ARDUINO SERIJSKI MONITOR

Uradićemo da Arduino komunicira sa računarom preko USB priključka, tako što ćemo koristiti paket Arduino IDE i odvojeni paket koji nam omogućava da dizajniramo dobar raspored ekrana.

Arduino je povezan sa računarom preko USB kabla. Ovo se koristi za napajanje i programiranje, ali isto tako može da se koristi za komunikaciju između skice i računara.

Komanda	Objašnjenje
Serial.begin(baud rate)	Šalje komandu komunikacije u "baud rate" (broj bauda).
Serial.print(variable)	Štampa (šalje) sadržaj promenljive na serijski port (u ASCII).
Serial.println(variable)	Isto, ali posle prelazi na sledeći red.
Serial.write(variable)	Štampa (šalje) sadržaj promenljive na serijski port (kao bajt).
variable = Serial.read()	Čita token (jedno slovo ili cifru) iz serijskog porta i skladišti ga u promenljivoj. Kada nema ništa na portu ova komanda ne čeka dolazeće podatke, već daje vrednost promenljivoj -1.
variable = Serial.available()	Proverava da li su dolazeći podaci primljeni (variable = true) ili nisu (variable = false).
void serialEvent()	Specijalna funkcija koja se poziva kada se nešto desi na serijskom portu (prekid).

Prvi korak je da podesite brzinu komunikacije.

Postoji nekoliko standardnih vrednosti od kojih možete izabrati. Što je veća vrednost, to je brža komunikacija. Za većinu projekata na vežbama nije važno koju brzinu ćete izabrati, sve dok izaberete istu brzinu na Arduino i računaru.

Uobičajena brzina je 9600 bauda.

Mi ćemo koristiti komandu Serial.print da odštampamo sadržaj brojača. Brojač ćemo nazvati t (mogli smo ga nazvati brojač, ali ne volimo da kucamo) i dati mu početnu vrednost 0. Zatim ćemo podesiti serijsku vezu sa računarom na 9600 bauda i poslati vrednost t na računar.

Sada ćemo morati da povećamo vrednost za 1 (ipak, mora da bude brojač), na primer pomoću iskaza „t=t+l“ ili „t++“ što znači potpuno isto. U sledećoj tabeli ćete pronaći najčešće skraćenice.

Skraćenica	Skraćeno od	Značenje
i+ +	i = i + 1	Povećava i za l.
i--	i = i - 1	Smanjuje i za 1.
i + = 5	i = i + 5	Sabira 5 sa i.
i-=5	i = i - 5	Oduzima 5 od i.
i*=5	i = i * 5	Množi i sa 5.
i/=5	i = i / 5	Deli i sa 5.

Primer:

```
int t=0;

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

void loop() {
	Serial.print(t);
	t++;
	delay(1000);
}
```

Nakon što se skica učita u Arduino možete da pokrenete ugrađeni terminal koji se zove serijski monitor (Serial Monitor).

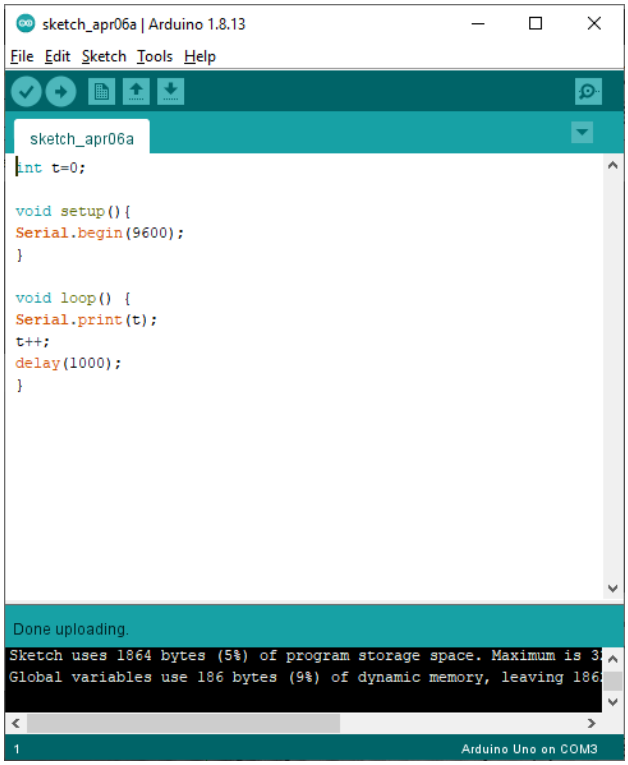
Ikona je sa desne strane na traci menija u Arduino IDE.

