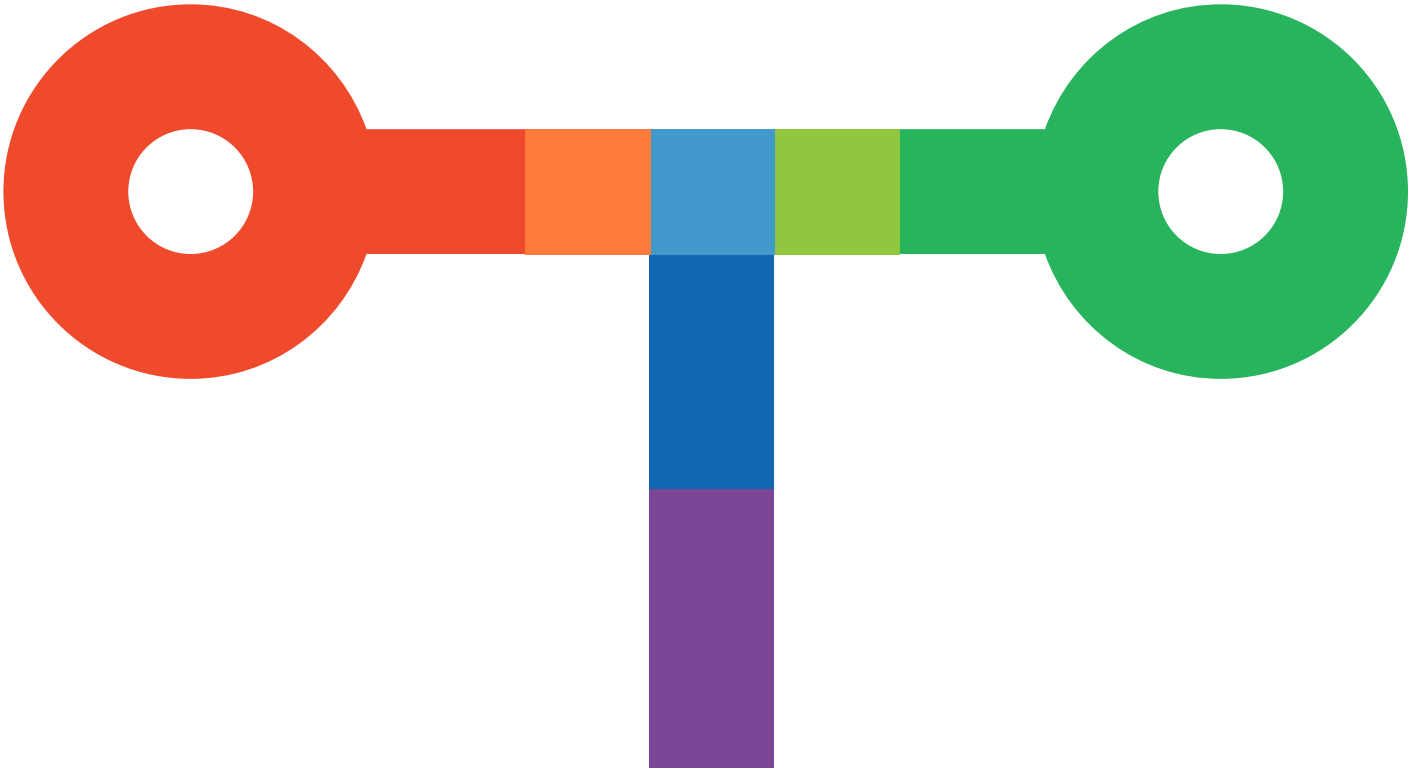
Mr. sc. Mateo Banović

TINKERCAD OSNOVE

PRIRUČNIK ZA NASTAVNIKE I NASTAVNICE



Podržano od:

< IT Girls >

UJEDINJENE NACIJE
BOSNA I HERCEGOVINA

TINKERCAD
OSNOVE

PRIRUČNIK ZA NASTAVNIKE I NASTAVNICE

Mateo Banović, mr. sc., tehničkog odgoja i informatike

2023.

SADRŽAJ

UVOD

Intenzivan IT razvoj, razvoj tehnike i tehnologije sa sobom nosi niz fenomena. Sve je digitalizirano, umreženo, automatizirano, sve ima računalnu kontrolu i vrlo često se odvija u realnom vremenu. Pred našim očima skoro sva mehanika evoluira u mehatroniku, podaci se sa tvrdih hardverskih medija „dižu u oblake“, lokalizirani upravljački sistemi i radno-razvojne platforme sele se na udaljene servere. Svi se sistemi u motornim vozilima računalno kontroliraju putem specijaliziranih softvera, svi uređaji u kućanstvu „postali su pametni“, kuće i stanovi su „pametniji“. Sve se povezuje. Različiti, nekada nespojivi sistemi spajaju se i srastaju. „Obrazovna paradigma STE(A)M“ nije nastala zbog potrebe da bude izmišljena, već je samo prirodan fenomen, jedan savremeni racionalan odgovor na ekstreman IT razvoj i razvoj tehnike i tehnologije koji potiče srastanje svega (telefon, računar, fotoaparat, kamera, sat itd., svi bez problema stanu u jedan „pametni mobitel“ ili, ako ga preciznije definiramo, može se slobodno reći da je to prijenosni, bežičnoj komunikaciji prilagođen računar, čije dimenzije limitira jedino veličina displeja i baterije).

Razvojna platforma Arduino, tačnije ATMEL ATMEGA 328p kontroler i hardversko okruženje koje ga čini kompatibilnim sa velikim brojem aktuatora/izvršnika (OUTPUT) i senzora (INPUT), savršeno se uklapa u edukacijski koncept STE(A)M. Dostupni standardizirani robotički i mehatronički setovi koji raspolažu kvalitetno izrađenim konstrukcijskim elementima (spojni elementi, različiti elementi za transformaciju i prijenos kretanja i sl.), izvršnicima i motorima. Idealni su za realizaciju predefiniranih vježbi, gdje se učenicima nude sheme sklapanja, preporučeni kodovi, a edukatorima maksimalno skraćuje vrijeme potrebno za pripremu časova. Najveći izazov za učenice i učenike jeste rješavanje konkretnih i izazovnih problema iz prakse i okruženja, gdje robotički setovi ne mogu puno pomoći.

Razvojna platforma Arduino omogućava razvoj projekata koji su plod učeničke i nastavničke kreativnosti, inovativnosti ili su se naprosto našli pred njima kao problem iz njihovog okruženja koji treba riješiti.

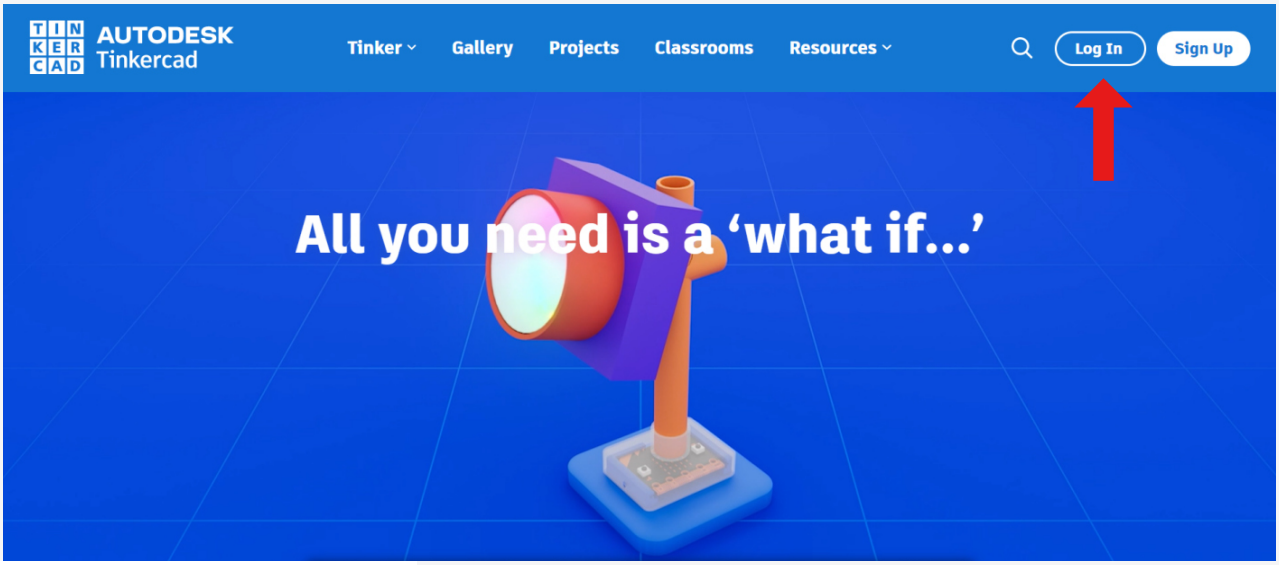
Izvršnici ili aktuatori, senzori, elektroničke komponente koje se koriste za razvoj u elementarnom su, višestruko dostupnom, standardnom i namjenski kompatibilnom obliku. Konstrukcije i mehanizmi se moraju kreirati, dizajnirati, razvijati i uklopiti u jednu cjelinu u potpunosti od početka. Posjedovanje 3D printera malim koracima postaje dio standarda i u školama u Bosni i Hercegovini. Mogućnost da se paralelno kreiraju, dizajniraju i razvijaju, s jedne strane, automatizirani mikroračunarski, elektronički, senzorski, izvršni sistemi (razvojne platforme Arduino), a s druge strane konstrukcije i mehanizmi (3D dizajn i 3D printanje) dovodi nas u ravan s vrhunskim interdisciplinarnim STE(A)M projektima, koji do kraja ogole i jasno ukažu na razvojnu fenomenologiju, gdje se spajaju u jedno nekada potpuno odvojene inženjerske grane poput elektrotehnike i mašinstva (mehatronika). To je, naravno, jasna implikacija da odvojeni odsjeci mehatronike koji postoje na elektrotehničkim i mašinskim fakultetima moraju srasti u jedan interdisciplinaran odsjek koji bi spram naučnih i obrazovnih potreba bio istovremeno na oba fakulteta. Taj skoro u potpunosti digitaliziran proces projektnog razvoja omogućava učenicima i učenicama da iskustveno spoznaju tendencije IT razvoja i razvoja tehnike i tehnologije.

Savremena inženjerska i naučna praksa inicijalni razvojni proces u potpunosti je preselila u simulaciono virtualno okruženje. Dobrobit takvog pristupa je višestruka. Simulaciono se testira da li je planirano projektno rješenje uopće izvedivo u praksi. Dolazi se do približno najboljeg i optimalnog rješenja, a da se u praksi nije upotrijebio ni jedan razvojni element. Ne troše se nepotrebno ekonomski resursi na nabavku elemenata koji bi bili korišteni u početnoj testnoj razvojnoj fazi. Kada se razvojni proces treba preseliti u praksu, moguće je napraviti skoro potpunu projektnu specifikaciju. U ovoj publikaciji predstavit će se simulaciono-razvojna platforma **AUTODESK TINKERCAD** bazirana na webu.

AUTODESK TINKERCAD

Tinkercad je besplatna aplikacija prilagođena korisniku namijenjena za 3D dizajniranje, elektroniku i kodiranje. Autodeskov Tinkercad je uspješno prošao reviziju Međunarodnog društva za tehnologiju u obrazovanju (ISTE) i time dobio pečat usklađenosti za obrazovne namjene. Recenzenti su utvrdili da ovaj resurs pomaže u izgradnji temeljnih tehnoloških vještina potrebnih za podršku ISTE standardima za učenike (ISTE, n.d.).
U ovoj publikaciji daju se detaljne smjernice za maksimalnu iskoristivost razvojnih i simulacionih resursa Tinkercada za simulacioni razvoj elektroničkih sklopova kojima se upravlja mikrokontrolerima i njihovu programsku kontrolu i kodiranje. Posebno je izdvojeno poglavlje sa smjernicama za predavače i predavačice, gdje su prikazani načini korištenja platforme Tinkercad kao veoma efikasnog predavačkog digitalnog pomagala.
AUTODESK Tinkercad je platforma bazirana na webu. Pristupa joj se kada u adresno polje web-preglednika upišemo: **tinkercad.com**

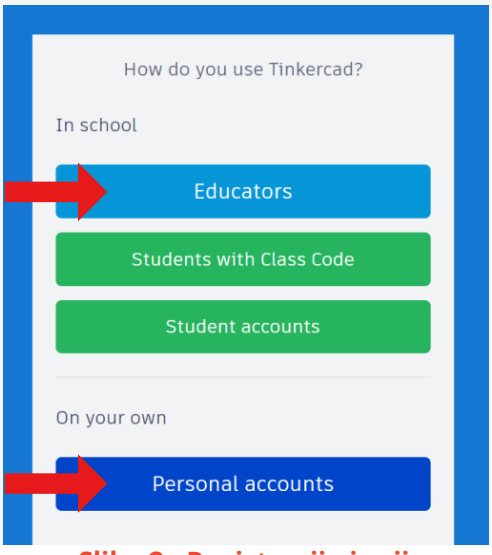
1



Slika 1 - TINKERCAD

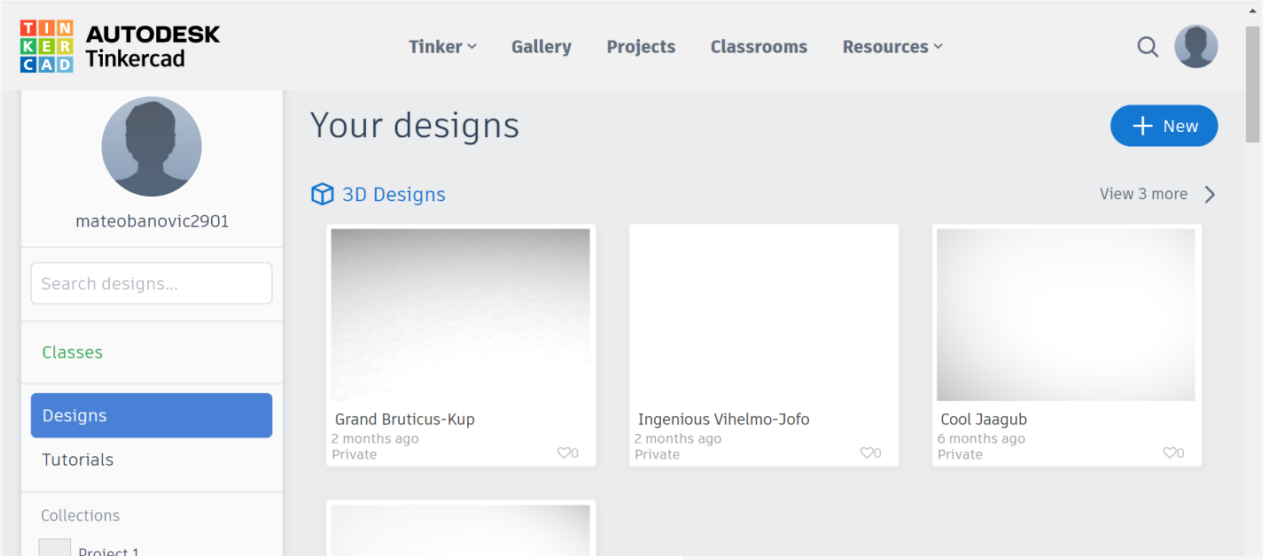
Nakon klika na **Log In** otvara se sljedeći dijaloški okvir:
Ako želimo koristiti Tinkercad isključivo za simulacioni razvoj vlastitih projekata, klikom na **„Personal accounts“** kreiramo korisnički nalog. Edukatori i edukatorice (nastavnici/nastavnice, profesori/profesorice i sl.) mogu napraviti naloge koji im omogućavaju interakciju s učenicima i učenicama, kreiranje virtuelnih učionica i svakodnevno praćenje i nadzor učeničkih simulacionih projekata. Učenici i učenice imaju mogućnost kreiranja učeničkih naloga, kao i pristup razredima putem razrednog koda i korisničkog imena.

2



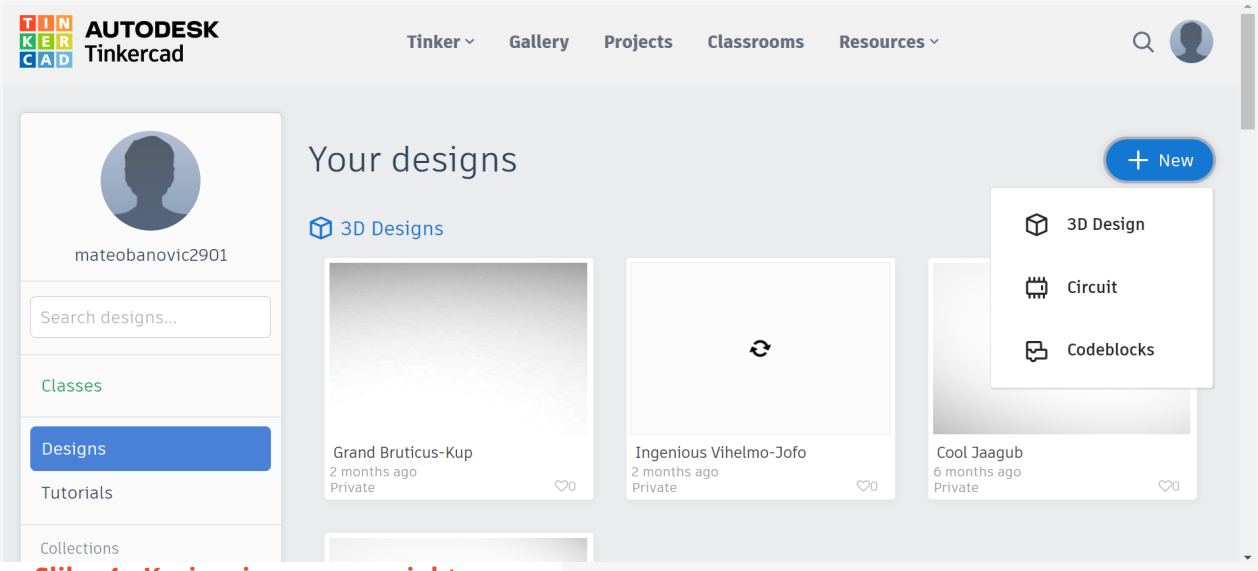
Slika 2 - Registracija i prijava

Nakon prijave, kreirani edukatorski profil u pregledniku ima izgled kao na **Slici 3**.



Slika 3 - Početni web-prozor Tinkercada

Na lijevoj strani web-prozora nalaze se podaci o korisniku (slika – opciono i nadimak). Ispod njih je tražilica pomoću koje se mogu pretraživati prethodni korisnički projekti, zatim se mogu birati kreirani razredi **(Classes)**, krenuti sa dizajniranjem **(Designs)**, pregledati tutorijali ako smo ih učitavali, te kreirati kolekcije srodnih projekata.



