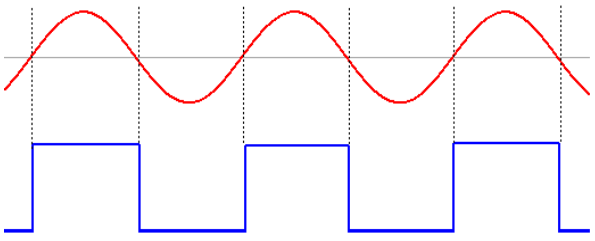
3. Joystick a RGB dioda

Obsah

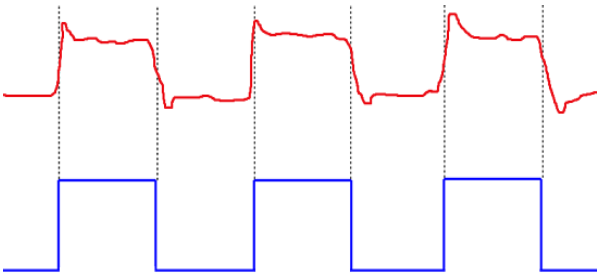
Analogový a digitální signál

Rozlišujeme dva druhy signálu, jakožto nosiče informace, *analogový* a *digitální*. Analogový signál charakterizujeme jako spojitou funkci v čase i napětí. Analogový signál tedy může nabývat libovolné hodnoty. Oproti tomu digitální signál spojitou funkcí není a proto může nabývat pouze omezeného množství hodnot (nejčastěji dvou).



Obrázek 1. Příklad analogového a digitálního signálu

Ač by měl dle definice digitální signál na obrázku níže nabývat pouze striktně jednu ze dvou hodnot (jak je zobrazeno v dolní části obrázku), v praxi se velikost signálu často drobně mění (jak je zobrazeno v horní části obrázku). Především změna signálu není zcela okamžitá, ale spíše pozvolná. Protože nás ovšem zajímá pouze jestli vstupní veličina je nebo není v daném stavu (např. jestli RGB LED svítí nebo ne), tyto drobné odchylky zanedbáváme. Tyto odchylky jsou navíc tak malé, že ani nebývají postřehnutelné okem.



Obrázek 2. Idealizovaný a reálný průběh digitálního signálu

Přípravky Arduino poskytují dva typy vstupů, analogové a digitální. Liší se rozdílným zpracováním příchozího napětí – digitální vstup pouze registruje, zda napětí na pinu je (5V) nebo není (0V), zatímco analogový měří velikost napětí a to poté převádí do digitální podoby na číslo. S digitálními vstupy (a výstupy) jsme se již setkali v předchozích cvičeních.

Na Arduinu nalezneme hned několik analogových vstupů, označovaných `A0` až `A5`. Na tyto piny můžeme přivést např. výstupy analogových senzorů. Analogový signál, které tyto senzory snímají je pro snadnější manipulaci nutné převést na signál digitální pomocí *analogově digitálního převodníku* (zkráceně A/D převodníku). A/D převodníky převádí vstupní analogovou veličinu na digitální signál pomocí vzorkování (proces odběru vstupního signálu v definovaných okamžicích). Při převodu analogového signálu na digitální vždy dochází ke ztrátě informace.

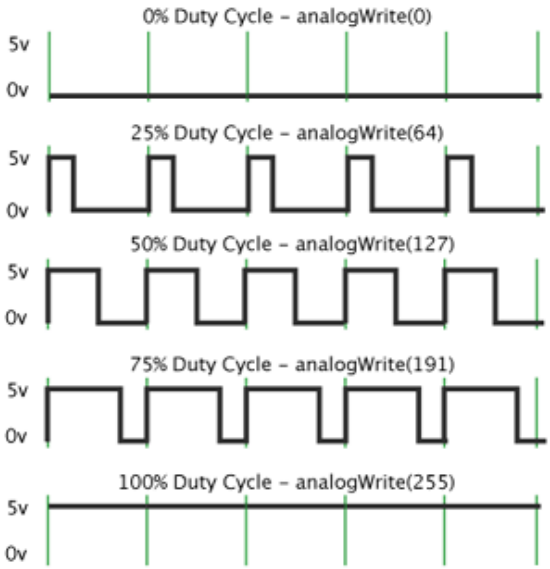
Některé analogové piny na Arduinu lze použít i jako digitální, stačí použít makro `Ax` jako číslo pinu (např. `digitalWrite(A3, value)`).