Proverite da li Arduino IDE koristi ispravnu baudnu brzinu (označeno brojem 3 na sledećoj slici), koja treba da bude jednaka baudnoj brzini u skici, što je 9600.

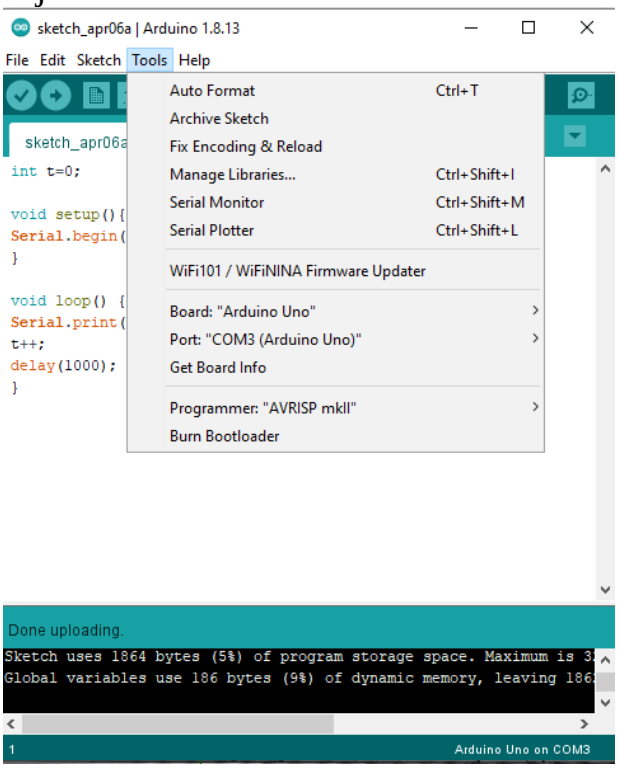
Rezultat je niz brojeva jedan pored drugog: 1234567891011 itd. Takođe, kada koristite komandu Serial.println, možete staviti svaki broj u odvojenom redu (isprobajte ovo).

Možda ste primetili da rezultat uvek počinje sa 0, bez obzira na to da li ste pokrenuli serijski monitor odmah ili nakon nekog vremena. To je zato što kada uključite serijski priključak on resetuje skicu, pa počinje ponovo.