Slika 4 - Kreiranje novog projekta

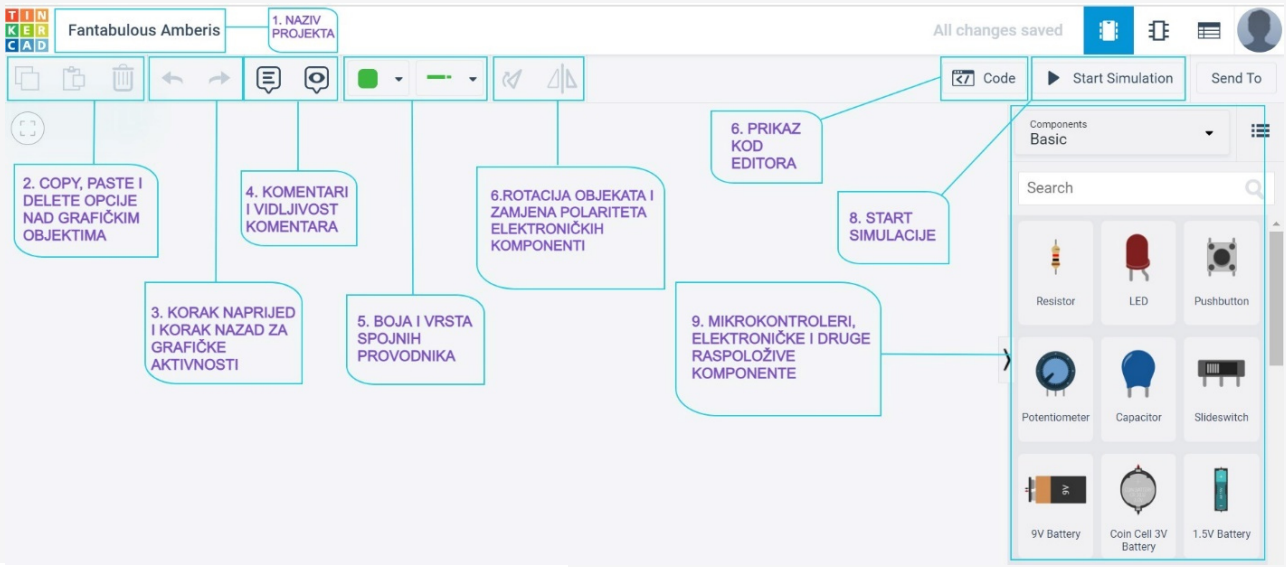
U gornjem desnom uglu dostupno je dugme **(+ New)** , gdje se nakon klika iz otvorenog menija može birati:

- **3D Design** (3D dizajn),
- **Circuit** (Elektronički krugovi kojima se upravlja mikroračunarima),
- **Codeblocks** (Blokovsko kodiranje).

Za dizajniranje i razvoj Arduino simulacionih projekata bira se: **Circuit.**

Razvojno okruženje Arduino Tinkercad

Pred korisnikom se sada nalazi razvojno okruženje **Tinkercad.** Radni prozor se sastoji od sljedećih elemenata **(Slika 5**):



Slika 5 - Tinkercad Circuit alati

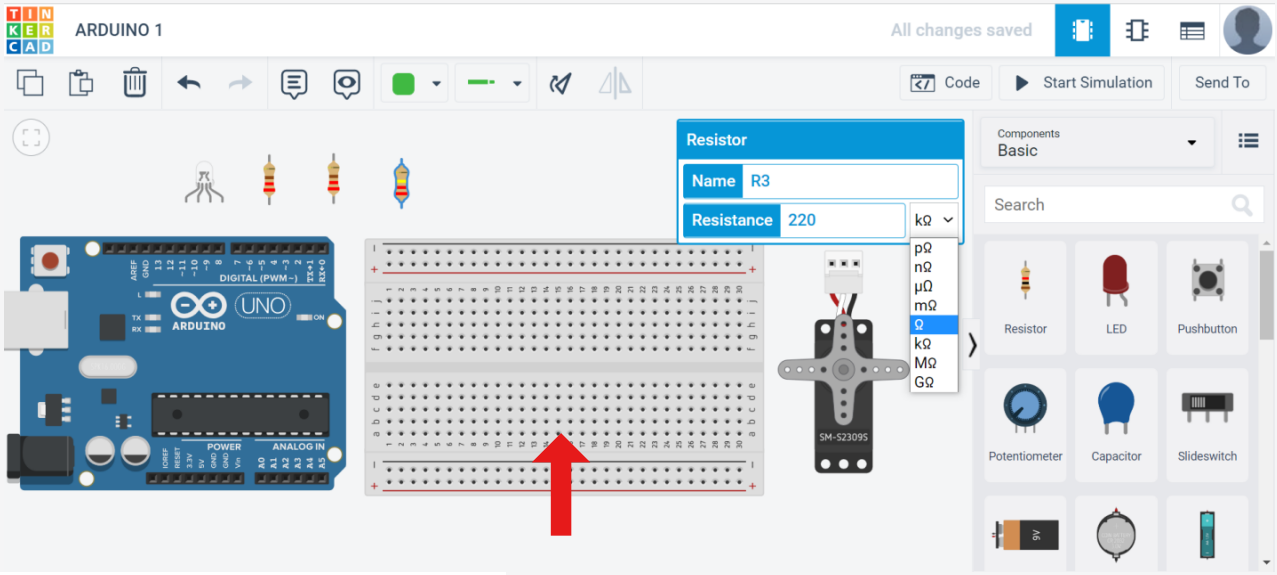
1. Tekstualno polje sa automatski dodijeljenim nazivom projekta, koji mijenjamo po želji.
2. Standardne naredbe za kopiranje **(copy, paste)** i brisanje **(delete)**, kojima se mogu kopirati i brisati klikom prethodno odabrani elementi postavljeni na radnoj površini, ili zalijepiti kopirani elementi na radnu površinu. Radna površina predstavlja prostor ispod grupe naredbi za brzi pristup.
3. Naredbe **undo** (korak nazad) i **redo** (korak naprijed) čija se funkcionalnost u ovom slučaju odnosi na brisanje, odnosno vraćanje posljednjih aktivnosti pri kreiranju sklopa na radnoj površini.
4. Da bi kreirani sklop bio razumljiviji, moguće je bilo gdje na radnoj površini postaviti komentare sa informacijama bitnim za funkcionalnost sklopa.
5. Sklopove razvijamo tako što mikrokontroler – Arduino Uno pločicu pomoću provodnika, matador pločice, elektroničkih elemenata, izvršnika (aktuatora) i senzora spajamo u električni sklop. Da bi shematski i logički sklop bio jasniji, pružena je mogućnost promjene boje izolacije na provodnicima, kao i odabir konektora najpogodnijih za sklop.
6. Odabrane elemente na radnoj površini rotiramo spram potreba za njihovo pozicioniranje na sklopu.
7. Klikom na komandno dugme **Code** otvara se okvir za kodiranje. Kada pišemo novi kod za razvijeni sklop, možemo birati da kodiramo pomoću blokova (koristi se kada se projekti rade sa učenicima i učenicama razredne nastave), kombinirano blokovima i tekstom i tekstualnim kodovima, koji se primarno i koriste.
8. Nakon što smo kreirali sklop, napisali kod, klikom na komandno dugme **Start Simulation** pokrećemo simulaciju sklopa.
9. Na desnoj strani nalazi se pomjerljivi okvir sa grupiranim elementima za razvoj sklopa.

Elemente možemo pretraživati upisom imena u polje za pretraživanje, ili ih pomoću klizača pronalaziti. Na radnu površinu iznosimo potrebne razvojne elemente tako što kliknemo na lijevi taster miša na željeni element, držimo taster i stisnuti element iznosimo na željeno mjesto na radnoj površini.

KREIRANJE I SIMULACIONI RAZVOJ PROJEKATA

6

Pri kreiranju simulacionog projekta, potreban nam je Arduino Uno mikrokontroler, aktuatori/ izvršnici, senzori i elektroničke komponente potrebnih karakteristika koje klikom na element možemo podešavati. Na **Slici 6** vidimo da se odabirom električnog otpornika pojavio padajući meni, na kojem se može promijeniti ime električnog otpornika, vrijednost njegove otpornosti kao i mjerna jedinica. Preciznije rečeno, odabirom bilo koje komponente iznesene na radnu površinu pojavljuje se plutajući meni koji uvijek nudi mogućnost promjene imena komponenti, te dodatna podešavanja spram njene varijabilnosti.

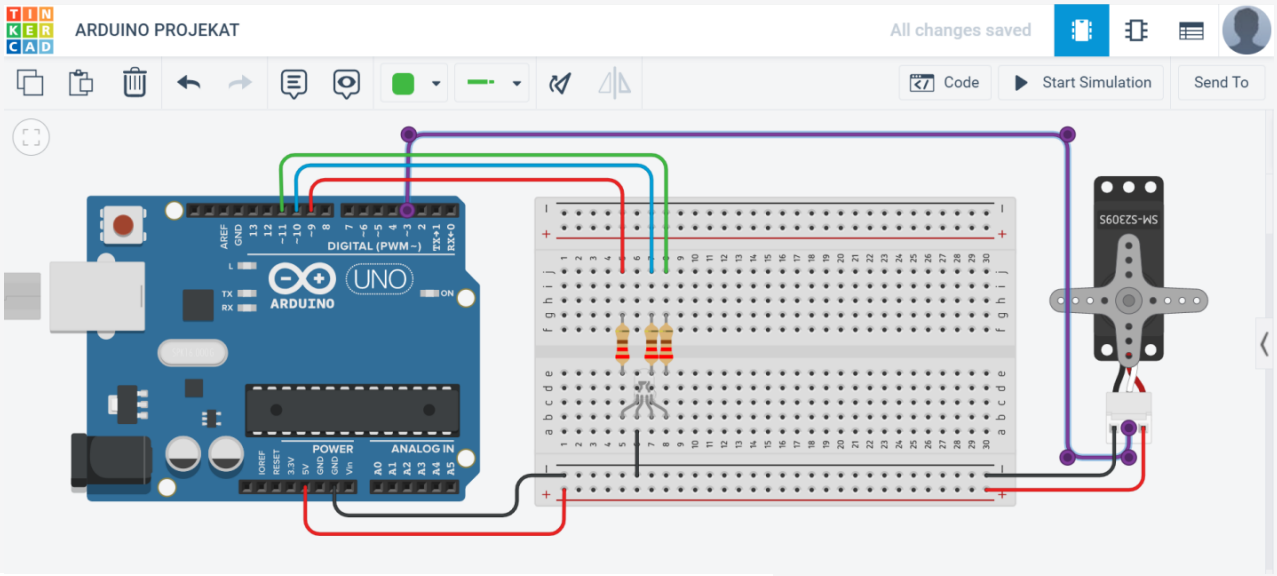


Slika 6 - Plutajući izbornik

Pločica matador **(breadboard)** predstavlja idealno rješenje za razvoj eksperimentalnih sklopova i izvođenje velikog broja vježbi. Praviti sklopove tako da se koriste vijci i matice, da se leme spojni kontakti i sl. spor je proces, a uz to elektroničke komponente nakon jedne vježbe ili projekta često se više ne mogu upotrijebiti. Pločica matador omogućava jednostavno ubodno spajanje elektroničkih komponenti u zadati sklop, te se te komponente nakon korištenja jednostavno uklone. Takvo praktično rješenje idealno je za izvođenje učeničkih vježbi i projekata, gdje je primarno usvajanje logike elektroničkih sklopova.

Slika 7 prikazuje simulacionu Tinkercad realizaciju sklopa upravljanog pomoću Arduino Uno pločice, gdje je shematsko spajanje izvedeno na pločici matador. Kao što je već naglašeno, vodiče povlačimo lijevim tasterom miša, savijamo ih po želji dodatnim klikom, a boja vodiča i vrsta konektora bira se spram potreba sklopa i njegove bolje preglednosti.

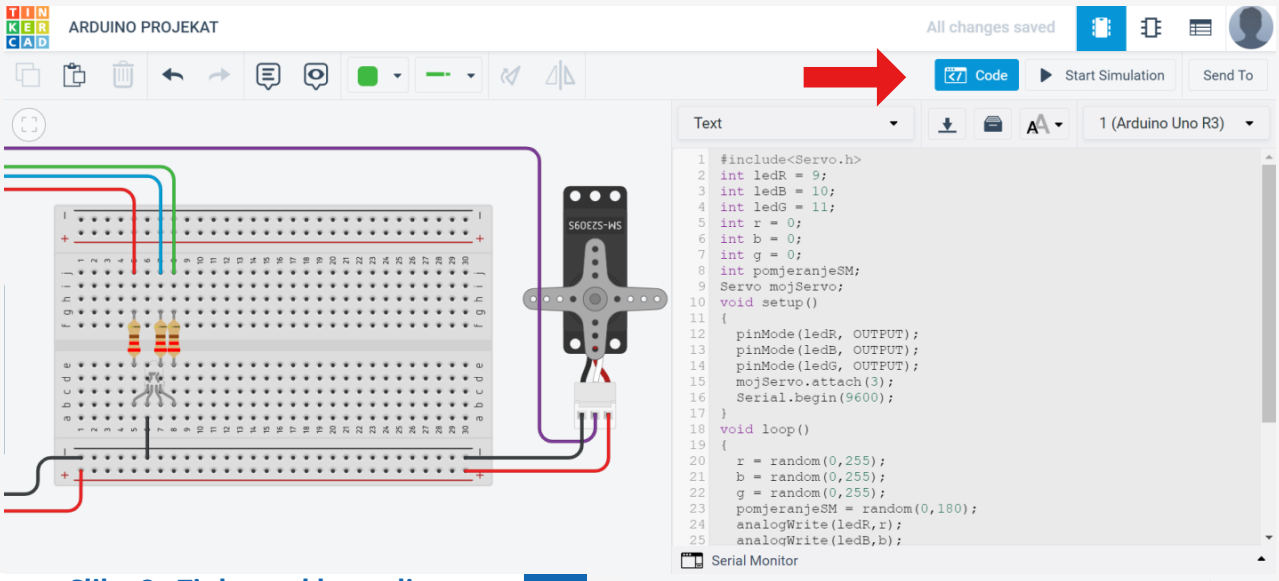
7



Slika 7 - Realizacija sklopa na pločici matador

Klikom na komandno dugme **Code**, s desne strane otvara se editor za pisanje Arduino IDE koda (C++). Kod se piše na isti način kao u standardnom editoru Arduino IDE **(Slika 8)**.

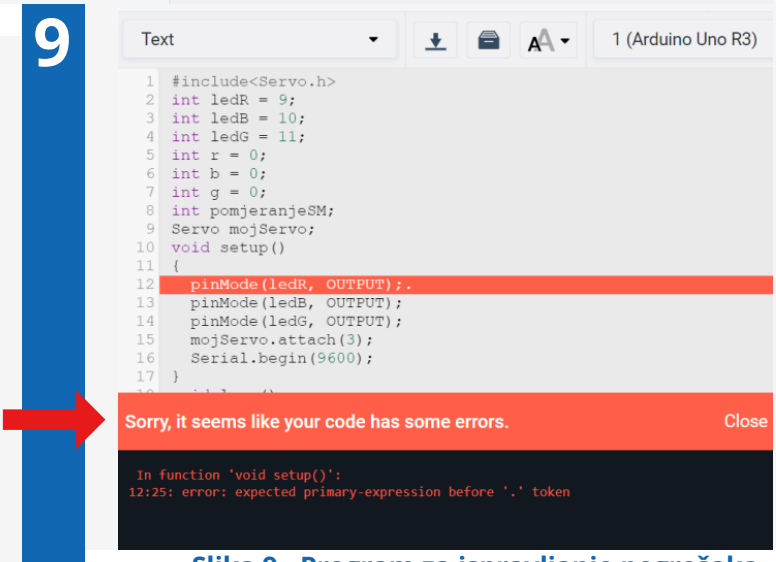
8



Slika 8 - Tinkercad kao editor

9

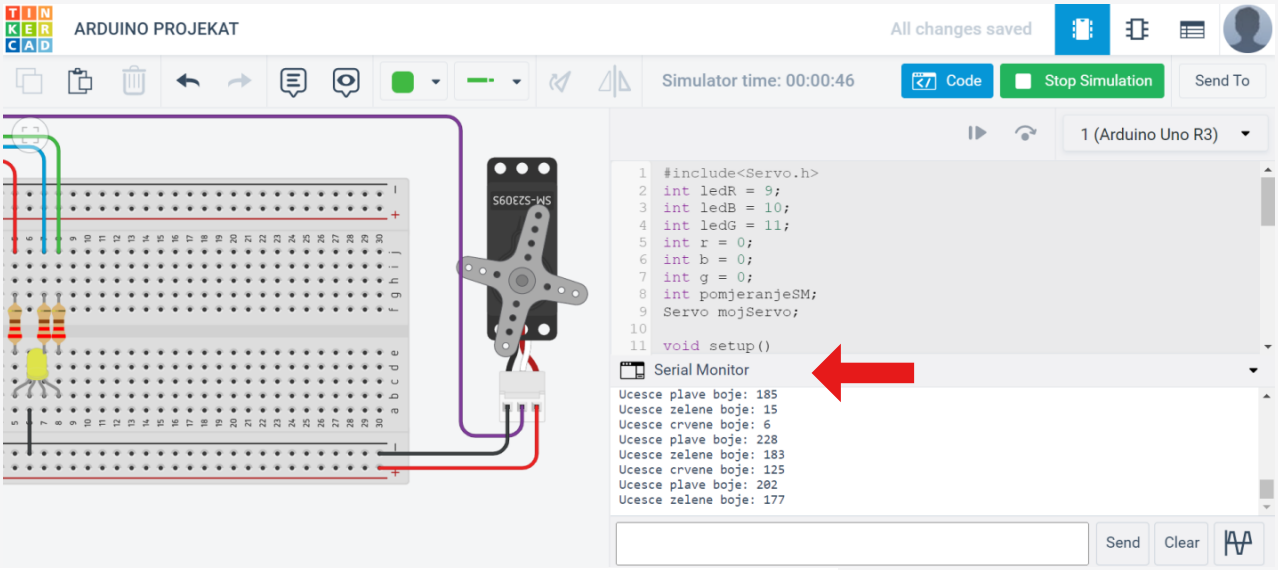
Klikom na komandno dugme **Start Simulation**, pokreće se simulacija rada sklopa. Ako postoje greške u kodu, bit će označene crvenom pozadinskom bojom i pritom će biti ispisana poruka: *Sorry, it seems like your code has same errors.* Ispod te poruke program za ispravljanje pogrešaka navodi liniju koda u kojoj se nalazi greška i daje njeno objašnjenje **(Slika 9).**



Slika 9 - Program za ispravljanje pogrešaka

Kada je kod uspješno sastavljen, pokreće se simulacija **(Slika 10)** i kreirani sklop simulira ono što bi trebao raditi kada se izvede u praksi. Klikom na komandno dugme **Serial Monitor**, ako je korištena biblioteka **Serial**, bit će prikazano sve ono što je metodama iz biblioteke **Serial** poslano prema serijskom monitoru, npr. **Serial.println().**

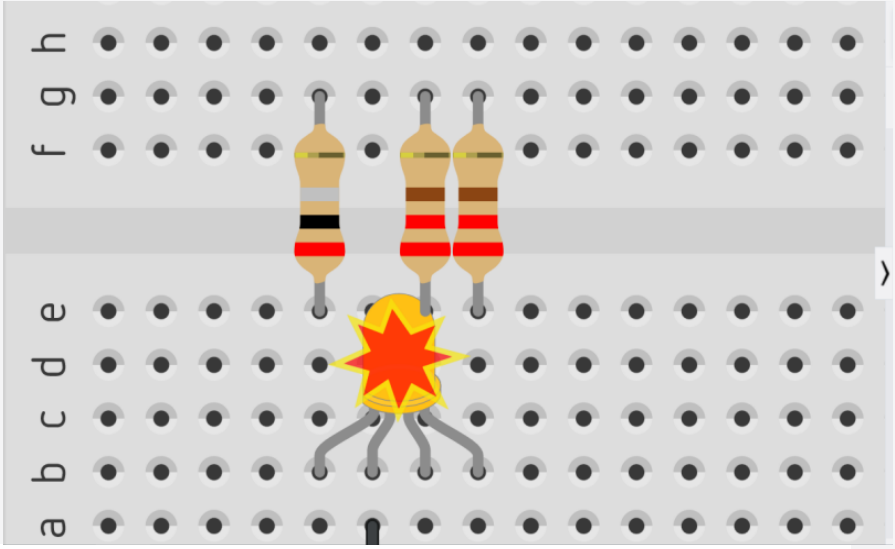
10



Slika 10 - Pokretanje simulacije za razvijeni sklop

Razvojna platforma Arduino za izvođenje projekata koristi osjetljive elektroničke komponente, senzore i aktuatore/izvršnike i, ako se oni na pogrešan način spoje ili se na njih dovede napon veći od granične vrijednosti koji ima određena elektronička komponenta ili struja veća od granične, može doći do njihovog većeg oštećenja ili uništavanja. Na **Slici 11** jedan od otpornika spojenih na RGB LED diodu ima nižu vrijednost od potrebne, te je struja koja protiče kroz diodu veća od dozvoljene i uništava je. Kada se navedeni primjer desi u praksi ostaje se bez elektroničke komponente. Ako se projekt prethodno kreirao, razvijao i testirao u simulacionom okruženju, pri pokretanju simulacije bit će simulirano stradanje elektroničke komponente. Na **Slici 11** vidi se kako je RGB LED dioda stradala. Kada se ta situacija desi na Tinkercadovom simulatoru, greška će biti otkrivena na vrijeme i pri izradi stvarnog sklopa već će postojati simulaciono testirano ispravno rješenje.