Ekvivalentně k A/D převodníkům existují i *digitálně analogové převodníky* (zkráceně D/A převodníky), které převádí vstupní digitální veličinu na analogový signál. Arduino Uno neobsahuje D/A převodník, k aproximaci analogového signálu používá techniku zvanou *pulsně šířková modulace* (PWM).

PWM lze v některých situacích považovat za jednoduchý D/A převodník. Simuluje analogový signál pomocí změny „šířky“ signálu, tzv. střídy (tj. poměru délky signálu v logické jedničce ku délce signálu v logické nule během jedné periody). Střídu (angl. Duty Cycle) vyjadřujeme poměrem nebo procentuálně. Pokud je hodnota střídy 100 %, nachází se signál neustále v logické jedničce, pokud je hodnota střídy 0 %, pak se signál nachází v logické nule.

Na některé digitální piny přípravku Arduino lze připojit PWM. Které to jsou zjistíte z dokumentace, nebo takové piny bývají označené vlnkou (~).

Na obrázku níže vidíme rozdíl v průběhu signálu při zápisu různé číselné hodnoty na pin pomocí vestavěné funkce `analogWrite()`. Takto lze snadno ovládat například jas LED.



Obrázek 3. PWM, pulsně šířková modulace

Analogové moduly použité na dnešním cvičení

Potenciometr

Potenciometr je odporová součástka, která realizuje proměnný napěťový dělič. Má 3 vývody a pozice prostředního vývodu určuje dělící poměr. Vztah mezi pozicí (rotací, posunem) a dělícím poměrem může být lineární, logaritmický, exponenciální...

Napětí na prostředním vývodu potenciometru budeme měřit funkcí `analogRead()`. Rozsah měřených hodnot je 0–1023 (10 bitů), které odpovídají napětí 0–5 V.

Joystick

Joystick je v podstatě potenciometr, nebo dvojice potenciometrů, který se samovolně vrací do jedné klidové polohy.

LED

Jas LED budeme ovládat pomocí PWM, funkcí `analogWrite()`. Rozsah hodnot jasu je 0–255.

Pokud máte RGB LED, v této úloze ji využijete, pokud ji nemáte, použijte 3 samostatné LED.

Bzučák

Obrázek 4. Bzučák

Bzučák (piezoměnič) budeme ovládat pomocí funkcí `tone()` a `noTone()`. Tyto funkce využívají jednotky PWM, kde místo střídy mění frekvenci.

Zapojení

POUŽITÉ SOUČÁSTKY:

- RGB LED
  - nebo 3 LED, 4 pro bonusovou úlohu
- potenciometr
- joystick
  - nebo 2 potenciometry
- pasivní bzučák
- 1 tlačítko (lze použít tlačítko joysticku)
  - 4 tlačítka pro bonusovou úlohu

Potenciometr

Vnější vývody potenciometru zapojíme na Ucc a GND. Prostřední vývod (jezdce) připojíme na analogový vstup. Ještě jednou si zkontrolujte, že jednotlivé vývody sedí (že jezdec je prostřední vývod).

Joystick

Modul joysticku by měl mít popsané vývody. Dva vývody na napájení (Ucc a GND), dva na analogový vstup a (volitelně) jeden na digitální vstup.

LED

LED zapojíme obvyklým způsobem.

Pokud máte RGB LED modul, dejte pozor, jestli modul obsahuje předřadné rezistory, nebo ne. U RGB LED si všímejte, zda má společnou anodu, nebo společnou katodu, změní to způsob zapojení. Pro každou barvu použijte vlastní předřadný rezistor.

Je potřeba zvolit piny schopné pracovat s PWM, protože budeme měnit jas LED.