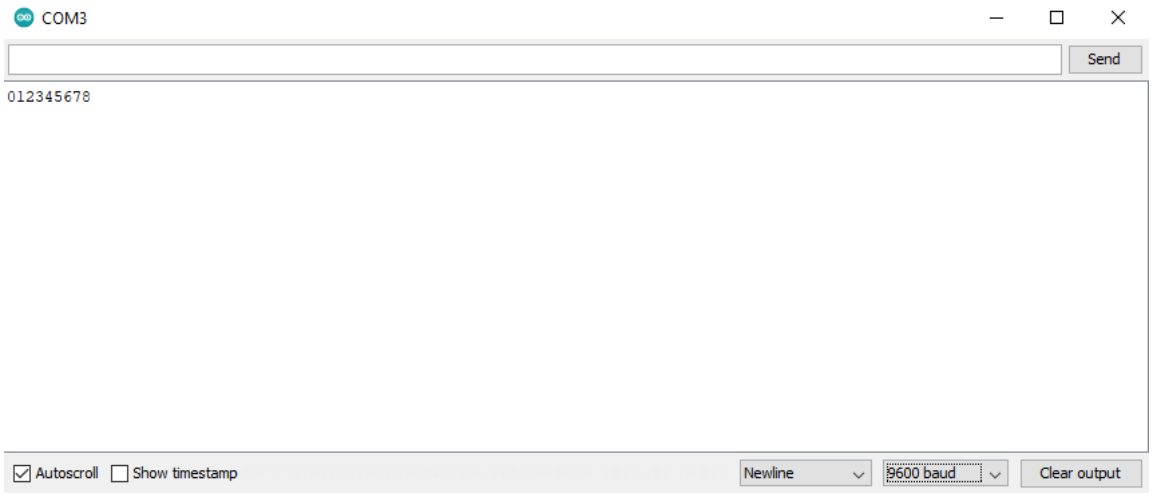
Unos kôda:



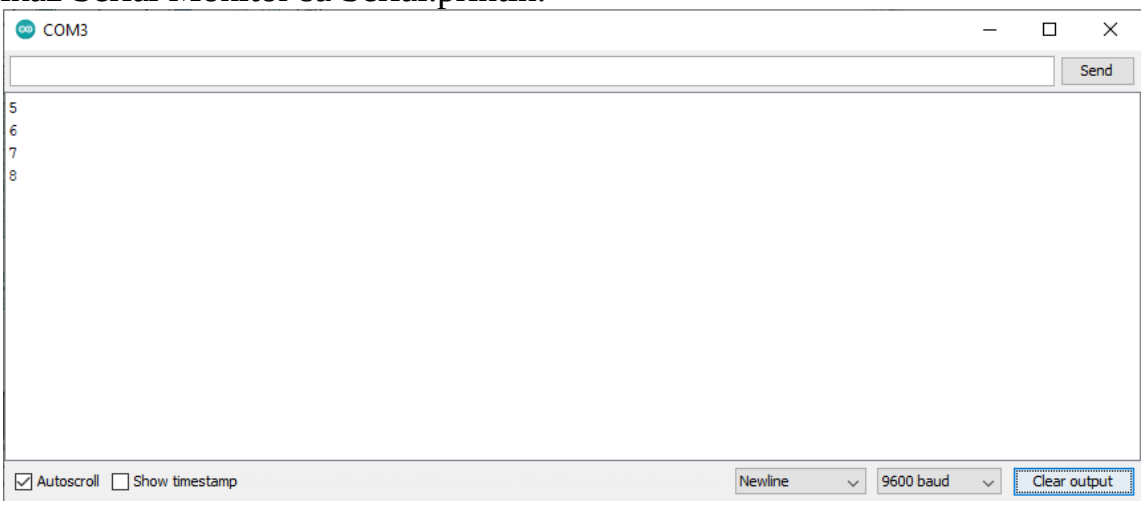
Serial Monitor u meniju Tools:



Prikaz Serial Monitor sa Serial.print:



Prikaz Serial Monitor sa Serial.println:



Ovo radi dobro i to je brz i jednostavan način za prikaz podataka iz Arduino skice na računar.

Razlika između Serial.print i Serial.write je u tome što je prva namenjena za slanje podataka koje čitaju ljudi, a druga se koristi za podatke koje čitaju računari (ili mašine).

Dobar raspored ekrana na računaru bez programiranja

Mogućnosti serijskog monitora su prilično ograničene. Ako iskoristite odvojeni komunikacioni paket poput HyperTerminal-a, Arduino će slati komande tako da se tekstovi štampaju na fiksnim lokacijama ekrana. To znači da možete napraviti dobar raspored, bez bilo kakvog programiranja na računaru.

HyperTerminal se koristio za prethodne verzije operativnog sistema Windows, kao što su Windows 7 i ranije.