11

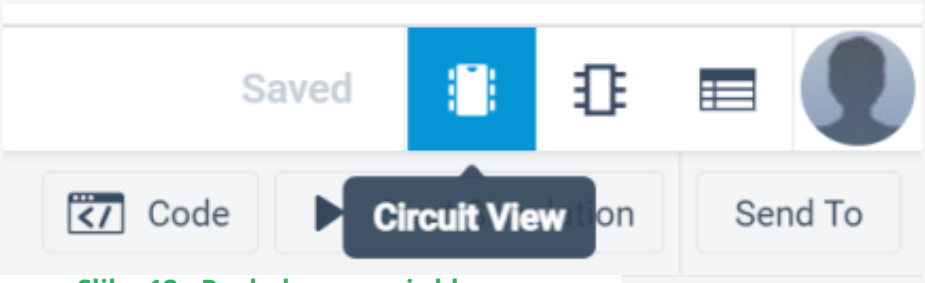


Slika 11 - Primjer greške na elektroničkom sklopu

IZRADA TEHNIČKE DOKUMENTACIJE

Nakon što smo projekat dizajnirali i razvili u radnom pogledu **Circuit View** (komandno dugme **Circuit View** je podrazumijevano odabrano – **Slika 12**), razvojno okruženje Tinkercad nudi dodatne opcije za automatizirano kreiranje tehničke dokumentacije.

12

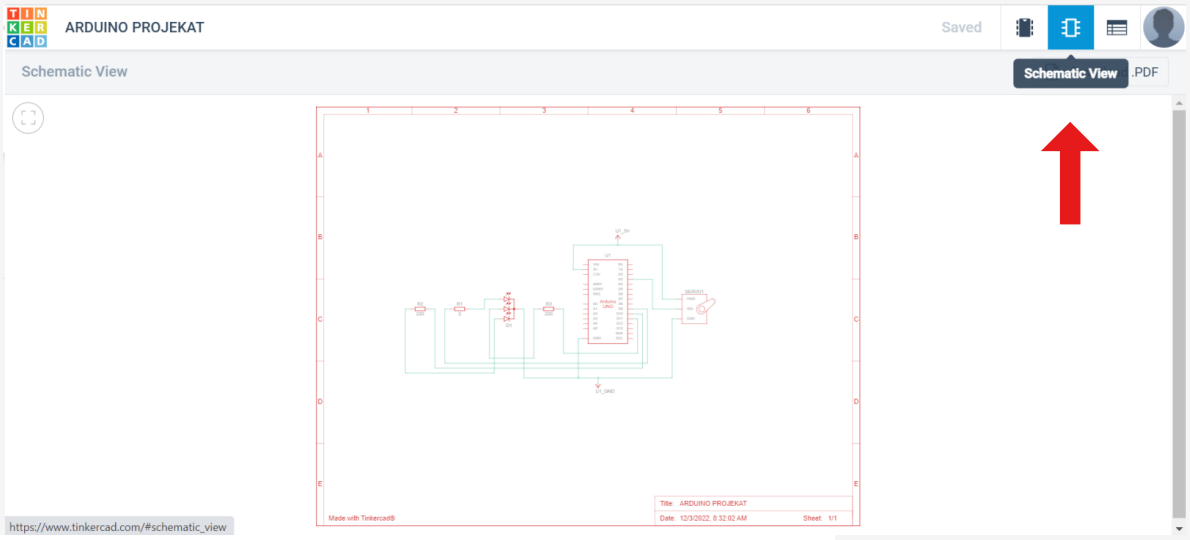


Slika 12 - Pogled za razvoj sklopa

Automatizirana izrada shema sklopa

Da bi imali konkretnu primjenu u praksi, svi sklopovi kada se razvijaju moraju imati kompletnu tehničku dokumentaciju u kojoj se koriste standardizirani simboli u elektrotehnici i elektronici. Izrađuje se klikom na komandno dugme u desnom gornjem uglu ispod kojeg se kada pozicioniramo kursor na njega pojavi oblačić sa porukom **Schematic View (Slika 13).**

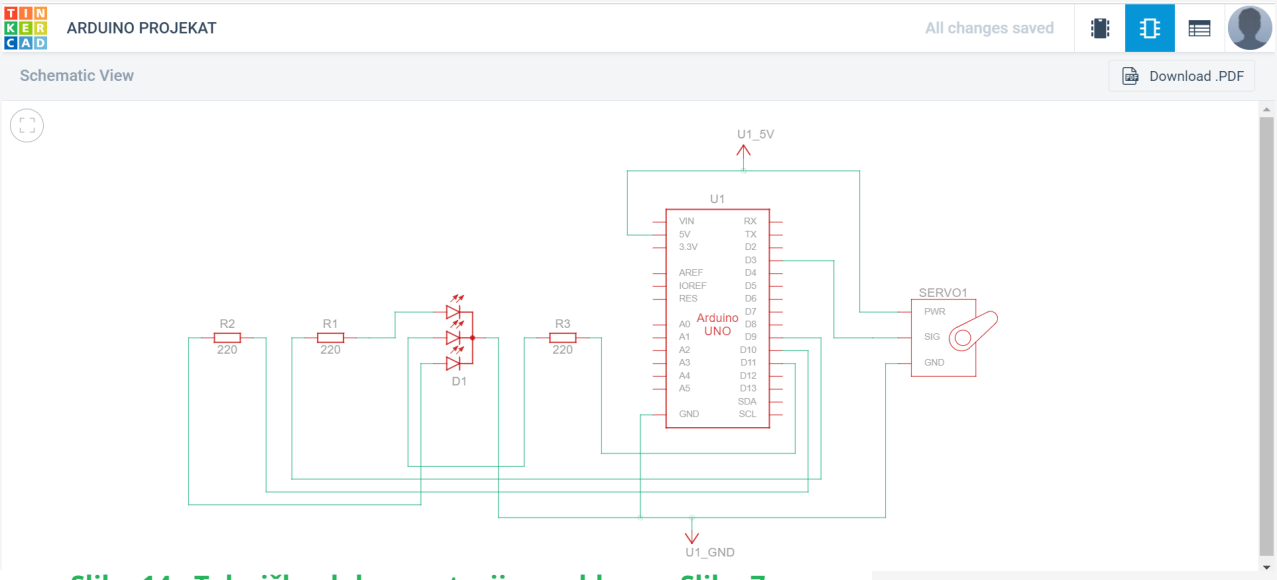
13



Slika 13 - Prikaz automatski generirane tehničke

Na **Slici 14** može se vidjeti automatski generirana elektronička shema projekta prikazanog na **Slici 7.** Analizom elektroničke sheme vidi se prikaz razvojnih komponenti pomoću simbola u elektronici, imena i vrijednosti kako su definirana u razvojnom prostoru **Circuit View**, te jasan prikaz svih dijelova gotovog sklopa. Ova automatizirana, višestruko korisna i dragocjena funkcionalnost štedi pri razvoju projekata vrijeme neophodno za izradu tehničke dokumentacije, a uz to se može koristiti kao obrazovni alat za učenje i bolje razumijevanje elektroničkih shema čija je logika i način funkcioniranja pri razvoju sklopa u pogledu **Circuit** već usvojena.

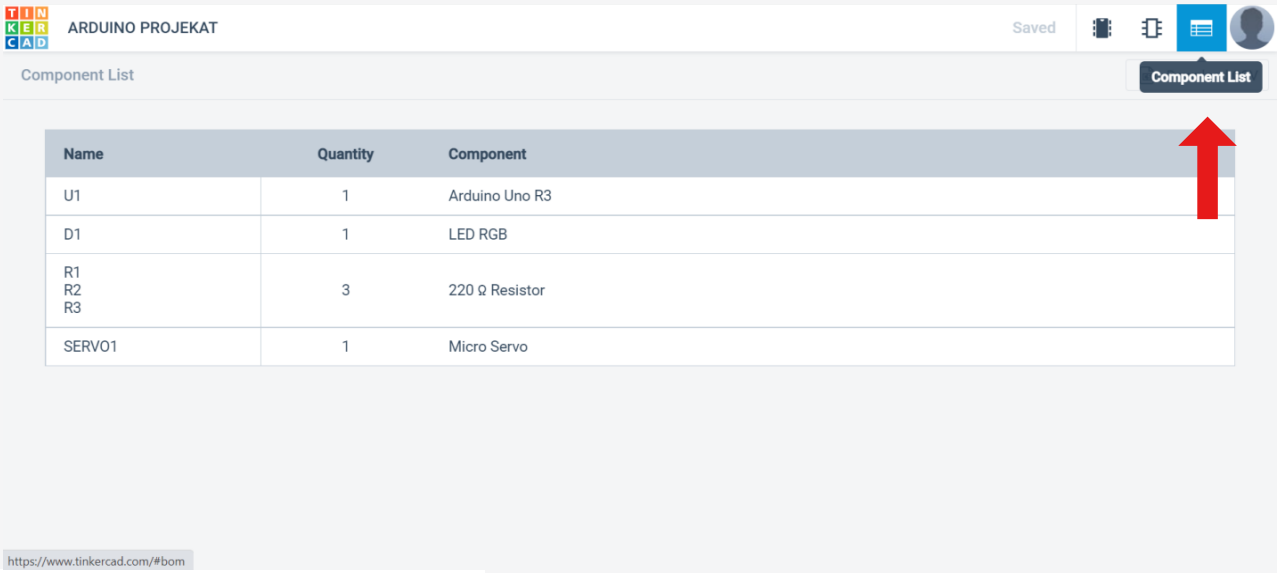
14



Slika 14 - Tehnička dokumentacija za sklop sa Slike 7.

Klikom na komandno dugme **Component List** automatizirano se stvara lista komponenata potrebnih za izvođenje projekta u praksi, koja je također obavezan element tehničke dokumentacije, odnosno, tzv. liste **BOM (engl. Bill of Materials) (Slika 15)**.

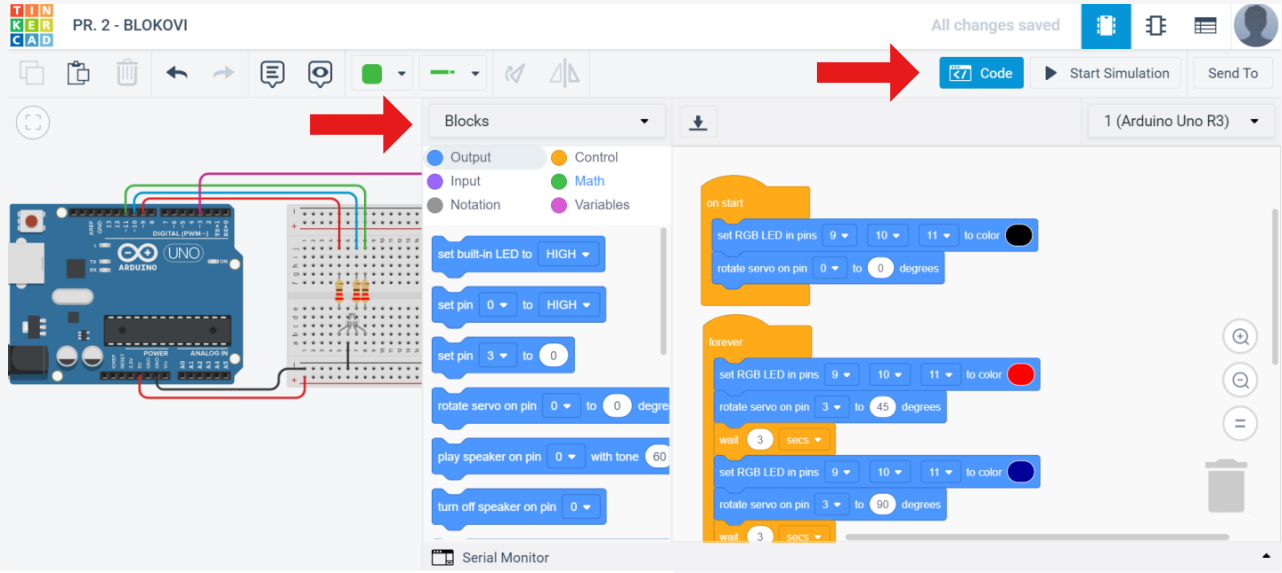
15



Slika 15 - Lista komponenti

BLOKOVSKO PROGRAMIRANJE

Tinkercad nudi mogućnost tekstualnog, blokovskog i kombiniranog kodiranja (blokovi i tekst istovremeno). Kodiranje pomoću blokova pogodno je u radu sa mlađim učenicama i učenicima. Kada se klikne na dugme **Code** otvara se radni prostor za kodiranje. Pri odabiru opcije **Blocks** površina za kodiranje će biti podijeljena na dva dijela. Desno se nalazi radni prostor u kojem se slažu blokovi, a lijevo je izbornik sa raspoloživim blokovima.



Slika 16 - Kodiranje pomoću blokova

Grupe blokova formirane su prema funkcionalnosti:

- **Plavu grupu** čine blokovi koji se odnose na izlazne **(Output)** funkcionalnosti na priključnim pinovima, kao što su npr. kontrola izlaznih vrijednosti napona, upravljanje servo motorima, mikrofonom, serijskim monitorom.

- **Ljubičasta grupa** rezervirana je za upravljanje ulaznim **(Input)** naponskim signalima, sa mogućnošću očitavanja digitalnih ulaznih senzorskih naponskih signala (binarne vrijednosti 0 i 1) i preciznije „analogne" kontrole (uzorkovano u rasponu od 0 do 1023), uz dodatak prilagođenih blokova za čitanje temperature, ugla na servo pinu, kontrolu ultrasoničnog senzora i sl.

- **Žuti blokovi** sadrže dva ključna događaja (funkcija **setup()** i funkcija **loop()**), naredbe za kontrolu toka programa i vremensku naredbu **wait**.

- **Blokovi** sa matematičkim i logičkim operatorima i drugim matematičkim funkcijama **zelene** su boje.

- **Rozi blokovi** omogućavaju kreiranje i korištenje varijabli.

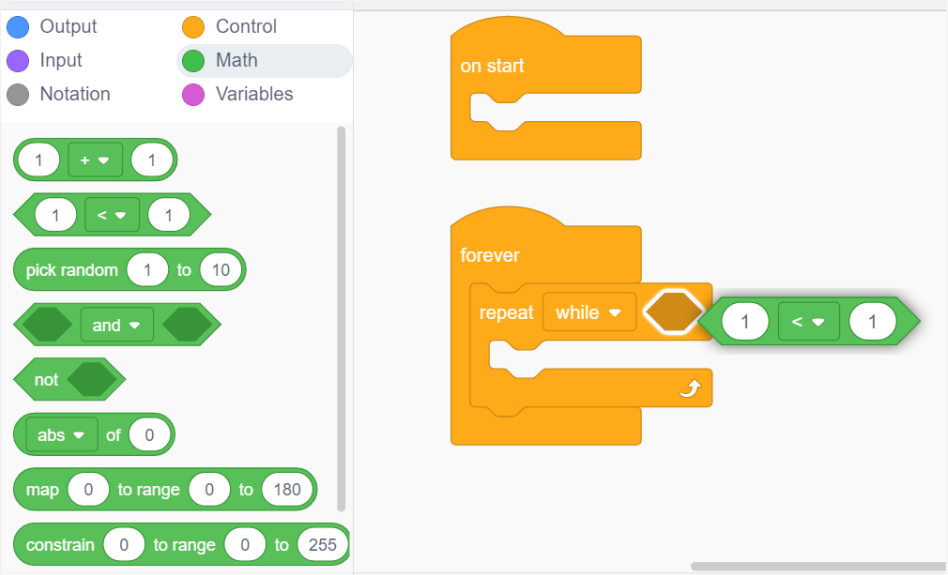
- **Siva grupa** sadrži blokove za komentare **(Slika 17)**.

17

Slika 17 - Raspoloživi blokovi grupisani po funkcionalnosti

Blokovi spram funkcionalnosti imaju različite oblike, gdje prate logiku standardnog blokovskog programiranja i spram funkcionalnosti su uklopivi u spojna mjesta predviđena za njih **(Slika 18).**

18



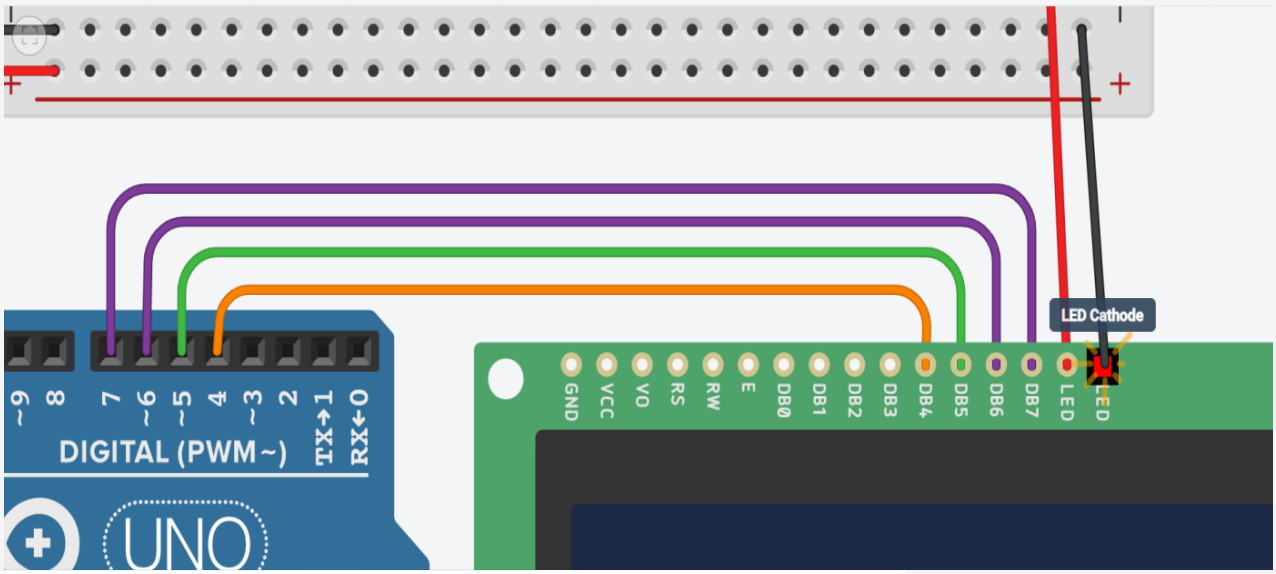
Slika 18 - Kodiranje pomoću blokova

Konkretni primjeri i način korištenja blokova detaljnije je predstavljeno u sljedećem poglavlju.

