Úvod do grafického RobotC

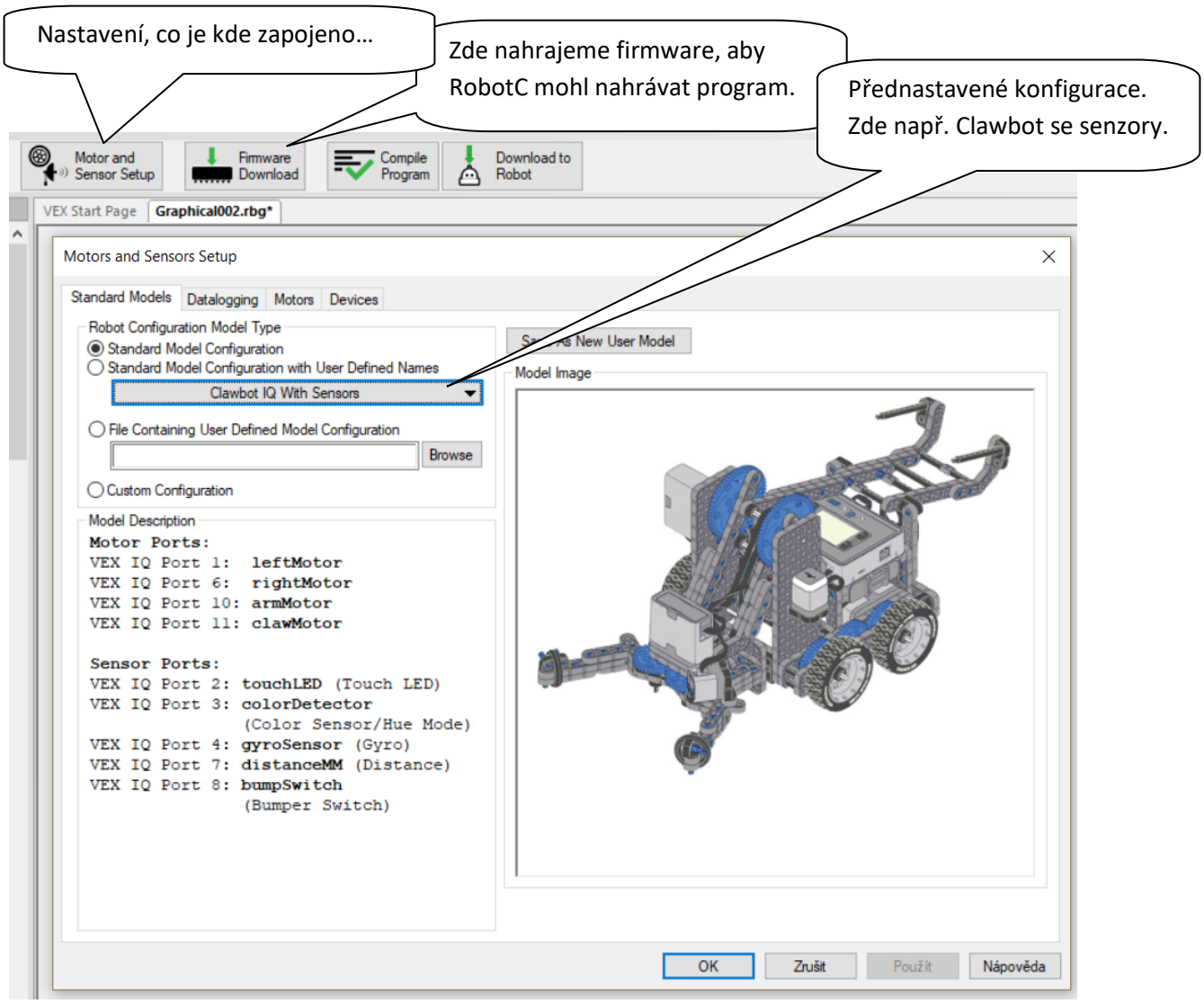
Nastavení

Po spuštění programu si vytvoříme **nový soubor** (New File).