U narednom primeru je korišćena demo verzija programa:

Advanced Serial Port Monitor to read and record COM port data

https://www.eltima.com/advanced-serial-port-monitor/

Svaki token ima numerički kôd, koji se zove ASCII.

Mali deo ovog kôda je prikazan u sledećoj tabeli. Postoji 128 kôdova (od 0 do 127, uključujući i 127) koji u celom svetu imaju potpuno isto značenje u svemu što čak prividno liči na računar. Osim toga, postoje kôdovi koji se razlikuju na osnovu zemlje ili jezika.

Na primer, kôdovi sa većim brojevima sadrže znak za evro €, znak autorskog prava ©, ali i znakove iz jezika u kojima se ne koristi latinično pismo, kao što su japanski i ruski.

Kôdovi od 0 do 31, uključujući i 31, ne mogu se odštampati i namenjeni su za komunikaciju između računara i terminala, odnosno u našem slučaju između Arduino-a i računara.

ASCII	opis ili token	štampanje
7	Terminal bell	ne
8	TAB	ne
9	Backspace	ne
10	Linefeed (prelazak u novi red)	ne
13	Carriage return (odlazak na početak reda)	ne
27	Escape	ne
32	Space	da
48	0	da
65	A	da
97	a	da

Posle tokena Escape (ASCII 27) komanda je poslata. Postoji desetine komandi, ali većina je nebitna za nas. Sve komande koje se koriste u ovom primeru su navedene u sledećoj tabeli.

VTIOO komanda	Objašnjenje
[H	Pomera kursor u gornji levi ugao
[x;yH	Pomera kursor na položaj (x, y)
[?25l	Sakriva kursor (čini ga nevidljivim)
[?25h	Prikazuje kursor
[0K	Briše red od kursora do kraja
[7m	Štampa belo na crno

Recimo da želimo da pomerimo kursor na lokaciju 10,15. To znači da počinjemo u gornjem levom uglu 10 položaja sa desne strane i 15 položaja na dole (napominjemo da za razliku od matematike tačka 0,0 je gornji levi ugao, a ne donji levi ugao).

Prvo treba za pošaljemo znak Escape kako bi obavestili HyperTerminal (ili sličan program) o slanju komande, a ne tokena koje treba odštampati.

Zatim šaljemo komandu [x;yH i unosimo 10 za x i 15 za y.

```
Serial.write(27);
Serial.print(„[10;15H”);
```

Vidite da smo se opredelili za Serial.write kada smo slali token Escape.

To je zato što ne nameravamo da ovo čita čovek: namenjeno je za program HyperTerminal.

Posle ovakvog zaključka možemo koristiti i komandu write za VT100 komandu i to je stvarno moguće. Ali pošto je to tekst, moguće su obe opcije. Ali budući da je komanda print fleksibilnija, koristićemo tu komandu.

Sledeće što ćemo sada štampati počinje sa položaja 10,15. Tako da će se pojaviti brojač „u sredini“ ekrana i čak možemo dodati dobar opisni tekst.

Na primer, "Promenljiva t je sada:" ("Variable t is now:").