TINKERCAD KAO PREDAVAČKI ALAT

Arduino Uno pločica, elektroničke i druge komponente veoma su malih dimenzija. Da bi se uopće vidjelo kako izgleda određena elektronička komponenta, mora joj se prići bliže. Složeni sklopovi formirani su od mnogobrojnih komponenti spojenih sa velikim brojem provodnika što učenicama i učenicima koji nemaju preveliko iskustvo sa elektroničnim sklopovima djeluje presloženo. Predavač/ica koristeći web razvojnu platformu Autodesk Tinkercad može skupinu s kojom radi korak po korak voditi tako što sklop postepeno razvija u simulacionom okruženju, uz mogućnost zumiranja svakog dijela i svake razvojne komponente, gdje se pored standardnih natpisa i oznaka koje postoje na određenim komponentama pri pozicioniranju kursora na konektore i spojna mjesta pojavljuju dodatna objašnjenja **(Slika 19)**, a da pri tome učenice i učenici spajaju i razvijaju sklop u praksi.

19



Slika 19 - Tinkercad elementi korisni za predstavljanje

Mogućnost istovremene radne interakcije sa većom grupom učenica i učenika višestruko poboljšava efikasnost i produktivnost predavačkog rada, gdje cijela grupa može na projektnom platnu da prati razvoj i stvaranje sklopa, i iste korake da izvodi u praksi koristeći Arduino Uno i druge razvojne komponente. Realizacija nastavnih projekata, gdje je prvo potrebno spojiti shemu sklopa, a zatim pisati upravljački program, već kod izrade elementarnih primjera traži kvalitetno vremensko planiranje nastavnih aktivnosti. Kada se pred učenike i učenice stavi gotova shema imalo složenijeg sklopa, njima treba vremena da uoče gdje koji provodnik trebaju spojiti, često griješe i spajanje traje duže. Kada predavač/ica korak po korak sklop razvija i objašnjava u simulacionom okruženju, a da pri tom učenici i učenice spajaju sklop koristeći Arduino Uno razvojnu platformu u učionici, broj shematskih učeničkih grešaka svodi se na minimum, a razvoj i spajanje sklopa traje skoro duplo kraće.

Škole imaju ograničen broj Arduino setova i ograničen broj razvojnih komponenti i iste setove koriste za rad sa više grupa, što implicira da se projekti i vježbe moraju završiti u planiranom vremenu, što je najčešće jedan školski čas (45 minuta) ili blok čas (90 minuta), što je, ukoliko je

izvodivo u praksi, poželjno vrijeme za rad sa razvojnim Arduino platformama. Autodesk Tinkercad, kada se koristi kao predavački alat za postepeni **(step by step)** interaktivni razvoj i spajanje sklopova, omogućava zahvaljujući produktivnosti i većoj brzini pri razvoju sklopa da se u predviđenom vremenu spoje, a zatim kodiraju složeniji sklopovi, što sigurno ne bi bilo moguće kada bi ga učenici i učenice spajali samostalno na osnovu gotove sheme sklopa.

TINKERCAD KAO PREDAVAČKI ALAT - PRIMJERI

Resursno korištenje Tinkercada kao predavačkog alata prikazano je primjerima **Sedam-segmentni displej** i **RGB led** i **servo motor.** Primjeri prikazuju postepeni razvoj sklopa i upravljačkog koda. Za odabrane primjere upravljački kod paralelno je prikazan blokovski i tekstualno. Ubrzan razvoj u IT sektoru znači stalan softverski razvoj; svakodnevno se pojavljuju novi softveri, nova razvojna okruženja. Da bi se uopće radilo i opstalo u IT sektoru nužno je cjeloživotno učenje. Paradigmatski pristup učenju podrazumijeva prepoznavanje ključnih pojmova, principa i primijenjene logike koja varira spram konkretne primjene i radno-razvojnog okruženja.

Savremeni kurikulumi i nastavni planovi i programi za nastavni predmet Informatika za oblast programiranja u osnovnoj školi, srednjoj školi i na fakultetima kreiraju se da bi se na početku proceduralnom, a zatim i objektno orijentisanom pristupu prilazilo iz više perspektiva. Da budemo precizniji, u periodu ranog osnovnog obrazovanja (period razredne nastave i šesti razred predmetne nastave) učenici i učenice kodiraju koristeći blokovsko radno okruženje gdje se periodično susreću sa različitim blokovskim programskim jezicima, a zatim se uvodi paralela između strukture složenih blokova i tekstualnog koda. Nakon toga slijedi uvod u tekstualno kodiranje na proceduralnom nivou, gdje se učenice i učenici prvo susreću sa programskim jezicima čija je sintaksa „više pitka“ (npr. Python), a potom prelaze u radno okruženje programskih jezika čija je sintaksa složenija (npr. C++), da bi se u periodu izučavanja programiranja u srednjoj školi i na fakultetima po istom principu približili objektno orijentisanom pristupu.

Kada se analiziraju naredbe za kontrolu toka programa, npr. naredbe iteracije, petlja će ponavljati linije koda ili strukturu povezanih blokova sve dok ne bude ispunjen uvjet (definiran u zaglavlju). Ovdje se može uočiti da se u zaglavlju petlje postavlja uvjet koji definira koliko će se puta kod u tijelu petlje ponavljati.

Petlje se koriste u svim okruženjima gdje se kodira pomoću blokova, a koriste se i u svim linijskim programskim jezicima. Blokovi iteracije mogu imati različit oblik. Osim toga, programski jezici imaju vlastitu sintaksu i semantiku. Kada učenici i učenice na svom obrazovnom putu imaju iskustven dodir sa programskim petljama stvaranim pomoću različitih blokova na različitim platformama i pisanim u različitim programskim jezicima različitom sintaksom i semantikom, spoznat će da se ključni principi prožimaju u svim razvojnim okruženjima, da petlja bez obzira na način na koji je napisana vrši ponavljanja do ispunjenja definiranog uvjeta. Kao što voćni sok kada ga iz boce natočimo u čašu ne prestaje biti voćni sok, tako ni ključne programske naredbe ne prestaju biti to što jesu kada se koriste u različitim programskim jezicima (metode, konstruktori, destruktori su funkcije koje logički isto funkcioniraju, ali se na prilagođen način definiraju, koriste i pozivaju). Kada djevojke i mladići po završetku formalnog obrazovnog puta izađu na tržište rada, njih će vjerovatno čekati novo razvojno okruženje, novi programski jezik koji će oni bez većih poteškoća usvojiti, ako je njihov formalni obrazovni put paradigmatski iskustveno popločan na prethodno opisan način.

Obrazovna paradigma zasnovana na ishodima učenja i STE(A)M obrazovnim principima za period osnovnog i srednjeg obrazovanja prati dobne indikatore. Prethodno je opisan primjer u kojem je analizirana naredba kontrole toka programa (iteracija). Predstavljen je jedini način da prema

planu učenici i učenice usvoje indikator ishoda učenja: „4.c. Razlikuje i primjereno koristi naredbe uvjeta, logičke operatore i petlje“ (APOSO, ZJNPP za tehniku i IT, 2016) na kraju osnovnog obrazovanja. Programiranje u sebi sadrži za učenike i učenice zahtjevne logičke cjeline, koje se usvajaju postavljanjem početne baze (npr. naredbe za kontrolu toka programa iskustveno blokovski izučavane), a zatim se godinu za godinom ta baza nadograđuje i širi (naredbe za kontrolu toka programa linijski u programskim jezicima korisnicima bližim i jednostavnijim po sintaksi i semantici, zatim u sl. programskom jeziku i konkretnoj projektnoj primjeni (npr. Arduino projekti)). Autor publikacije je kao recenzent prepoznao na više zvaničnih dokumenata kurikularne nedostatke i otklonio ih u NPP programu za informatiku u osnovnim školama Brčko distrikta (Grupa autora, 2022).

Sedam-segmenti displej

Za realizaciju projekta neophodne su sljedeće komponente:

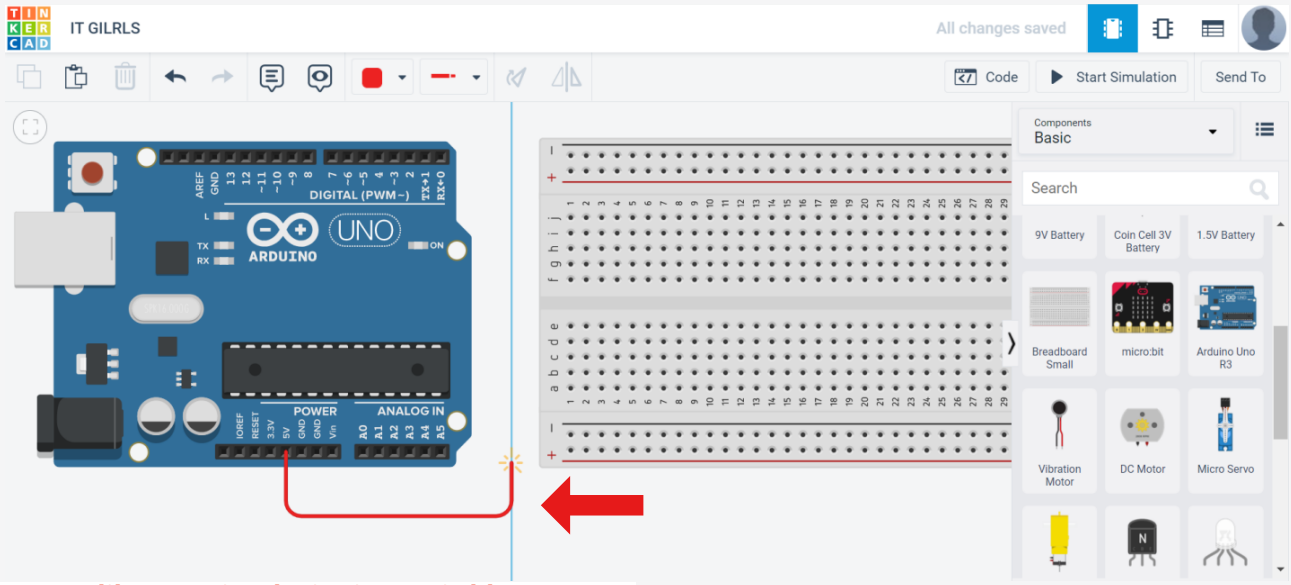
20

Name	Quantity	Component
U1	1	Arduino Uno R3
Digit1	1	Cathode 7 Segment Display
R1 R2	2	220 Ω Resistor

Slika 20 - Projekat sedam-segmentni displej – lista komponenti

Pozicioniranjem kursora na željeni element koji se nalazi u pomjerljivom izborniku komponenata na desnoj strani radnog prozora, držanjem lijevog tastera miša i premještanjem komponente na željeno mjesto na radnoj površini razvija se sklop. Na **Slici 21** prikazano je kako se pomjeranjem kursora i klikom na mjesto gdje provodnik želimo saviti on vodi do spojnog mjesta na matador pločici.

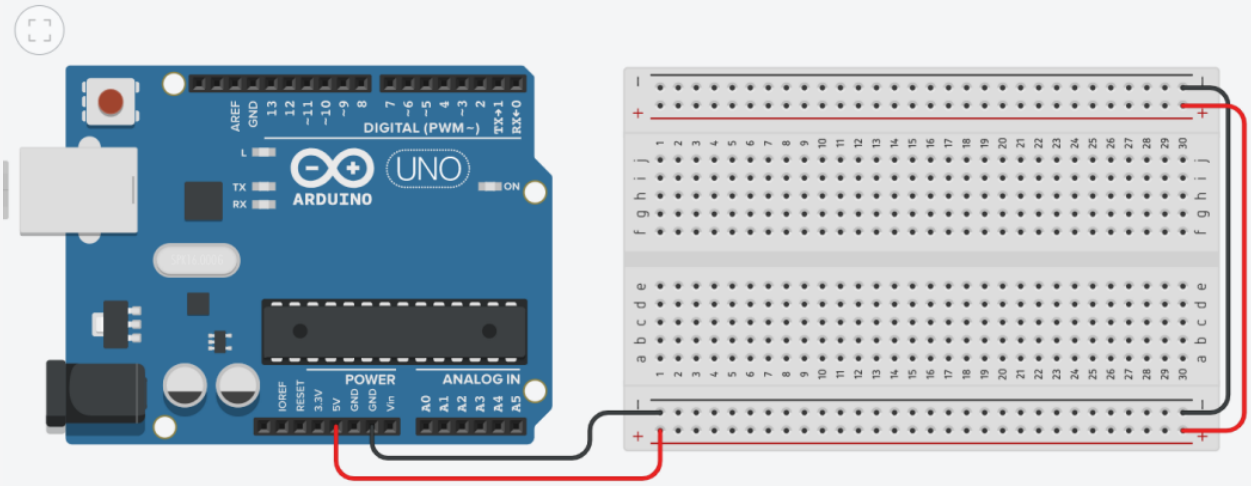
21



Slika 21 - Simulacioni razvoj sklopa

Na isti način povučeni su i spojeni provodnici na plus i minus liniju matador pločice s obje njene strane **(Slika 22).**

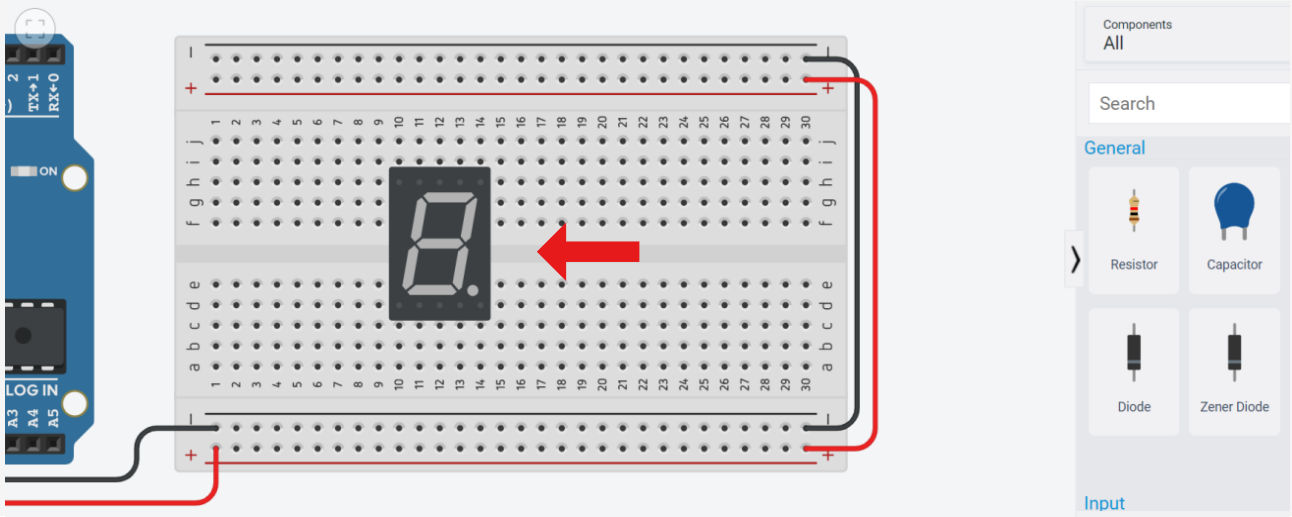
22



Slika 22 - Simulaciono spajanje vodiča na matador

U sljedećem koraku iz menija komponenti unosi se s**edam-segmentni displej**. Postavljamo ga tako da mu je DP tačka u donjem desnom uglu. Gledajući s lijeve strane, prvu nožicu sedam-segmentnog displeja postavljamo u kolonu **10**, gornje nožice idu u red **h**, a donje u red **d (Slika 23)**.

23

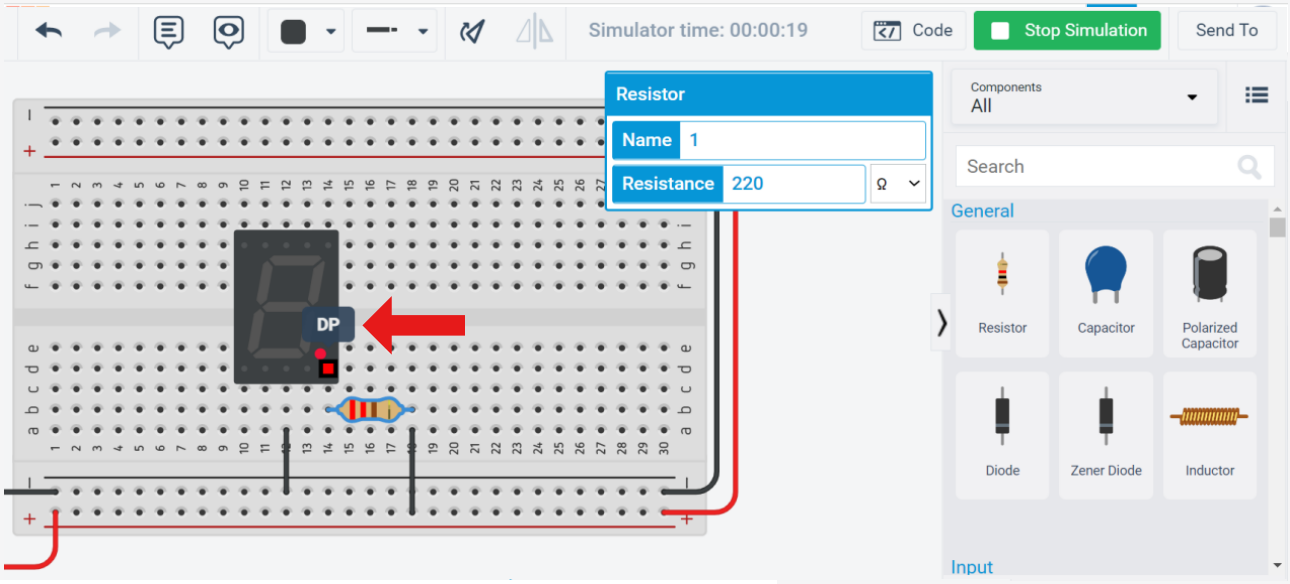


Slika 23 - Precizno pozicioniranje komponenti korištenjem koordinata na matador pločici

Sedam-segmentni displeji mogu biti proizvedeni u spoju sa zajedničkom katodom ili zajedničkom anodom. LED diode imaju radni napon između 2 i 4 V, te je zbog toga u strujni krug potrebno postaviti električni otpornik. U sedam-segmentni displej se prilikom proizvodnje ugrađuje osam LED dioda. Prvo što trebamo uraditi jeste provjeriti izvedbu sedam-segmentnog displeja. Iz menija komponenti iznosi se električni otpornik, klikom na dugme za rotaciju otpornik se rotira u horizontalni položaj. Pri selekciji otpornika pojavljuje se plutajući meni gdje navodimo ime otpornika (1), vrijednost otpornika (220) i mjernu jedinicu (Ω). Bira se vrijednost 220 Ω jer se ti otpornici nalaze u standardnim Arduino setovima. Otpornik zatim postavljamo u red **b**, kolonu **14** gdje se nalazi tačka i kolonu **18** na matador pločici. Kolonu **18** spajamo sa plus linijom, a kolonu **12** u koju je spojen zajednički izvod (katoda ili anoda) sa minus linijom. Ovim smo formirali jednostavan strujni krug i učenici koji su prateći korake spajali stvarne komponente trebaju samo priključiti USB kabl na port računara i na Arduino pločicu.