Pak **zkontrolujeme**, zda je potřeba **nahrát do robota firmware**[1)]. Výchozí nastavení totiž ukazuje na displeji položky Driver Control, Demos a sloty 1 až 4 pro programy. Pro použití programu RobotC musíme nahrát jeho firmware, a to **kliknutím na tlačítko Firmware Download**. Na displeji se potom zobrazí kromě Driver Control a Demos ještě položky Auto Pgms a TeleOp Pgms (programy pro autonomní jízdu a programy pro jízdu s pomocí ovladače).

Dále můžeme přistoupit ke **konfiguraci robota** (tlačítko **Motor and Sensor Setup**).

Pokud stavíme *Clawbota* podle návodu přiloženého ke stavebnici, potom stačí použít přednastavené hodnoty (Clawbot IQ nebo Clawbot IQ With Sensors):



---

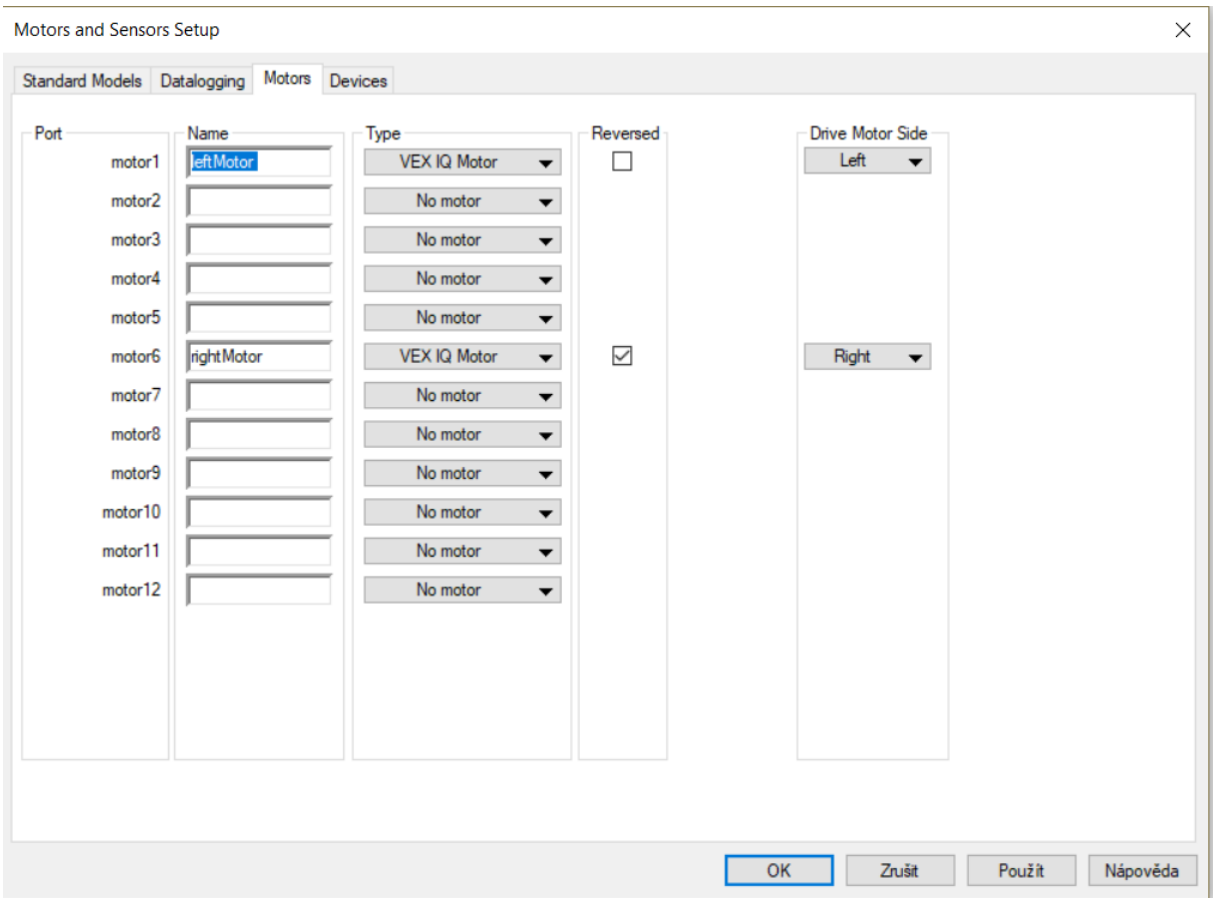
1) Někdy se stane, že mozek a senzory nebo motory nemají uvnitř stejný firmware, tuto situaci robot hlásí na displeji. Toto řešíme pomocí **VEXOS Utility**, která se nainstaluje současně s instalací RobotC, případně je na webu vexrobotics.com. Návod k použití na videu: https://youtu.be/4Ti89ErD3VI .