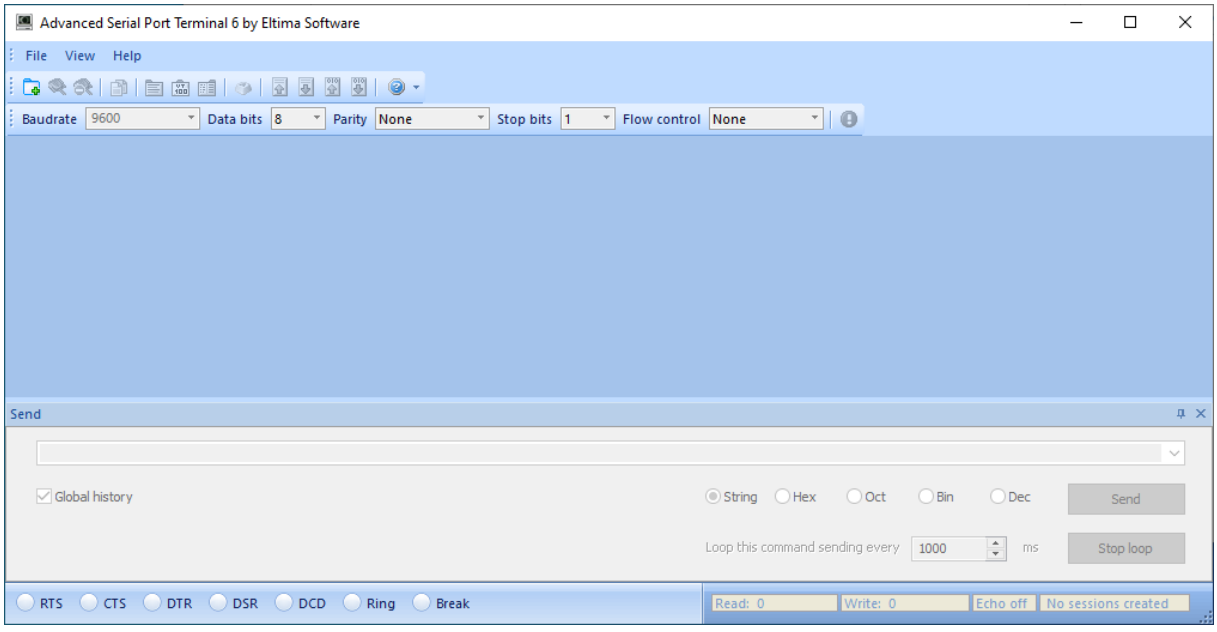
```
int t;

void setup(){
        Serial.begin(9600);
        Serial.write(27);
        Serial.print("[?25l");
}
void loop() {
        Serial.write(27);
        Serial.print("[10;15H");
        Serial.println("Variable t is now: ");
        Serial.println(t);
        t++;
        delay(1000);
}
```

Program HyperTerminal treba neka podešavanja kako bi komunicirao sa Arduino. Uverite se da je Arduino povezan sa računarom pre nego što nastavite.

1. Pokrenite program HyperTerminal ili sličan program (Advanced Serial Port Terminal). Ekran za vezu će se automatski otvoriti.