Ako je tačka na sedam-segmentnom displeju zasvijetlila, onda je on proizveden u spoju sa zajedničkom katodom, a ako ne zasvijetli radi se o spoju sa zajedničkom anodom ili o neispravnom displeju. Tinkercad klikom na sedam-segmentni displej omogućava njegovo podešavanje u spoj sa zajedničkom katodom ili zajedničkom anodom, te se ovaj spoj formira isključivo zbog provjere displeja u praksi. Klikom na link **Start simulation** pokreće se simulacija. Na **Slici 24** vidljivo je da DP tačka svijetli, odnosno sedam-segmentni displej podešen je u spoj sa zajedničkom katodom.

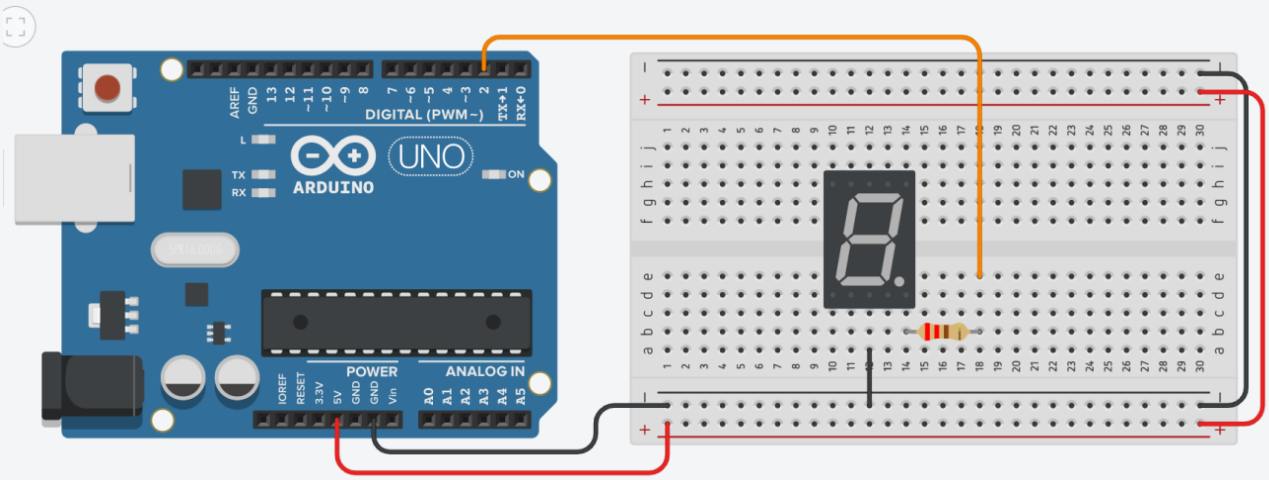
24



Slika 24 - Provjera tipa Sedam-segmentnog displeja

Kada je utvrđeno u kojem je spoju sedam-segmentni displej, briše se provodnik koji spaja kraj električnog otpornika sa plus linijom (učenici i učenice sklanjaju provodnik sa matador pločice) i iz iste kolone **(18)** povlači se provodnik koji se spaja na digitalni pin 2 Arduino pločice **(Slika 25).** Sada je moguće napisati upravljački kod za kreirani sklop. U dijelu koji slijedi opisan je način kreiranja jednostavnog blokovskog koda za kreirani sklop. Učenici i učenice nižih razreda osnovne škole upravljačke programe za kreirane sklopove slažu koristeći blokove. Da bi se primjerom potkrijepio prethodno opisan kontinuirani proces usvajanja indikatora ishoda učenja, njegova nadogradnja, kao i prepoznavanje prožimajuće logike, paralelno sa kodom stvorenim pomoću blokova prikazan je identičan tekstualni kod.

25



Slika 25 - Sedam-segmentni displej (spoj sa zajedničkom katodom)

Na lijevoj strani **Slike 26** postavljen je on start blok koji se pokreće pri pokretanju simulacije. Analogno njemu s desne strane nalazi se funkcija `setup()`, gdje se, kako ime funkcije i ukazuje, pišu početne postavke koje je potrebno podesiti samo na startu (npr. postavka pinova na ulazni ili izlazni režim rada). Upravljački programi učitani na mikrokontroler po pravilu se nalaze u režimu neprekidnog rada. Navođenjem primjera primjene mikrokontrolera jasno je i zašto je tako:

- poziv lifta uvijek treba biti moguć,
- brava na ulazu u zgradu uvijek se odgovarajućim pinom mora moći otključati i sl.

26


```
void setup()
{

}

void loop()
{

}
```

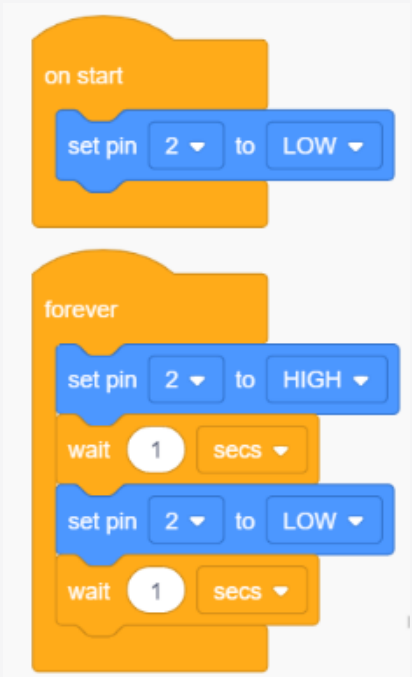
Slika 26 - Paralelan razvoj blokovskog i tekstualnog koda (korak 1)

Iz navedenih primjera jasno je zašto je u programskom kodu obavezan ***forever*** blok, odnosno beskonačna petlja `loop()` u Arduino IDE okruženju.

Na **Slici 27** prikazan je upravljački kod pomoću kojeg će se DP tačka na sedam-segmentnom displeju beskonačno paliti i gasiti sa stankom od jedne sekunde. Na lijevoj strani u ***on start*** bloku postavljen je ***set pin*** blok iz plave skupine ***Output.*** Na desnoj strani u funkciji `setup()` moralo se korištenjem funkcije `pinMode(2, OUTPUT)` na poziciji prvog argumenta navesti broj digitalnog pina te čiji se režim rada postavlja i u kojem će režimu raditi – izlazni režim `(OUTPUT).` Zatim se funkcijom `digitalWrite(2, LOW)` digitalni pin 2 postavlja na nisko naponsko stanje `(LOW)`.

U ***forever*** blok postavljena su dva ***set pin*** bloka gdje je prvi postavljen na visoko naponsko stanje, a drugi na nisko naponsko stanje. Ispod svakog set pin bloka postavljena je naredba ***wait*** kojom se prethodno postavljena naredba „zaledi“ na zadano vrijeme. Na desnoj strani tok programa se „zaledi“ korištenjem funkcije `delay(1000)`.
Nakon paralelne analize programskih kodova realiziranih pomoću blokova i linijskog koda, vidljivo je da je korištenjem blokova automatiziran proces određivanja režima rada `(OUTPUT)`, da je eliminirana mogućnost pogreške u pisanju koda, jer se blokovi samo iznesu i postave na željeno mjesto. Međutim, logika programa je ista. Na početku se izvršava ***on start*** blok, a zatim se program beskonačno ponavlja u ***forever*** bloku. Na drugoj strani isto rade funkcija `setup()` i `loop()`.

27

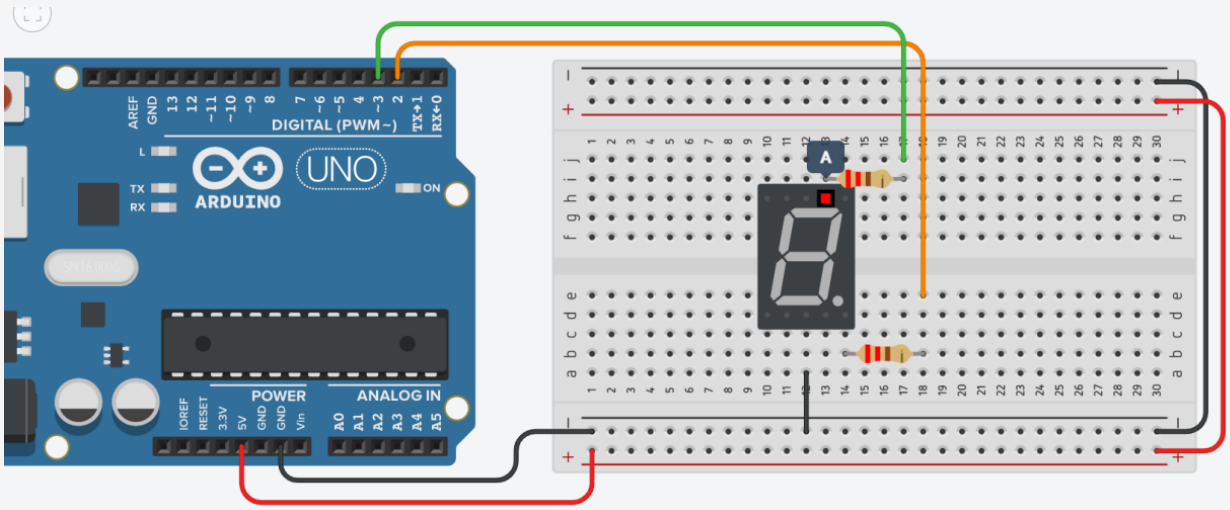



```
void setup()
{
  pinMode(2, OUTPUT);
  digitalWrite(2, LOW);
}

void loop()
{
  digitalWrite(2, HIGH);
  delay(1000);
  digitalWrite(2, LOW);
  delay(1000);
}
```

Slika 27 - Paralelan razvoj blokovskog i tekstualnog koda

Učenice i učenici koji su logiku beskonačne petlje usvojili pomoću blokova sad samo trebaju da se usmjere na sintaksu koda jer logički program radi na isti način. Linijska struktura im je odmah jasnija jer su upoznati sa logikom funkcioniranja korištene petlje.

28



Slika 28 - Paralelan razvoj blokovskog i tekstualnog koda (korak 3)


```
void setup()
{
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);

void loop()
{
  digitalWrite(2, HIGH);
  delay(1000);
  digitalWrite(2, LOW);
  digitalWrite(3, HIGH);
  delay(1000);
  digitalWrite(3, LOW);
}
```

Slika 29 – Paralelan razvoj blokovskog i tekstualnog koda (korak 4)

U razvojno okruženje sklopa **(Slika 28)** unosi se još jedan otpornik čija vrijednost također iznosi **220 Ω**. Njegovu lijevu nožicu postavljamo na koordinatu ***i13*** matador pločice. Drugu nožicu otpornika spajamo na koordinatu ***i17*** te iz kolone ***17*** matador pločice vodi se provodnik i spaja na digitalni pin 3 Arduino pločice.

Programski kod se proširuje naredbom ***set pin*** za pin 3 u ***on start*** bloku, odnosno sa po dva ***set pin*** bloka i ***wait*** bloka u ***forever*** bloku, čime se po pokretanju simulatora DP tačka jednu sekundu pali, zatim jednu sekundu gasi, a potom se segment A jednu sekundu pali, a zatim jednu sekundu gasi i sve se u beskonačnoj petlji ponavlja. Na desnoj strani na isti način se proširuje linijski kod.
Učenici i učenice koristeći istu šablonsku logiku sklop i programski kod mogu samostalno proširiti na sve segmente sedam-segmentnog displeja, tako da se u nizu (DP, A, B, C, D, E, F, G) redom pale i gase.

RGB led i servo motor

Lista komponenti potrebnih za realizaciju projekta:

30

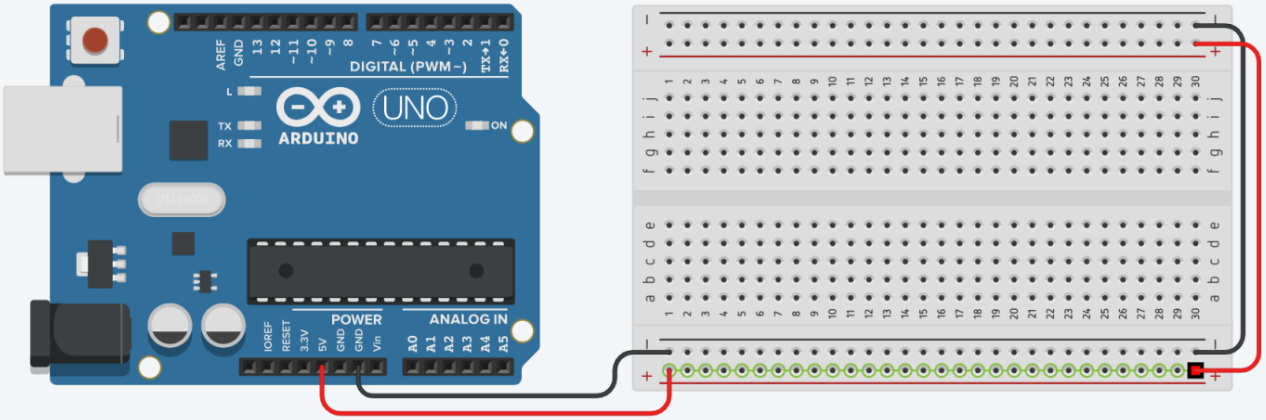
Name	Quantity	Component
U1	1	Arduino Uno R3
D1	1	LED RGB
R1 R2 R3	3	220 Ω Resistor
SERVO1	1	Micro Servo

Slika 30 – Lista komponenti za projekat RGB led i servo motor

Zadatak predviđa rotiranje **servo motora** od startnih **0⁰**, pri čemu **RGB led dioda** ne svijetli, zatim za **45⁰** gdje **RGB led dioda** svijetli prvom odabranom bojom, za **90⁰** gdje **RGB led dioda** svijetli drugom odabranom bojom i za **180⁰** gdje **RGB led dioda** svijetli trećom odabranom bojom te se na kraju **servo motor** vraća na početnih **0⁰**, pri čemu **RGB led dioda** ne svijetli.

Učenice i učenici na projektnom platnu prate postepen razvoj sklopa i istovremeno spajaju stvarne komponente koje se nalaze ispred njih. Prvi korak je uvijek dovođenje plusa i minusa sa Arduino pinova na plus i minus liniju na matador pločici **(Slika 31)**.

31

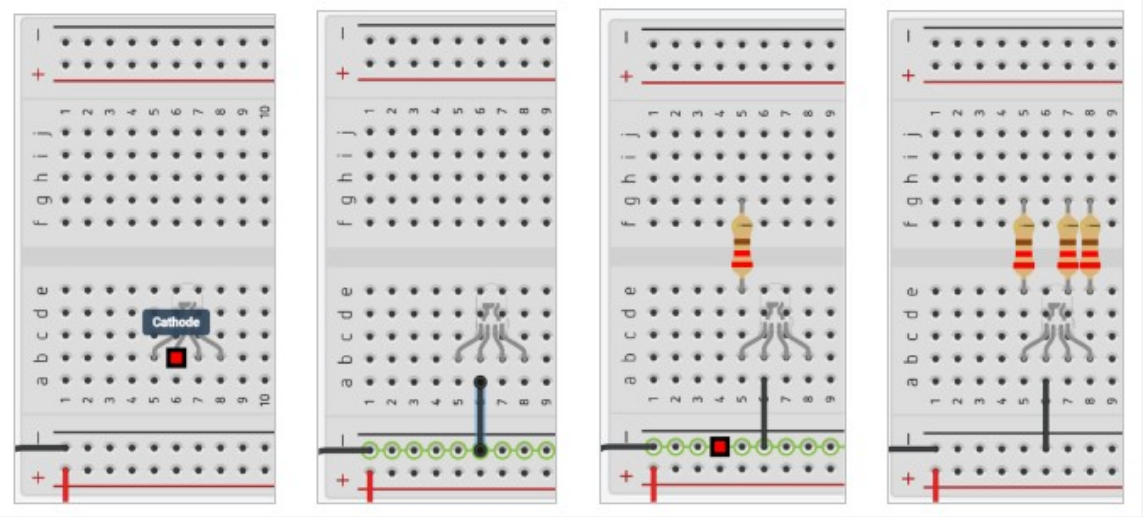


Slika 31 – Simulacioni razvoj projekta 2 (korak 1)

Iz menija komponenti iznosi se **RGB led dioda** i na matador pločici spaja se tako da se u redu ***b*** i koloni ***6*** spoji katoda (najduža nožica na **RGB led diodi**), u istom redu lijevo (kolona ***5***) spoji se nožica za crvenu boju i desno u kolonama ***7*** i ***8*** nožice za plavu i zelenu boju **(Slika 32)**. Katoda (kolona ***6***) se spaja s minus linijom **(Slika 33)**, a zatim se iznosi otpornik čija se otpornost podešava na **220 Ω** i spaja u kolonu gdje se nalazi crvena nožica (kolona ***5***) i s druge strane preko razdjelnika matador

pločice električni otpornik se spoji drugom nožicom u istu kolonu **(Slika 34)**. Postupak se ponovi na isti način sa dva nova otpornika iste otpornosti u kolonama ***7*** i ***8*** za spoj sa plavom i zelenom nožicom **(Slika 35)**.

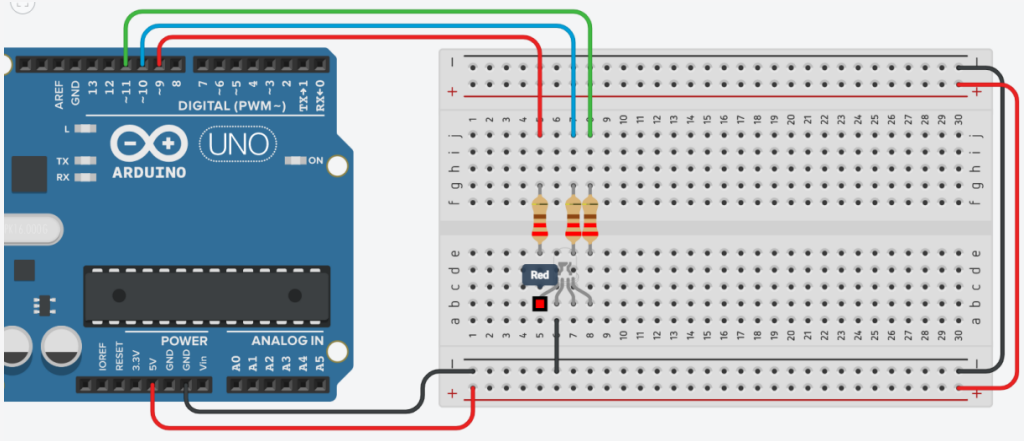
32
33
34
35



Slika 32 – (korak 2) Slika 33 – (korak 3) Slika 34 – (korak 4) Slika 35 – (korak 5)

Sada se crvena nožica **RGB led diode** preko električnog otpornika **R1** iz kolone ***9*** provodnikom spaja na digitalni pin ***9*** na Arduino pločici. Na isti način se plava i zelena nožica **RGB led diode** preko električnih otpornika **R2** i **R3** spaja na digitalne ulaze **10** i **11** na Arduino pločici **(Slika 36)**. Učenici i učenice su na nastavi iz likovne kulture naučili da se miješanjem tri osnovne boje (crvena, plava i zelena) mogu dobiti sve druge. Zbog toga se za spajanje **RGB led diode** moraju koristiti digitalni PWM (Pulse Width Modulation) koji imaju mogućnost modulacije širine pulsa (PWM na Arduino pločici označeni su simbolom „ ~ “). Pulsno-širinska modulacija nudi mogućnost korištenja raspona vrijednosti od **0** do **255** te se na taj način korištenjem funkcije `analogWrite(9, 255)` na svakom od spojenih pinova mogu podešavati vrijednosti čijim se miješanjem dobije željena boja.