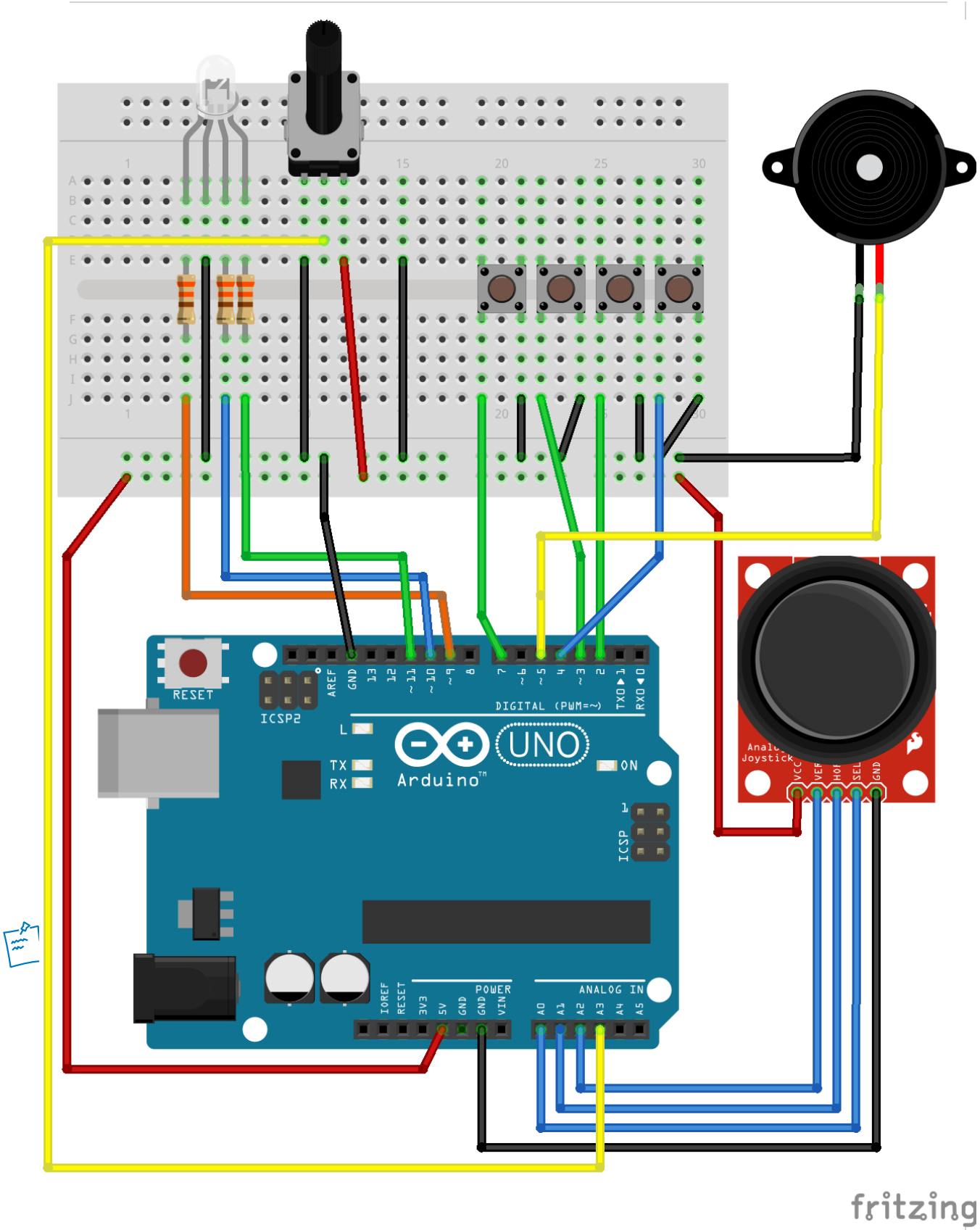
Tlačítka

Tlačítka zapojíme obvyklým způsobem.