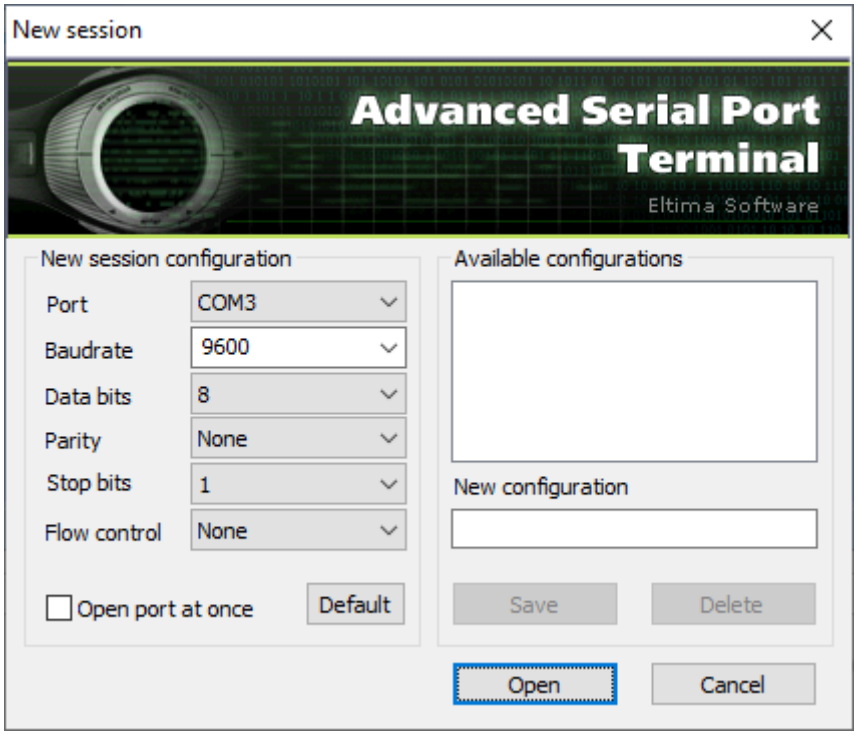
Advanced Serial Port Terminal



2. Otvorite novu sesiju na sledeći način: File \ New Session

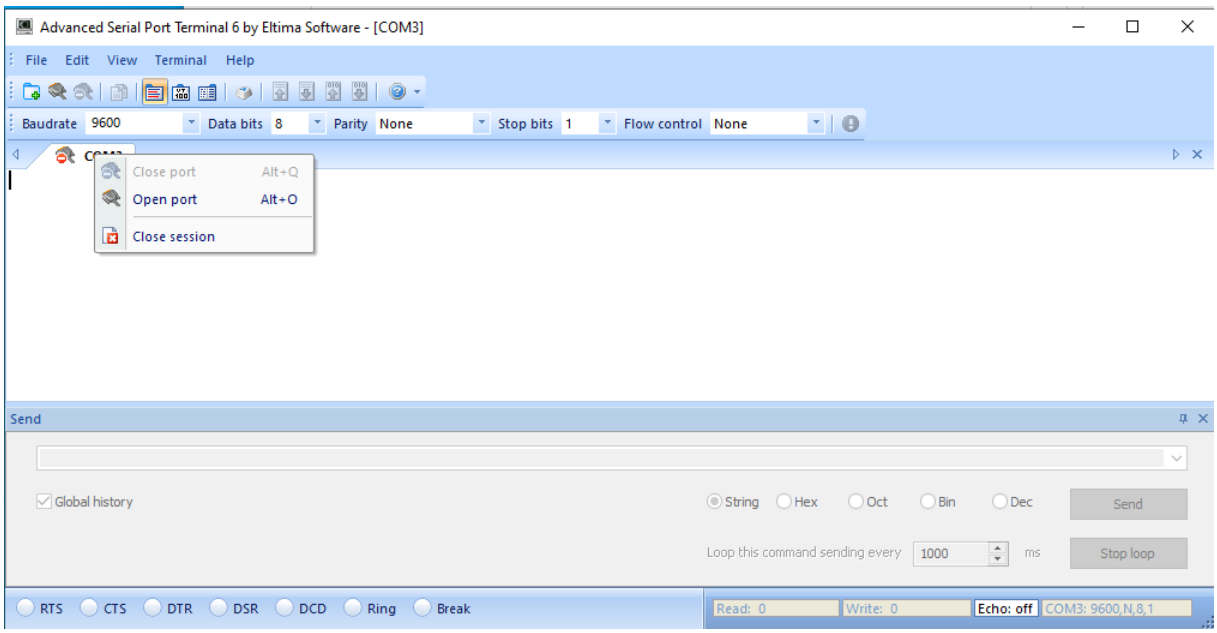
3. Unesite sledeće podatke i kliknite Open.