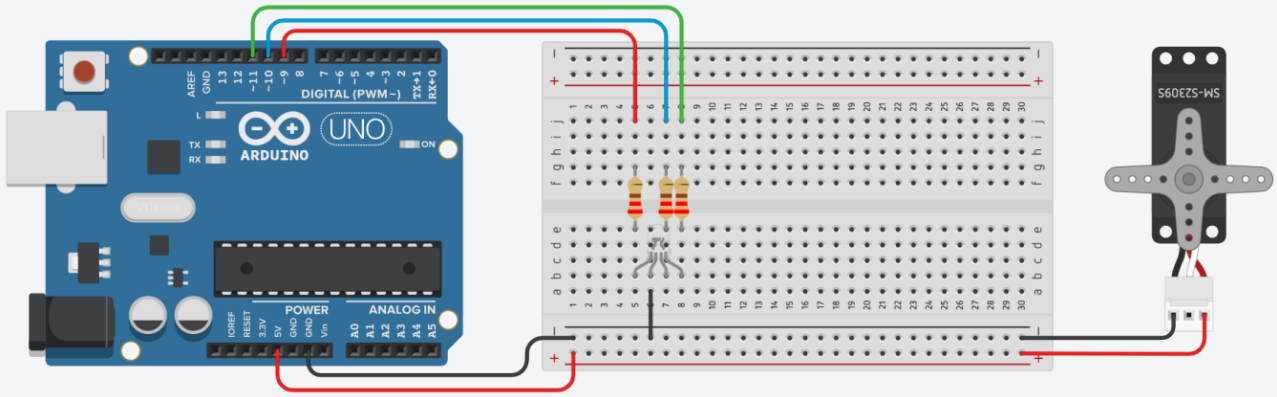
36



Slika 36 – Simulacioni razvoj projekta 2 (korak 6)

Preostaje još da se na sklop poveže servo motor. Iz menija sa komponentama ubacuje se **servo motor** (Micro Servo: SM – S2309S), a njegovi plus (crvena boja provodnika) i minus (crna boja provodnika) pinovi spajaju se provodnicima sa plus i minus linijama na matador pločici **(Slika 36).**

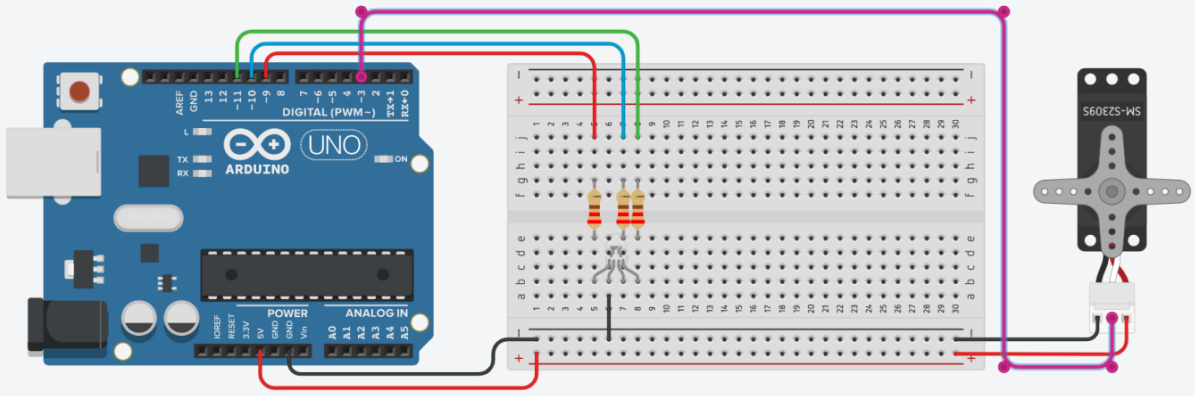
37



Slika 37 – Simulacioni razvoj projekta 2 (korak 7)

Srednji pin sa servo motora spaja se na digitalni pin **3** na Arduino pločici **(Slika 37)**. Servo motor također se mora spojiti na pin koji ima mogućnost pulsno-širinske modulacije jer se ugao servo motora koji dolazi u standardnim Arduino setovima može pomjerati od **0⁰** do **180⁰**.

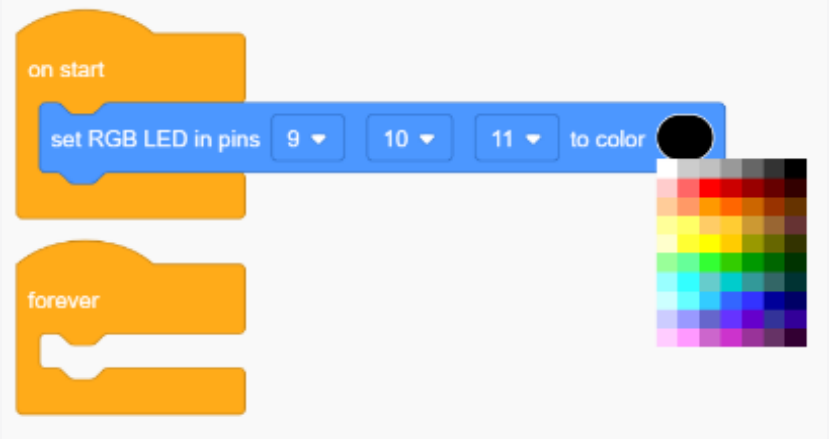
38



Slika 38 – Simulacioni razvoj projekta 2 (korak 8)

Nakon što je spajanje sklopa dovršeno, potrebno je napisati upravljački kod **(Slika 39).** Na lijevoj strani se u ***on start*** blok unosi plavi izlazni blok, gdje se biraju pinovi na koje su spojene crvena, plava i zelena nožica **RGB led diode**, te se bira crna boja iz palete boja (kada je odabrana crna boja **RGB led dioda** ne svijetli). Na desnoj strani u funkciji `setup()` funkcijama `pinMode(9, OUTPUT)` pinovi na koje je spojena **RGB led dioda** postavljaju se u izlazni režim rada. Zatim se funkcijama `analogWrite(9, 0)` na poziciju drugih argumenata pišu nule i tada **RGB led dioda** ne svijetli **(Slika39).**

39



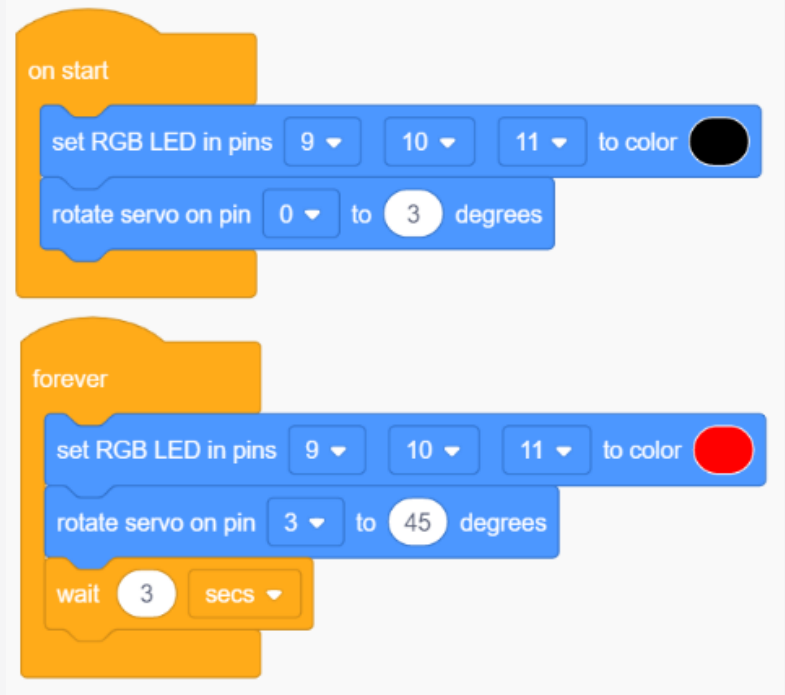
Slika 39 – Paralelno blokovsko i linijsko kodiranje za projekat 2 (korak 1)

```
void setup()
  {
    pinMode(9, OUTPUT)
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);
  }

void loop()
  {

  }
```

Na lijevoj strani u ***on start*** blok unosi se plavi izlazni blok za kontrolu **servo motora** i u njemu se podešava priključni pin **(3)** i početni nulti ugao **(Slika 40).** Tekstualni kod je složeniji te se na početku mora uključiti biblioteka **`Servo.h`** preprocesorskom naredbom **`#include<Servo.h>`**. U biblioteci je napisana klasa **Servo** koja sadrži metode za upravljanje **servo motorom**. U drugoj liniji koda za klasu **`Servo`** instancira se objekat koji je nazvan **mojServo.** Nakon toga se u funkciji **`setup()`** za objekat **mojServo** pomoću pristupnog operatora **`(.)`** poziva metod **`attach(3)`** pomoću kojeg se na pin **3** Arduino pločice zakači signalni pin **servo motora.** Pozivom metoda **`mojServo.write(0)`**, u funkciji **`setup()`** na startu se servo motor pozicionira na nulti ugao.

U ***forever*** bloku mijenja se boja na crvenu i ugao servo motora se pomjera za **45⁰** te se blokom ***wait*** proces zaledi na 3 sekunde. U funkciji **`loop()`** vrijednost crvenog pina **RGB led diode** postavljena je na **255**, a zeleni pin i plavi pin ostaju na vrijednosti **0** i ta kombinacija daje crvenu boju na **RGB led diodi.** Nakon što **RGB led dioda** zasvijetli crvenom bojom linijom koda **`mojServo.write(45)`** servo motor se pomjera za ugao od **45⁰** te se proces funkcijom **`delay(3000)`** zaustavi na 3 sekunde.

40



Slika 40 – Paralelno blokovsko i linijsko kodiranje za projekat 2 (korak 2)

```
#include<Servo.h>
Servo mojServo;
void setup()
  {
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);
    mojServo.attach(3);
    mojServo.write(0);
  }
void loop()
  {
    analogWrite(9, 255);
    analogWrite(10, 0);
    analogWrite(11, 0);
    delay(3000);
    mojServo.write(45);
  }
```

Na već opisani način proširujući kod sa tri prethodno korištena bloka, **RGB led dioda** se postavlja na novu boju, servo motor se pomjera za **45⁰** i proces se opet zaledi na 3 sekunde **(Slika 41).**

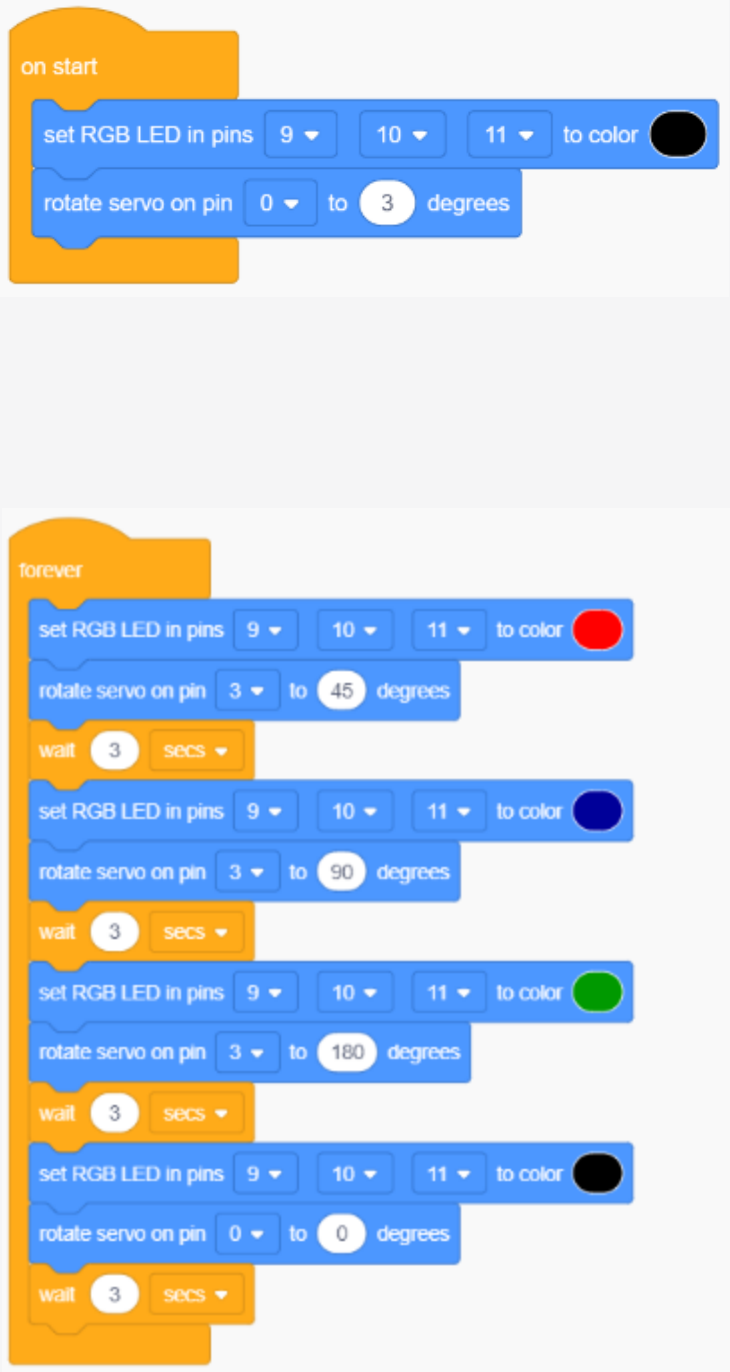
41

Slika 41 – Paralelno blokovsko i linijsko kodiranje za projekat 2 (korak 3)

```
#include<Servo.h>
Servo mojServo;
void setup()
  {
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);

    mojServo.attach(3);
    mojServo.write(0);
  }

void loop()
  {
    analogWrite(9, 255);
    analogWrite(10, 0);
    analogWrite(11, 0);
    delay(3000);
    mojServo.write(45);

    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 153);
    delay(3000);
    mojServo.write(90);
  }
```

Po istoj šablonskoj logici učenici i učenice samostalno slažu blokove ili pišu tekstualni kod kojim dioda zasvijetli novom odabranom bojom, **servo motor** se pomjera na **180⁰** i proces pauzira tri sekunde. Na kraju koda se **RGB led dioda** postavi na stanje kada ne svijetli, a **servo motor** se vraća na početni nulti ugao **(Slika 42).**

42



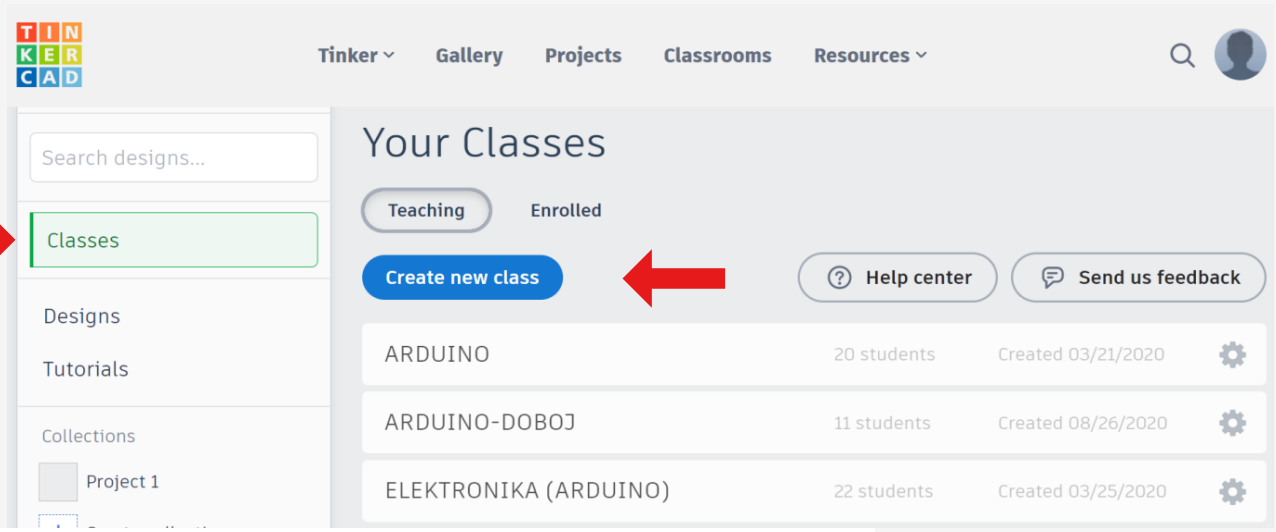

```
#include<Servo.h>
Servo mojServo;
void setup()
 {
    pinMode(9, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(11, OUTPUT);
    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);
    mojServo.attach(3);
    mojServo.write(0);
  }

void loop()
  {
    analogWrite(9, 255);
    analogWrite(10, 0);
    analogWrite(11, 0);
    delay(3000);
    mojServo.write(45);

    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 153);
    delay(3000);
    mojServo.write(90);

    analogWrite(9, 0);
    analogWrite(10, 153);
    analogWrite(11, 0);
    delay(3000);
    mojServo.write(180);

    analogWrite(9, 0);
    analogWrite(10, 0);
    analogWrite(11, 0);
    delay(3000);
    mojServo.write(0);
  }
```

Slika 42 – Paralelno blokovsko i linijsko kodiranje za projekat 2 (korak 4)

AUTODESK TINKERCAD KAO ALAT ZA ONLINE NASTAVU I SAMOSTALAN UČENIČKI RAD

Edukatorice i edukatori mogu putem kreiranog korisničkog naloga za edukatore koristiti dodatne Tinkercad edukatorske resurse u vidu online virtuelnih učionica. Klikom na link **Classes** na desnoj strani web-preglednika **(Slika 43)**, ispod natpisa **Your Classes** (tvoji razredi) nalaze se linkovi **Teaching** (podučavaš) i **Enrolled** (upisan).

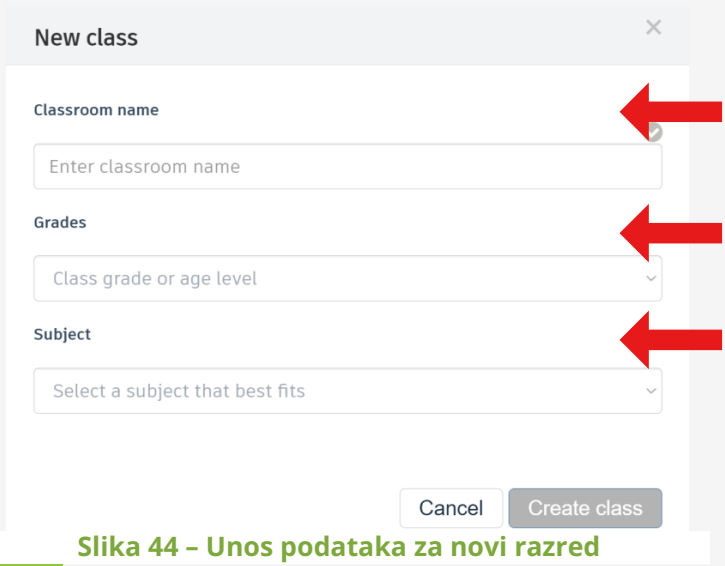
43



Slika 43 – Tinkercad učionice za učenike i nastavnike

Klikom na link **Teaching** pojavljuje se komandno link-dugme **Create new class**, a ispod njega se nalaze vremenskom hijerarhijom poredani pristupni linkovi koji vode prema kreiranim učionicama.
Odabirom linka **Create new class** pred edukatorima se pojavljuje plutajući meni **(Slika 44)** gdje unose ime razreda koji kreiraju. Opciono se mogu popunjavati polja **Grades** i **Subject**.
Klikom na textbox **Grades** edukatorica ili edukator ima mogućnost iz plutajućeg menija odabrati

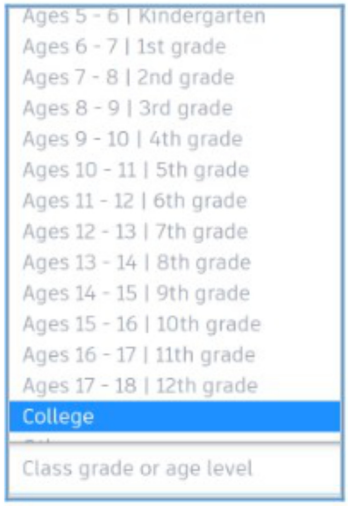
44



Slika 44 – Unos podataka za novi razred

učeničku dob ili odgovarajući razred **(Slika 45).** Kada je edukacioni kurs uže određen, klikom na textbox **Subject** **(Slika 46)** edukatorica ili edukator ima mogućnost da bira konkretnu oblast iz plutajućeg menija.

