenere traccia di quale sensore si accoppia con quale colore del LED. Usa `const int` come tipo di dato.

Variabili per immagazzinare le letture dei sensori e il livello di luce di ogni LED

Aggiungi variabili per i valori dei sensori in ingresso e per i valori di luminosità dei LED. Si può usare il tipo di dati `int` per tutte le variabili.

Imposta la direzione dei piedini digitali e configura la porta seriale

Nel `setup()`, avvia la comunicazione seriale a 9600 bps. Come nell'esempio precedente, la usi per visualizzare i valori dei sensori tramite il monitor seriale. Inoltre, puoi vedere i valori mappati che utilizzi per sfumare il LED. Inoltre, definisci i piedini del LED come uscite con `pinMode()`.

Leggi il valore di ogni sensore di luce

Nel `loop()` leggi i valori del sensore sui piedini A0, A1 e A2 con `analogRead()` e memorizza il valore delle variabili. Metti un breve `delay()` tra ogni `analogRead()` per dar tempo all'ADC (convertitore analogico-digitale) di fare il suo lavoro.

Riporta le letture del sensore al computer

Riporta su una riga i valori del sensore.
`"\t"` è equivalente a premere il tasto `"tab"` sulla tastiera.

```
const int greenLEDPin = 9;
const int redLEDPin = 11;
const int blueLEDPin = 10;

const int redSensorPin = A0;
const int greenSensorPin = A1;
const int blueSensorPin = A2;
```

```
int redValue = 0;
int greenValue = 0;
int blueValue = 0;

int redSensorValue = 0;
int greenSensorValue = 0;
int blueSensorValue = 0;
```

```
void setup() {
 Serial.begin(9600);

 pinMode(greenLEDPin,OUTPUT);
 pinMode(redLEDPin,OUTPUT);
 pinMode(blueLEDPin,OUTPUT);
}
```

```
void loop() {
 redSensorValue = analogRead(redSensorPin);
 delay(5);
 greenSensorValue = analogRead(greenSensorPin);
 delay(5);
 blueSensorValue = analogRead(blueSensorPin);

 Serial.print("Raw Sensor Values \t Red: ");
 Serial.print(redSensorValue);
 Serial.print("\t Green: ");
 Serial.print(greenSensorValue);
 Serial.print("\t Blue: ");
 Serial.println(blueSensorValue);
```

Converti le letture dei sensori

La funzione per modificare la luminosità dei LED tramite PWM è `analogWrite()`. Ha bisogno di due parametri: il piedino da modificare e un valore tra 0 e 255. Il secondo rappresenta il ciclo di lavoro che Arduino produce sul piedino specificato. Un valore di 255 imposta il piedino `HIGH` per tutto il tempo, portando alla massima luminosità il LED collegato. Un valore di 127 imposta il piedino `HIGH` per metà del periodo, rendendo il LED più sfumato. Il valore 0 imposta il piedino `LOW` per tutto il tempo, spegnendo il LED. Per convertire la lettura di un sensore da un valore tra 0 e 1023 a uno tra 0 e 255 per `analogWrite()`, dividi per 4 la lettura del sensore.

Riporta i livelli di luce del LED che sono stati calcolati

Invia i nuovi valori mappati su una riga separata.

Imposta i livelli di luce del LED

USALA

Quando hai programmato e collegato Arduino, apri il monitor seriale. Il LED è probabilmente di un colore biancastro, a seconda del colore predominante della luce nella tua stanza. Guarda i valori provenienti dai sensori nel monitor seriale: se sei in un ambiente con illuminazione stabile, il numero sarà probabilmente abbastanza consistente.