Port	COM3
Baudrate	9600
Data bits	8
Parity	None
Stop bits	1
Flow Control	None

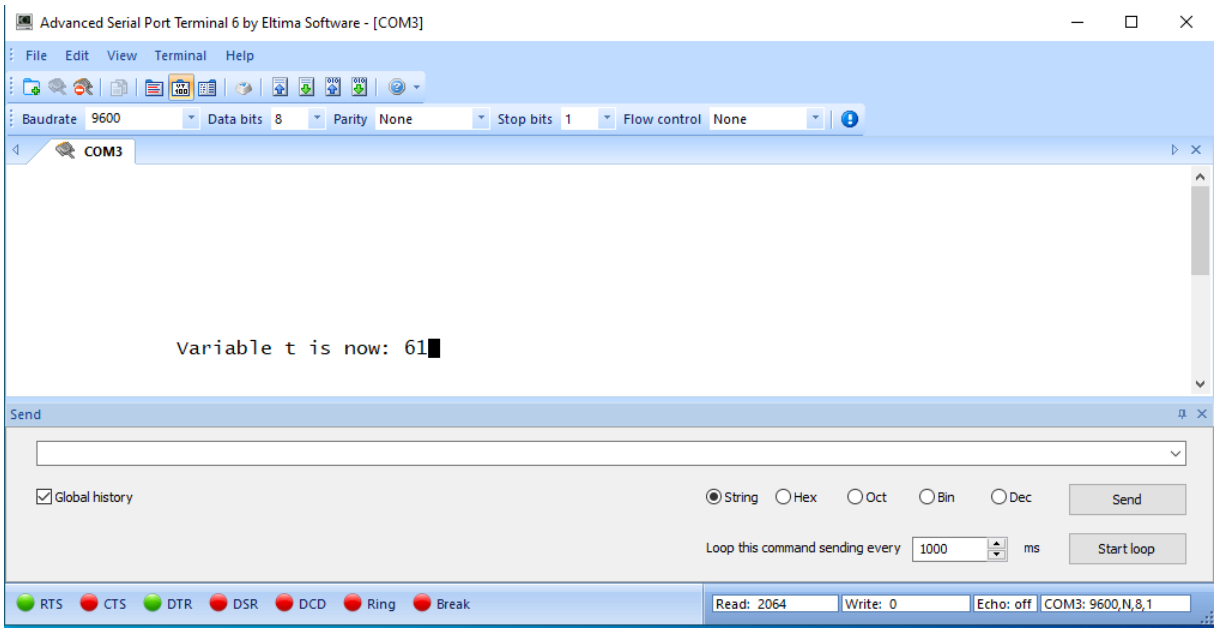


4. Serijska veza je sada otvorena i pojaviće se brojač kao što je prikazano na sledećoj slici.

Desni klik na COM3 pa Open port



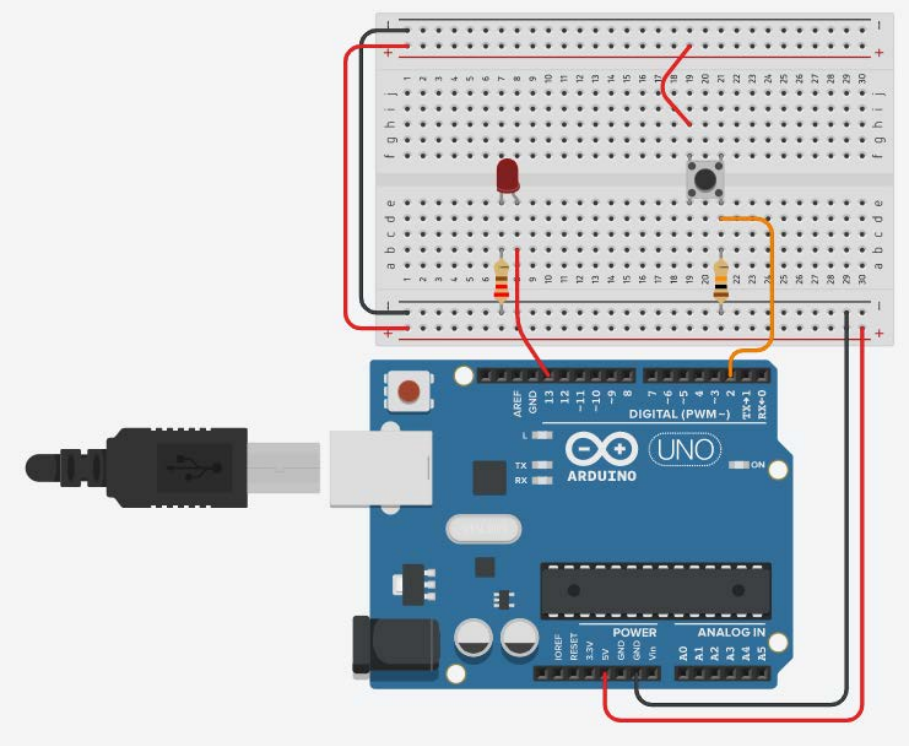
Prikaz terminala za odgovarajući prikaz namenjen ljudima:
VT100 Terminal (Alt + T)