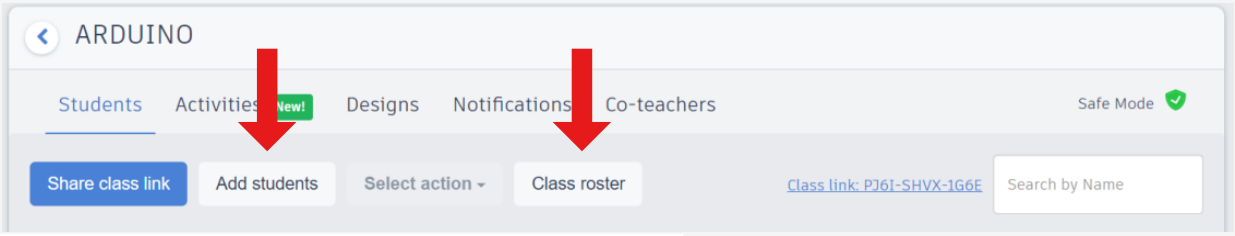
45
46



Slika 45 – Postavljanje dobnih kategorija **Slika 46 – Primarna oblast poučavanja**

Kada je razred kreiran, automatski je generiran pristupni kod **(Class Code)**, te se u razred mogu dodati učenici i učenice **(Add students)** i može se pogledati spisak razreda **(Class roster)**.

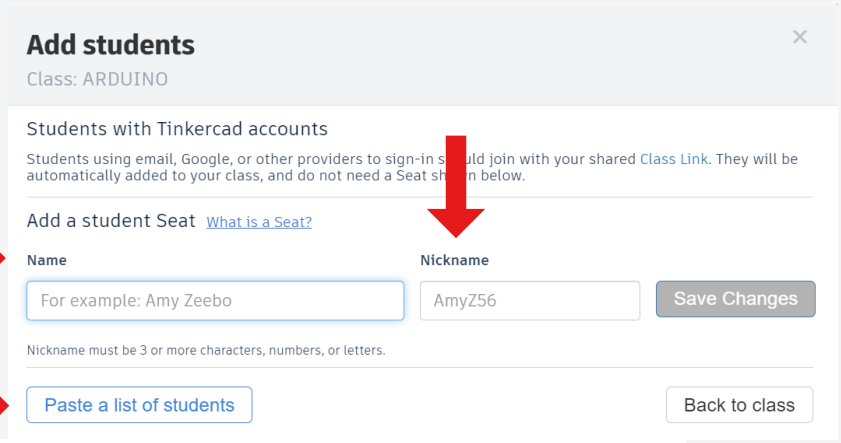
47



Slika 47 – Dodavanje novih učenika u razred

Kada se klikne na komandno link-dugme **Add students** **(Slika 47)**, pojavljuje se dijaloški okvir u kojem se unosi ime učenice ili učenika te se nakon unosa automatski generira pristupni **Nickname.** Ukoliko već postoji lista učenika i učenica, ona se u cijelosti može ***zalijepiti*** u okviru koji se otvara nakon klika na komandno link-dugme **Paste a list of students** **(Slika 48)**.

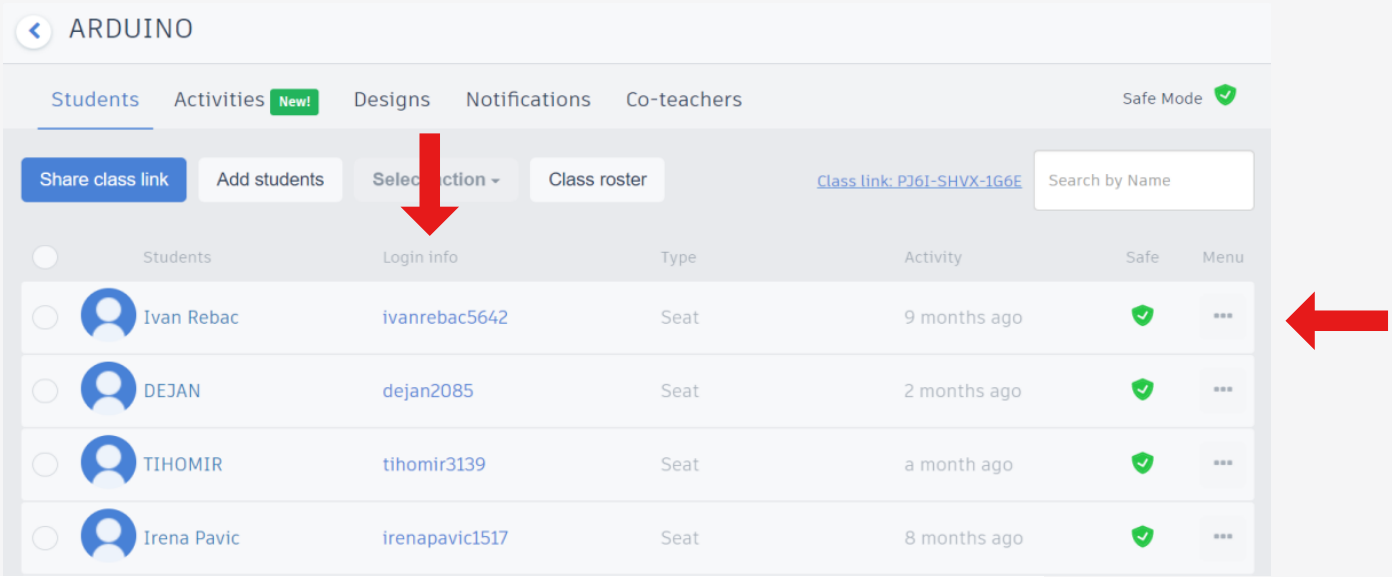
48



Slika 48 – Unos liste učenika i učenica

Nakon što je formiran razred, vidljiva je lista učenika i učenica, s podacima o njihovim imenima, **login** informacijama, njihovim aktivnostima. Klikom na tri tačkice koje se nalaze na kraju reda sa informacijama o učenici ili učeniku otvara se plutajući meni sa dodatnim opcijama, gdje se npr. može izbrisati učenik iz razreda ako ga više ne pohađa **(Slika 49).**

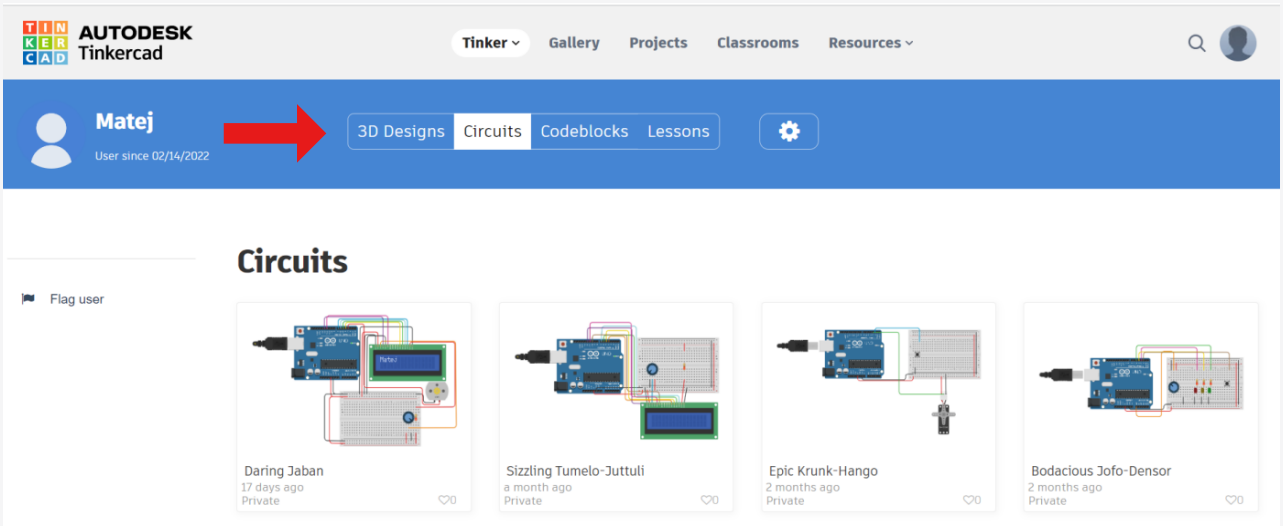
49



Slika 49 – Spisak učenika i učenica i njihovih pristupnih podataka

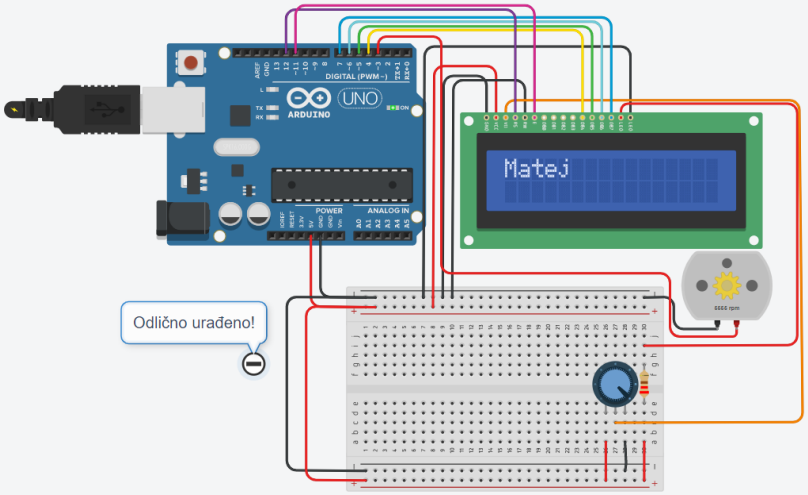
Kada edukatorica ili edukator klikne na učenicu ili učenika iz liste vidljiv je prozor sa učeničkim aktivnostima i urađenim vježbama i projektima razvrstanim u pripadajuće oblasti (**3D Designs**, **Circuits**, **Codebloks** i **Lessons**). Klikom na link **Circuits** edukatorici ili edukatoru vidljivi su Arduino projekti koje su učenici i učenice radili **(Slika 50)**.

50



Slika 50 – Edukatorski pristup učeničkim projektima

Učeničke projekte edukatorica ili edukator može editovati, popravljati ili učenicima ostavljati komentare ili izvještaje. Na **Slici 51** vidljiv je komentar edukatora: **„Odlično urađeno!“.**
Tokom pandemije COVID-19 rad sa učenicima u potpunosti je bilo moguće prebaciti u virtualno, simulaciono okruženje. Korištenjem virtualnih web-baziranih Tinkercad učionica omogućava se učenicama i učenicima da rad na razvoju sklopova i njihovom kodiranju ne završavaju u učionici, već da ga nastavljaju kod kuće



Slika 51 – Online interakcija edukatora/ica i učenika/ica

NAZIV PROJEKTA:

ROBOTSKA RUKA - MANIPULATOR

PREDMETI:

Informatika
Tehnička kultura
Fizika

Projekt se realizira u okviru vannastavnih aktivnosti, uključuje nastavne predmete informatika, tehnička kultura i fizika.

OPIS PROJEKTA:

Projekat predviđa izradu (montažu) Robotske ruke tipa: KSR10 Velleman i njenu kontrolu pomoću Arduina, H-mostova i drugih električnih komponenti. Učenice pod vodstvom nastavnica i nastavnika montiraju robotsku ruku, razvijaju upravljački sklop, pišu upravljački programski kod, testiraju ga u virtualnom Tinkercad okruženju. Realiziraju sklop u praksi, testiraju ga, prezentiraju, a kasnije koriste kao nastavno učilo.

PLANIRANE PROJEKTNE AKTIVNOSTI:

Aktivnost 1.
Informiranje članica IT Girls kluba o predstojećim projektnim aktivnostima.

Aktivnost 2.
Kratka motivaciona prezentacija u kojoj se učenicama predstavljaju primjeri primjene robotskog manipulatora u praksi (proizvodni sistemi, automobilska industrija i sl.).

Aktivnost 3.
Specifikacija potrebnih materijala za realizaciju projekta.

Aktivnost 4.
Podjela radnih zadataka između nastavnica i nastavnika uključenih u realizaciju projekta
(nastavnice i nastavnici informatike, tehničke kulture i fizike).

Aktivnost 5.
Razvoj sklopa u Tinkercad simulacionom okruženju
(spojne sheme).

Aktivnost 6.
Kodiranje sklopa u Tinkercad simulacionom okruženju.

Aktivnost 7.
Testiranje sklopa u Tinkercad simulacionom okruženju.

Aktivnost 8.
Montaža Robotske ruke tipa: KSR10 Velleman

Aktivnost 9.
Spajanje priključnih provodnika na robotsku ruku

Aktivnost 10.
Izrada sklopa kontroliranog Arduino UNO R3 mikrokontrolerima prema prethodno kreiranim shemama u Tinkercadu.

Aktivnost 11.
Povezivanje motora robotske ruke sa sklopom

Aktivnost 12.
Testiranje projekta

Aktivnost 13.
Prezentacija projekta i njegovo korištenje u nastavi kao nastavnog učila

PARCIJALNA UKLJUČENOST PREDMETA

Informatika	• Razvoj sklopa u simulacionom Tinkercad okruženju • Izrada upravljačkog programskog koda • Simulaciono testiranje projekta • Realizacija Arduino projekta u praksi • Prezentacija projekta
Tehnička kultura	• Montaža Robotske ruke tipa: KSR10 Velleman • Realizacija Arduino projekta u praksi • Spajanje robotske ruke s upravljačkim sklopom • Prezentacija projekta
Fizika	• Analiza odvojenih strujnih krugova na H-mostu (upravljački strujni krug i pogonski strujni krug) • Realizacija Arduino projekta u praksi • Prezentacija projekta

ISHODI UČENJA I DOBNI IDIKATORI

Informatika

Oblast: 5. Digitalno društvo
Komponenta: 5.1. Virtuelni svijet

Ishod učenja:
5.1.4. Primjenjuje digitalne tehnologije pri učenju.

Dobni indikator:
5.1.4.a. 4.b. Primjenjuje besplatne online alate za komunikaciju i saradnju pri izradi timskog zadatka.

Oblast: 4. Rješavanje problema primjenom IT-a
Komponenta: 2. Programiranje

Ishod učenja:
4.2.4. Analizira i povezuje elemente programiranja.

Dobni indikator:
4.2.4.a. Stvara programe koji provode algoritme za postizanje datih ciljeva.

Ishod učenja:
4.2.5. Rješava probleme upotrebom programskog jezika.

Dobni indikator:
4.2.5.e. Koristi skriptni jezik.

Tehnička kultura

Oblast: 2. Tehnika i tehnologija
Komponenta: 2.2. Savremene tehnologije

Ishod učenja:
2.2.4. Analizira složene tehničke sisteme.

Dobni indikator:
2.2.4.c. Objašnjava sisteme upravljanja, geometrijske principe kretanja i mehaničku osnovu robota.

Ishod učenja:
2.2.5. Primjenjuje savremene tehnologije.

Dobni indikator:
2.2.5.a. Koristi se računarom u tehničkom crtanju.

(APOSO, ZJNPP za tehniku i IT, 2016)

Oblast: 1. Tehničko znanje i stvaralaštvo
Komponenta: 1.4. Praktičan rad

Ishod učenja:
1.4.9. Izrađuje i sklapa različite modele samostalno ili prema tehničkoj dokumentaciji.

Dobni indikator:
1.4.9.e. Poznaje električne sheme i postupke spajanja elemenata u funkcionalnu cjelinu.
1.4.9.f. Sastavlja električna strujna kola prema zadanoj shemi.

Oblast: 2. Tehnika i tehnologija
Komponenta: 2.2. Savremene tehnologije

Ishod učenja:
2.2.5. Primjenjuje savremene tehnologije.

Dobni indikator:
2.2.5.a. Koristi se računarom u tehničkom crtanju.

Fizika

Oblast: 3. Elektromagnetizam
Komponenta: 3.2. Električna struja

Ishod učenja:
3.2.2. Sastavlja i evaluira strujna kola.

Dobni indikator:
3.2.2.b) Crta i tumači shemu strujnog kola sa serijski i/ili paralelno spojenim potrošačima, te sastavlja odgovarajuće realna i virtualna (simulacije) strujna kola.

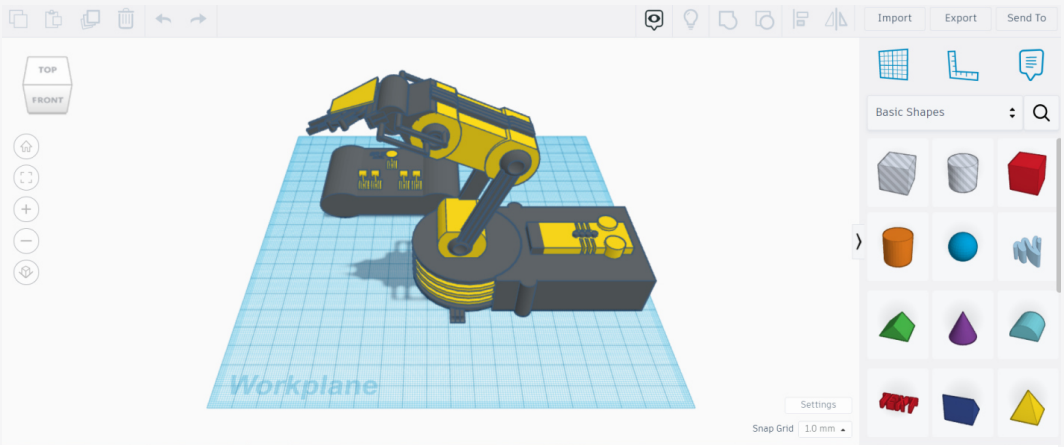
(APOSO, ZJNPP za fiziku, 2017)

OČEKIVANI REZULTATI PROJEKTA

STEM učilo koje se može koristiti na nastavi informatike, tehničke kulture i fizike.

Robotska ruka tipa: KSR10 Velleman skupa sa uputama za montažu komercijalno je dostupna na lokalnom tržištu. Može se uz postojeće upute printati pomoću 3D printera, što bi projekat kvalitativno podiglo na dodatni nivo (**Slika 52** (Tinker, 2016.)).