Spegni la luce nella stanza in cui stai lavorando e guarda cosa succede ai valori dei sensori. Con una torcia elettrica, illumina ogni sensore e nota come i valori cambiano nel monitor seriale e come il LED cambia colore. Quando le fotoresistenze sono coperte con un filtro, reagiscono solo alla luce di una certa lunghezza d'onda. Questo ti dà la possibilità di cambiare ogni colore in modo indipendente.

```
31  redValue = redSensorValue/4;
32  greenValue = greenSensorValue/4;
33  blueValue = blueSensorValue/4;
```

```
34  Serial.print("Mapped Sensor Values \t Red: ");
35  Serial.print(redValue);
36  Serial.print("\t Green: ");
37  Serial.print(greenValue);
38  Serial.print("\t Blue: ");
39  Serial.println(blueValue);
```

```
40  analogWrite(redLEDPin, redValue);
41  analogWrite(greenLEDPin, greenValue);
42  analogWrite(blueLEDPin, blueValue);
43 }
```

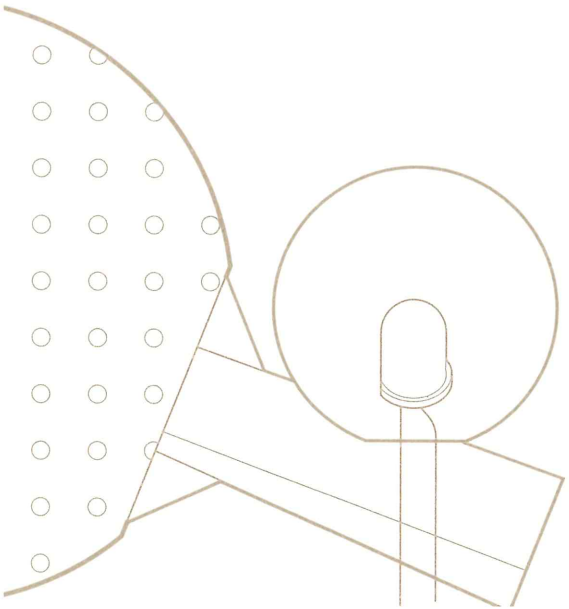
Avrai notato che il valore d'uscita della fotoresistenza non copre tutto l'intervallo da 0 a 1023. Per questo progetto va bene, ma per una spiegazione più dettagliata di come calibrare l'intervallo effettivo, vedi il Progetto 06.

Probabilmente hai notato che la sfumatura dei LED non è lineare. Quando il LED è circa a metà luminosità, sembra che non diventi più luminoso. Questo perché i nostri occhi non percepiscono la luminosità in maniera lineare. La luminosità del LED dipende non solo dal livello di `analogWrite()`, ma anche dalla distanza della luce dal diffusore, la distanza dell'occhio dalla luce e la luminosità della luce rispetto al resto della luce nella stanza.

Come puoi usarla per sapere se fuori è una bella giornata mentre stai lavorando in casa? Quali altri tipi di sensori puoi utilizzare per controllare il colore del LED?

Il LED di per sé è abbastanza carino, ma non è una vera lampada. Tuttavia, ci sono vari modi per diffondere la luce e a farla somigliare a una lampadina tradizionale a incandescenza. Per esempio, una pallina da ping pong con un foro per il LED può diventare un bel diffusore. Altre possibilità sono quelle di coprire la luce con colla traslucida o levigare la superficie del LED. Non importa quale opzione scegli, si perderà almeno un po' di luminosità quando è diffusa, ma probabilmente sarà molto più gradevole.

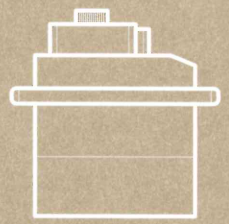
Non più limitato al solo accendere e spegnere luci, ora hai il controllo sulla luminosità. analogWrite() è la funzione che ti permette di comandare in PWM i componenti collegati ai piedini 3, 5, 6, 9, 10, 11, variando il ciclo di lavoro.



La palla da ping pong tagliata per ospitare il LED.
Fig. 4

05

POTENZIOMETRO



SERVOMOTORE

MOTORE ARM

CONDENSATORE DA 100 uF

CONNETTORE A PETTINE (3 piedini)

INGREDIENTI