Primer: Prekidač i Serial Monitor:

Vratite se na primer 7.2. U navedenom primeru ćemo dodati opciju obaveštavanja u Serial Monitor da li je taster (prekidač) pritisnut ili ne. Interval podešavanja treba da bude 1 sekunda.

Skica:



Komponente:

Name	Quantity	Component
U1	1	Arduino Uno R3
D1	1	Red LED
R1	1	220 Ω Resistor
S1	1	Pushbutton
R2	1	10 kΩ Resistor

Primer kôda:

```
int buttonState = 0;
void setup()
{
        pinMode(2, INPUT); // čita se sa njega status tastera
        pinMode(13, OUTPUT);
        Serial.begin(9600);
}
void loop() {
          buttonState = digitalRead(2); // čita stanje sa pina 2
          if (buttonState == HIGH)
          {
                    digitalWrite(13, HIGH);
          }
           else
          {
                    digitalWrite(13, LOW);
          }
          if (digitalRead(2)==HIGH){
                Serial.println("Switch is closed");
        }
          else{
                        Serial.println("Switch is open");
        }
  delay(1000);
}
```

Primer: Tajmer sa automatskim ponovnim pokretanjem

Sada kada znamo kako da pročitamo prekidač i kako da uključimo i isključimo LED, pravljenje tajmera je prilično jednostavno.

Nakon što ste pritisnuli prekidač, LED treba da se uključi. Nakon unapred podešenog vremena (kao primer koristićemo 5 sekundi), LED treba da se ponovo isključi.

Možemo da koristimo iskaz if-then i kašnjenje od 5000 milisekundi (ms).

Dok je LED uključen, skica ne obraća pažnju na prekidač. To nije potrebno, jer je LED već uključen i može biti uključen duže od toga.

Ali zamislite da želite da korisnik može ponovo da pokrene LED sa kašnjenjem od 5 sekundi svaki put kada pritisne prekidač. To bi bilo veoma zgodno kada se projekat koristi za osvetljavanje stepenica.

Prekidač možete montirati na svakom spratu i korisnik može da pritisne prekidač u prolazu, tako što će svaki put resetovati kašnjenje od 5 sekundi. To je mnogo jednostavnije nego čekati pored prekidača da se svetlo isključi, a zatim ponovo pritisnuti prekidač.

Rešenje je da skica ne čeka 5 sekundi, već 500 puta 10 milisekundi.

Ukupno kašnjenje je i dalje isto, ali sada skica može da proverava prekidač svakih 10 ms.

Nemoguće je pritisnuti prekidač tako brzo da ga skica propusti. Svaki put kada se pritisne prekidač, skica resetuje brojač na 500. Sve dok je brojač veći od nule, skica :oduzima 1 i čeka 10 ms. Samo kada brojač više nije veći od 0, skica prekida oduzimanje i isključuje LED.

Primer kôda:

```
int button = 2;
int led = 13;
int counter = 0;

void setup() {
	pinMode(led, OUTPUT);
	pinMode(button, INPUT);
}

void loop(){
	if (digitalRead(button) == HIGH) {
		counter=500;
		digitalWrite(led, HIGH);
}
if (counter>0){
	counter--;
	delay(10);
}
else{
	digitalWrite(led, LOW);
}
}
```