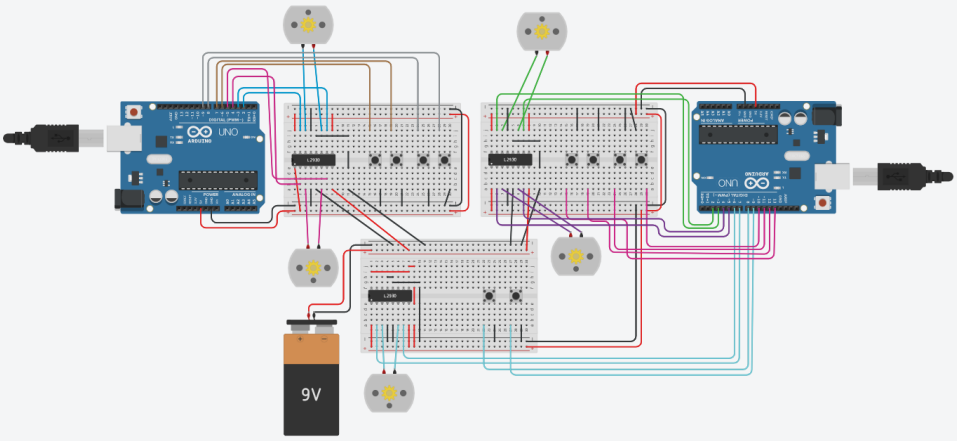
52



Slika 52 – KSR10 Velleman robotska ruka razvijena u Tinkercadu

Izgled testnog sklopa razvijenog u simulacionom okruženju Tinkercad prikazan je na **Slici 53.**

53



Slika 53 – Testni sklop razvijen u Tinkercadu

Pomoću Tinkercada automatski generirana lista komponenti prikazana je na **Slici 53:**

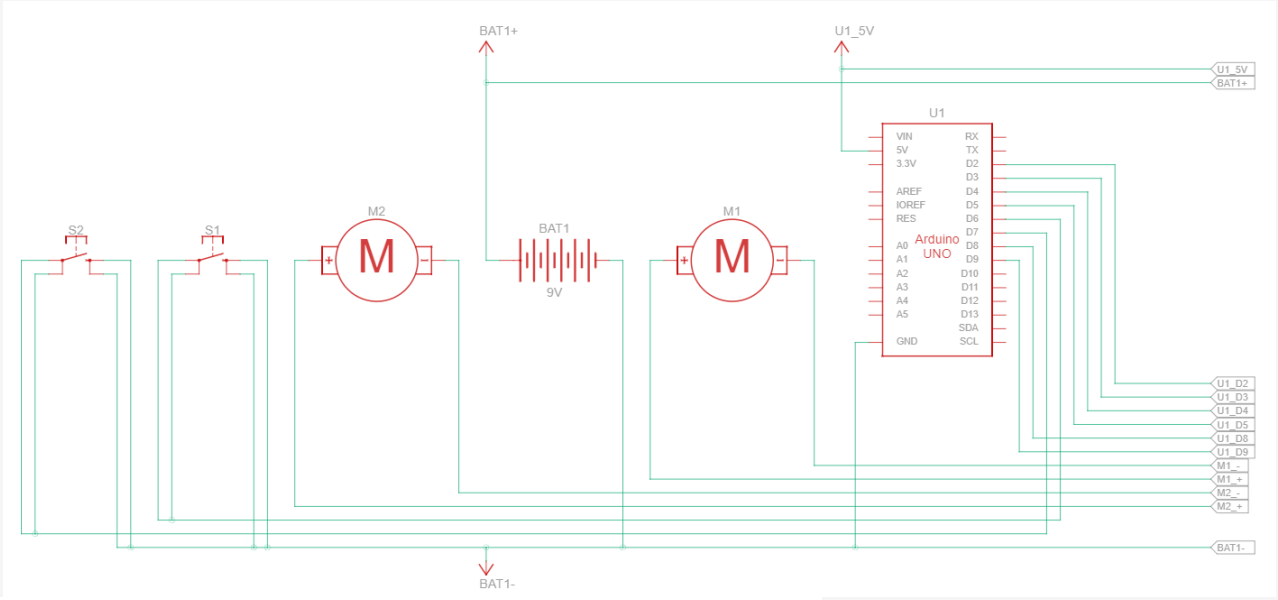
54

Name	Quantity	Component
U1 U3	2	Arduino Uno R3
U2 U4 U5	3	H-bridge Motor Driver
M1 M2 M3 M4 M5	5	DC Motor
BAT1	1	9V Battery
S3 S4 S1 S2 S5 S6 S7 S8 S9 S10	10	Pushbutton

Slika 54 – Lista komponenti za projekat Robotska ruka

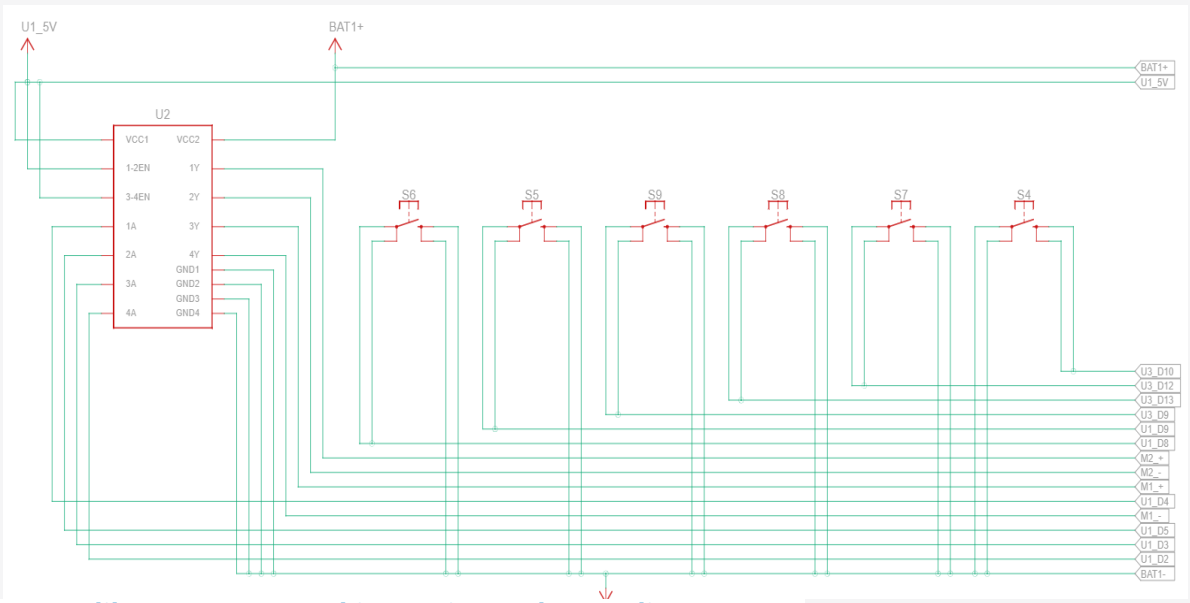
Automatski generirane elektroničke sheme prikazane su na **Slici 55, 56 i 57.**

55



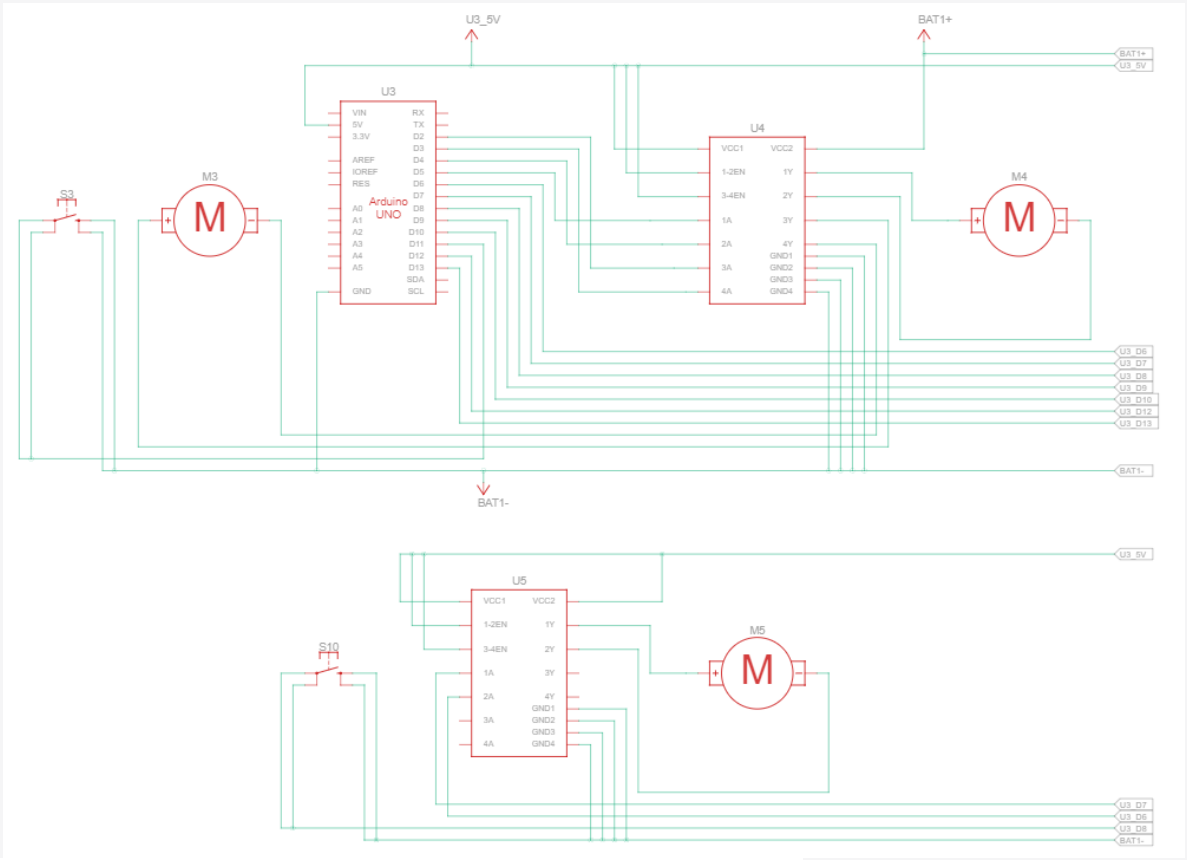
Slika 55 – Automatski generirana shema (dio 1)

56



Slika 56 – Automatski generirana shema (dio 2)

57



Slika 57 – Automatski generirana shema (dio 3)

Slijede upravljački kodovi za dva Arduino UNO mikrokontrolera korištena u sklopu:

UPRAVLJAČKI KOD ZA PRVI ARDUINO UNO MIKROKONTROLER

```
int m1a = 2;
int m1b = 3;
int m2a = 4;
int m2b = 5;
int T1a = 6;
int T1b = 7;
int T2a = 8;
int T2b = 9;
int citanje1, citanje2, citanje3, citanje4;

void setup()
{

  pinMode(m1a, OUTPUT);
  pinMode(m1b, OUTPUT);
  pinMode(m2a, OUTPUT);
  pinMode(m2b, OUTPUT);
  pinMode(T1a, INPUT_PULLUP);
  pinMode(T1b, INPUT_PULLUP);
  pinMode(T2a, INPUT_PULLUP);
  pinMode(T2b, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
  citanje1 = digitalRead(T1a);
  citanje2 = digitalRead(T1b);
  citanje3 = digitalRead(T2a);
  citanje4 = digitalRead(T2b);
  Serial.println(citanje1);
  Serial.println(citanje2);
  Serial.println(citanje3);
  Serial.println(citanje4);
  delay(500);

  if(citanje1 == 0)
  {
    digitalWrite(m1a, HIGH);
    digitalWrite(m1b, LOW);
    delay(300);
  }
   else if (citanje2 == 0)
  {
    digitalWrite(m1a, LOW);
    digitalWrite(m1b, HIGH);
    delay(300);

  }
  else
  {
    digitalWrite(m1a, LOW);
    digitalWrite(m1b, LOW);
  }

   if(citanje3 == 0)
```

```
  {
    digitalWrite(m2a, HIGH);
    digitalWrite(m2b, LOW);
    delay(300);
  }
   else if (citanje4 == 0)
  {
    digitalWrite(m2a, LOW);
    digitalWrite(m2b, HIGH);
    delay(300);
  }
  else
  {
    digitalWrite(m2a, LOW);
    digitalWrite(m2b, LOW);
  }
}
```

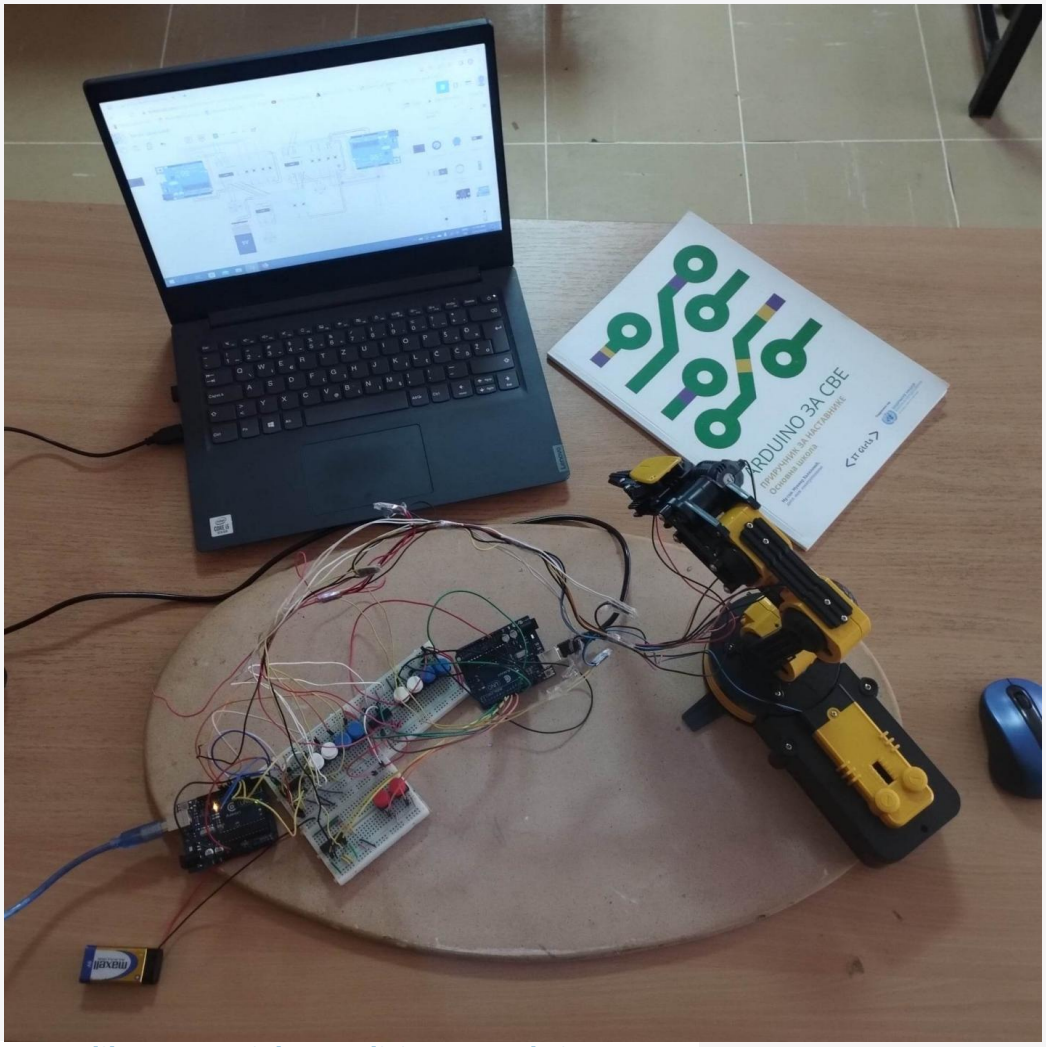
UPRAVLJAČKI KOD ZA DRUGI ARDUINO UNO MIKROKONTROLER

```
int m3a = 2;
int m3b = 3;
int m4a = 4;
int m4b = 5;
int m5a = 6;
int m5b = 7;
int T3a = 10;
int T3b = 11;
int T4a = 12;
int T4b = 13;
int T5a = 8;
int T5b = 9;
int citanje5, citanje6, citanje7, citanje8, citanje9, citanje10;

void setup()
{

  pinMode(m3a, OUTPUT);
  pinMode(m3b, OUTPUT);
  pinMode(m4a, OUTPUT);
  pinMode(m4b, OUTPUT);
  pinMode(m5a, OUTPUT);
  pinMode(m5b, OUTPUT);
  pinMode(T3a, INPUT_PULLUP);
  pinMode(T3b, INPUT_PULLUP);
  pinMode(T4a, INPUT_PULLUP);
  pinMode(T4b, INPUT_PULLUP);
  pinMode(T5a, INPUT_PULLUP);
  pinMode(T5b, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
  citanje5 = digitalRead(T3a);
  citanje6 = digitalRead(T3b);
  citanje7 = digitalRead(T4a);
  citanje8 = digitalRead(T4b);
  citanje9 = digitalRead(T5a);
  citanje10 = digitalRead(T5b);
  Serial.println(citanje5);
  Serial.println(citanje6);
  Serial.println(citanje7);
  Serial.println(citanje8);
  Serial.println(citanje9);
  Serial.println(citanje10);
  delay(500);

  if(citanje5 == 0)
  {
    digitalWrite(m3a, HIGH);
    digitalWrite(m3b, LOW);
    delay(300);
  }

   else if (citanje6 == 0)
```

```
  {
    digitalWrite(m3a, LOW);
    digitalWrite(m3b, HIGH);
    delay(300);
  }
  else
  {
    digitalWrite(m3a, LOW);
    digitalWrite(m3b, LOW);
  }

   if(citanje7 == 0)
  {
    digitalWrite(m4a, HIGH);
    digitalWrite(m4b, LOW);
    delay(300);
  }

   else if (citanje8 == 0)
  {
    digitalWrite(m4a, LOW);
    digitalWrite(m4b, HIGH);
    delay(300);
  }

  else
  {
    digitalWrite(m4a, LOW);
    digitalWrite(m4b, LOW);
  }
  if(citanje9 == 0)
  {
    digitalWrite(m5a, HIGH);
    digitalWrite(m5b, LOW);
    delay(300);
  }

   else if (citanje10 == 0)
  {
    digitalWrite(m5a, LOW);
    digitalWrite(m5b, HIGH);
    delay(300);
  }
  else
  {
    digitalWrite(m5a, LOW);
    digitalWrite(m5b, LOW);
  }
}
```

Projekat realiziran u praksi prikazan je na **Slici 58.**

58



Slika 58 – Projekat realiziran u praksi

ZAKLJUČAK

Osnove Tinkercada publikacija je koja nastavnike i nastavnice, profesore i profesorice, učenike i učenice treba da uvede u osnove ove kvalitetne razvojne simulacione platforme. Tinkercad se nastojalo prikazati iz perspektive edukatora, korisnika simulaciono-razvojne platforme, kao predavački alat, kao kvalitetan alat za online nastavu i online učenje i kao moćna platforma za simulacioni razvoj složenih STE(A)M projekata. Pri tome se pratila STE(A)M obrazovna paradigma zasnovana na ishodima učenja. Nastojale su se istaći dobrobiti za učenice i učenike koje proizilaze realizacijom Arduino STE(A)M projekata i postojanjem IT GIRLS klubova za stvaranje stvarnog dodira sa informacionom tehnologijom i savremenom tehnikom, što rezultira usvajanjem učeničkih kompetencija, znanja i vještina aktuelnih na savremenom tržištu rada.

LITERATURA

ISTE. (n.d.). TINKERCAD. Preuzeto 3. 12. 2022. iz **https://www.iste.org/standards/seal-of-alignment/tinkercad**

TINKERCAD, A. (n.d.). AUTODESK TINKERCAD WEB PLATFORMA. Preuzeto 8. 12. 2022. iz **https://www.tinkercad.com**

APOSO. (2016). ZJNPP za tehniku i IT. U o. i. Agencija za predškolsko, ZJNPP za tehniku i informacione tehnologije definisana na ishodima učenja. Mostar

Grupa autora. (2022). NPP ZA INFORMATIKU - BRČKO DISTRIKT. U G. AUTORA. Brčko.

APOSO. (2017). ZJNPP za fiziku. U ZJNPP za fiziku definisana na ishodima učenja. Mostar: APOSO.

Samuel – Tinker. (2016). Robotic arm „edge“. Preuzeto 12. 12. 2022. iz **https://www.tinkercad.com/things/17VjmIXrG7D-robotic-arm-edge-3312016**