Pokud si zatím postavíme jen podvozek podle návodu ze stavebnice, nebo podobný, jako je na obrázku vpravo, stačí z přednastavených konfigurací zvolit *Standard Drive Base*, ale ukážeme si i, co kde nastavujeme, pokud v sekci Motors volíme motory ručně:

V prvním sloupci *Name* si motory pojmenováváme. V ukázce je výchozí anglický název *leftMotor*, což je v programování dobrý přístup. Zvolit můžeme i jiné jméno, např. *levyMotor* nebo *levýMotor* je pořád lepší než *levý motor*. Hlavní je, abychom se pak ve svém programu vyznali.

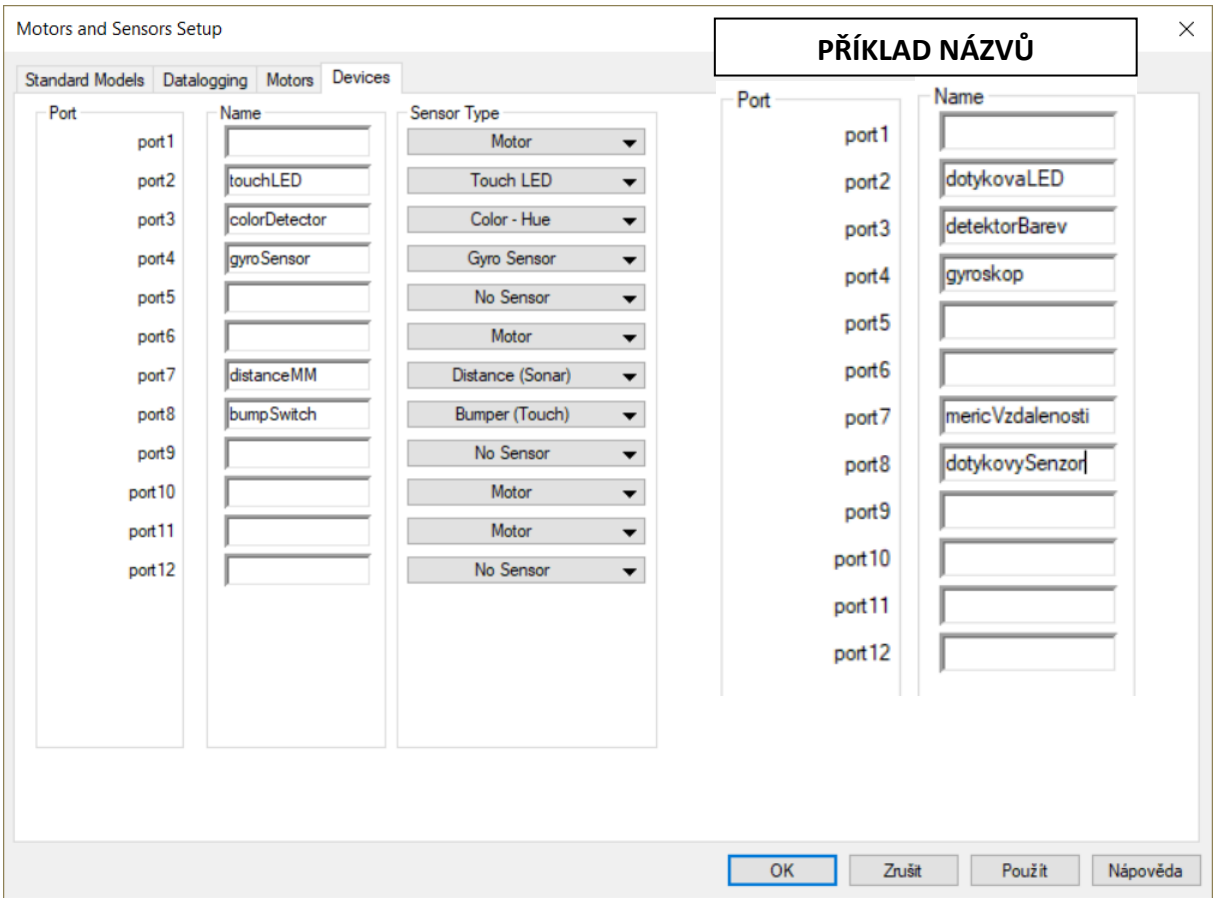
Ve druhém sloupci *Type* u všech motorů vybereme *VEX IQ Motor*.