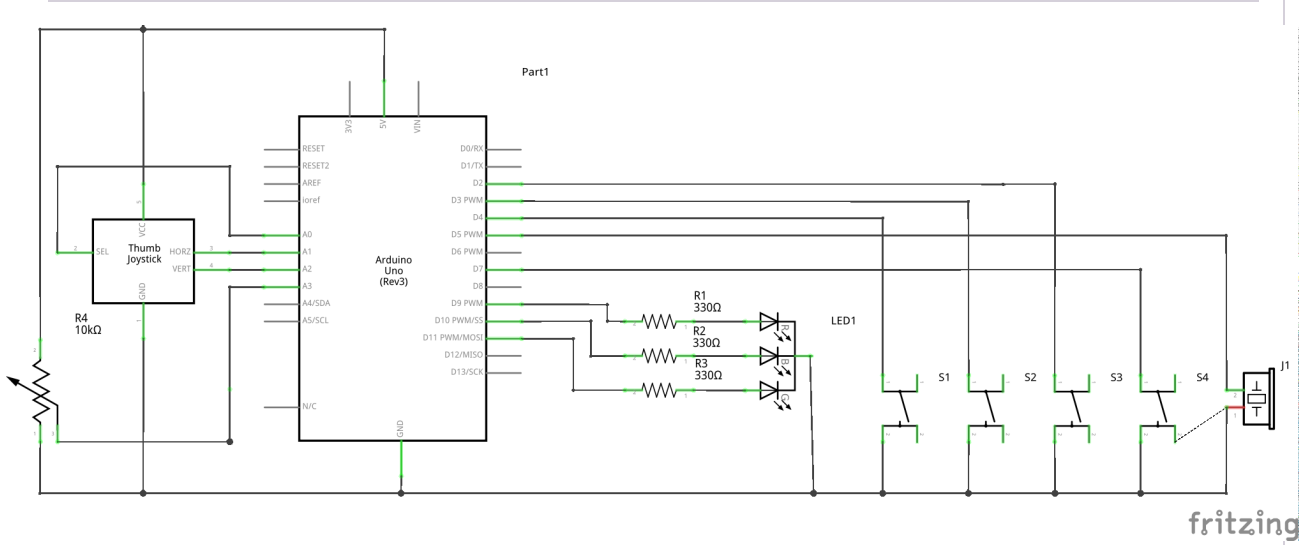
Bzučák

Bzučák zapojíme mezi pin schopný pracovat s PWM a zem. Ještě jednou zkontrolujte, že máte bzučák schopný pracovat s napětím 5 V.

▼ Příklad zapojení



Obrázek 5. Zapojení součástek na breadboardu



Obrázek 6. Schéma zapojení

Úlohy

V 1. a 2. úloze si vyzkoušíte práci s joystickem, potenciometrem a piezoměničem. V první části budete sledovat, jak se data získaná z joysticku v čase mění (př. jak moc vlevo se joystick posunul). Budete tedy získaná data vnímat „analogově" (analogově je v uvozovkách, protože stejně je analogová hodnota převedena s určitou přesností na diskrétní množinu hodnot rozsahu příslušné periferie (př. u RGB na 0 až 255)). Ve druhé části budete naopak budete pracovat s přečtenými hodnotami jako s digitálními, tedy bude vás zajímat pouze jestli je nebo není joystick v určité poloze (tj. vlevo, vpravo, ...).

1. Joystick - analogový přístup (3 b)

Detekujte pohyb joysticku a podle jeho polohy (nahoře, dole, vpravo, vlevo) rozsviťte RGB LED (pro každý směr zvolte jinou barvu). Jas RGB LED regulujte podle toho, jak moc je joystick vychýlen od základní pozice ve středu. Pokud tedy bude joystick v krajní poloze, bude LED svítit nejjasněji, zatímco ve středu bude vypnutá.

Nejdříve si nastudujte použití funkce `analogRead()` v dokumentaci. Vypište si na sériový monitor hodnoty souřadnic x a y.

Dále pomocí funkce `analogWrite()` zapněte příslušné LED. Jako jeden z parametrů této funkce zvolte aktuální hodnotu souřadnice joysticku (x pro horizontální pohyb, y pro vertikální pohyb). Pozor! Jelikož se rozsah souřadnice joysticku a jasu RGB LED liší, je třeba rozsah souřadnice „přemapovat“ na hodnoty od 0 do 255. K tomu můžete využít funkce `map()`, viz poznámka. Vypisujte na sériový monitor jak původní hodnotu souřadnice joysticku, tak i přemapovanou hodnotu, použitou pro ovládání LED.

Shrnutí:

- každému směru přiřaďte jinou barvu
- ve směru šikmo by se měly míchat barvy příslušných směrů
- při plynulém pohybu joystickem by se neměly barvy měnit skokově (ani při „kroužení“ joystickem v maximálním vychýlení)
- jas by se měl zvětšovat s vychýlením joystiku

Pro „přemapování“ rozsahu hodnot jedné periferie na druhý (př. potenciometr a RGB LED) můžete využít funkce `map()`: https://www.arduino.cc/en/Reference/Map

2. Joystick - digitální přístup (2 b)

Ve druhé části úlohy opět podle polohy joysticku rozsviťte RGB LED (pro každý směr zvolte jinou barvu). Nyní ovšem regulujte intenzitu jasu RGB LED pomocí potenciometru. Pokud je tedy joystick nakloněn např. vlevo, rozsviťte příslušnou LED o intenzitě jasu, která odpovídá aktuální poloze potenciometru.