Ve třetím sloupci u jednoho motoru zatrhneme *reversed*, protože je vůči levému otočený o 180° a točil by se dozadu. *Drive Motor Side* se používá pouze u podvozku a vyplňujeme zde, zda jde o levý nebo pravý motor. Pokud by šlo o motor ovládající rameno nebo klepeta, necháváme zde *None*.



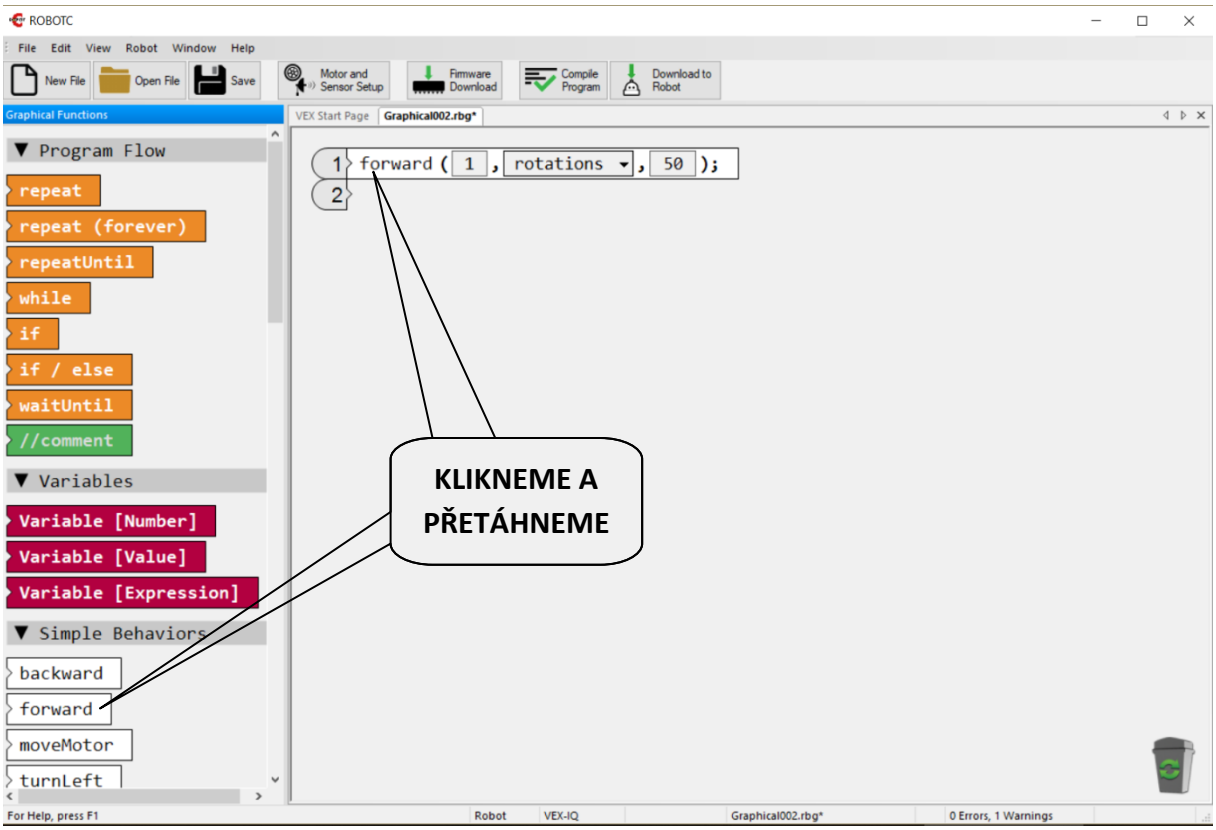
Nastavení záložky Devices se nás nyní netýká, pokud si stavíme jen jezdící platformu, nebo Clawbota bez senzorů apod., ale jakmile nějaké senzory zapojíme, tak je zde nazveme a nastavíme.