Pokud se joystick nachází uprostřed ponechte LED vypnuté. Při stisknutí tlačítka zapněte piezo bzučák a výšku tónu ovládejte přes potenciometr. Využijte funkcí `tone()` a `noTone()`. Při puštění tlačítka bzučák vypněte. Můžete využít tlačítka joysticku, pokud ho máte připojené.

Pokud náhodou používáte desku Arduino Esplora: V knihovně Esplora je chyba, po použití bzučáku nefunguje červená LED. V takovém případě se podívejte na verzi stránky tohoto cvičení ze semestru B192. Tam najdete návod na řešení této chyby.

Shrnutí:

- barvy *není* nutné míchat
- jas by měl být ovládán pouze pozicí potenciometru
- tón by měl znít pouze při stisku tlačítka

Můžete použít tuto základní strukturu programu:

```
#define UP_TRESHOLD          // zde vhodně nastavte hranici, která určí, že je joystick v pozici nahoře
#define DOWN_TRESHOLD        // zde vhodně nastavte hranici, která určí, že je joystick v pozici dole
#define RIGHT_TRESHOLD       // zde vhodně nastavte hranici, která určí, že je joystick v pozici vpravo
#define LEFT_TRESHOLD        // zde vhodně nastavte hranici, která určí, že je joystick v pozici vlevo

#define CENTER 0
#define LEFT 1
#define UP 2
#define RIGHT 3
#define DOWN 4
#define PUSH 5


int readJoystickKey(){
    if (button == LOW) {
      return PUSH;
    }

    if (yPos > UP_TRESHOLD) {     // joystick je nahoře
      return UP;
```

```
    }

    if (yPos < DOWN_TRESHOLD) {   // joystick je dole
      return DOWN;
    }

    if (xPos > RIGHT_TRESHOLD) {  // joystick je vpravo
      return RIGHT;
    }

    if (xPos < LEFT_TRESHOLD) {   // joystick je vlevo
      return LEFT;
    }

    return CENTER;
}


void setup()
{

}

void loop()
{
      switch (pos_joystick)
      {
        case LEFT:
        case RIGHT:
        case UP:
        case DOWN:
        case CENTER:
        case PUSH:
      }
}
```

3. úloha - hra Bum do krtka (bonus: 2 b)

Vytvořte vlastní verzi hry Bum do krtka na Arduinu. Princip hry je jednoduchý - na (RGB) LED budou v krátkých intervalech problikávat 4 barvy (každá je přiřazena jednomu tlačítku) a vaším cílem je stihnout stisknout správné tlačítko než se rozsvítí další barva. Vítězí ten, který zvládne správně stisknout více než polovinu tlačítek ze zadaného intervalu. Hra by měla umět poznat podvádění (např. stisknu všechna tlačítka).

Zkraťte hru na 5 pokusů, ať to netrvá tak dlouho kontrolovat.

Použijte tuto základní strukturu programu:

```
void loop()
{
  int num = (random(60000) % 4);              // je třeba, aby barvy blikaly náhodně

  switch(num)
  {
    case 0:
      zde rozsviťte červenou RGB LED
      push_button(SWITCH_1, "red");       // pokud se rozsvítí červená LED, musíte zmáčknout SWITCH 1
      break;

    case 1:
      zde rozsviťte zelenou RGB LED
      push_button(SWITCH_2, "green");     // pokud se rozsvítí zelená LED, musíte zmáčknout SWITCH 2
      break;

     case 2:
      zde rozsviťte modrou RGB LED
      push_button(SWITCH_3, "blue");       // pokud se rozsvítí modrá LED, musíte zmáčknout SWITCH 3
      break;

     case 3:
      zde rozsviťte žlutou RGB LED
      push_button(SWITCH_4, "yellow");     // pokud se rozsvítí žlutá LED, musíte zmáčknout SWITCH 4
  }

   if( na RGB LED již probliklo 30 barev )
   {

    Zde vypište počet správně stisknutých tlačítek

    if( byla správně zmáčknuta více než polovina tlačítek )
    {
      Zde vypište na monitor * Winner! *
    }
    else
    {
      Zde vypište na monitor * Loser! *
    }
  }
 }

void push_button( číslo tlačítka, název barvy )
{
    for(int i=0; i < 4000; i++)                    // velikost i určuje dobu rozsvícení LED
    {
      if( je stisknuté správné tlačítko )
      {
        ...
        zde vypište na monitor jméno rozsvícené barvy
```

```
            ...
        }
    }
}
```