Podobně jako u motorů platí, že názvy si volíme sami, a ve sloupci *Sensor Type* vybíráme, o jaký senzor se jedná. Kde svítí *Motor*, tak na tuto položku samozřejmě nesaháme. Příklad nastavení pro *Clawbota se senzory* je na následujícím obrázku.



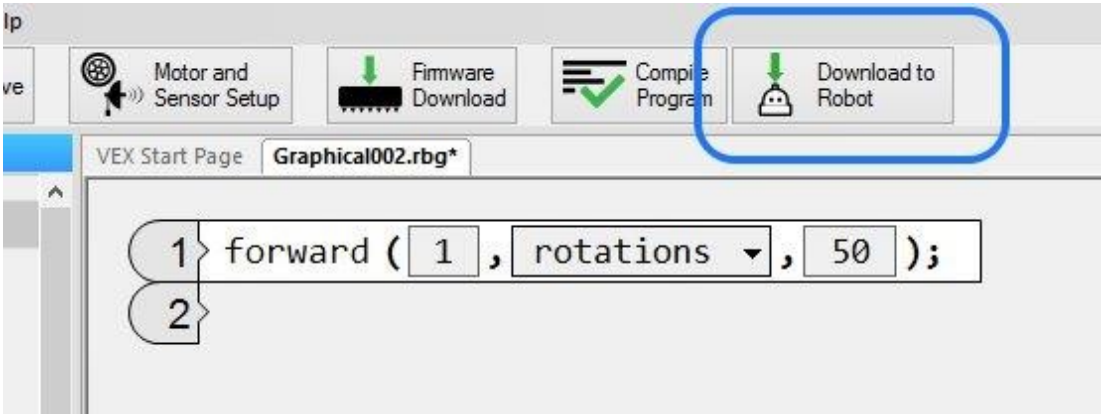
Programujeme MiniVEXe

Příkazy vybíráme a přetahujeme myší z nabídky vlevo do očíslovaných řádků vpravo.



Přetažením příkazu jsme vlastně vytvořili náš první program, nyní jej **nahrajeme do robota**.

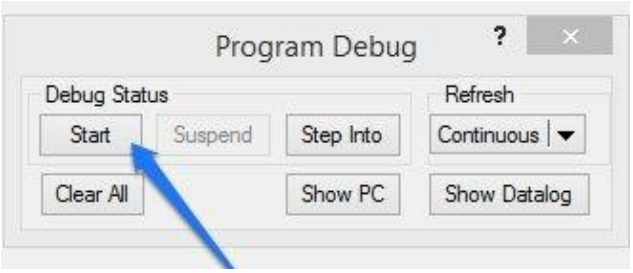
Připojíme mozek robota do USB počítače, zapneme jej a stiskneme **Download to Robot**[2]. Počítač nás v tuto chvíli ještě poprosí o **uložení našeho programu** (to se zkrátka dělá vždy, než může dojít ke kompilaci a nahrání programu).



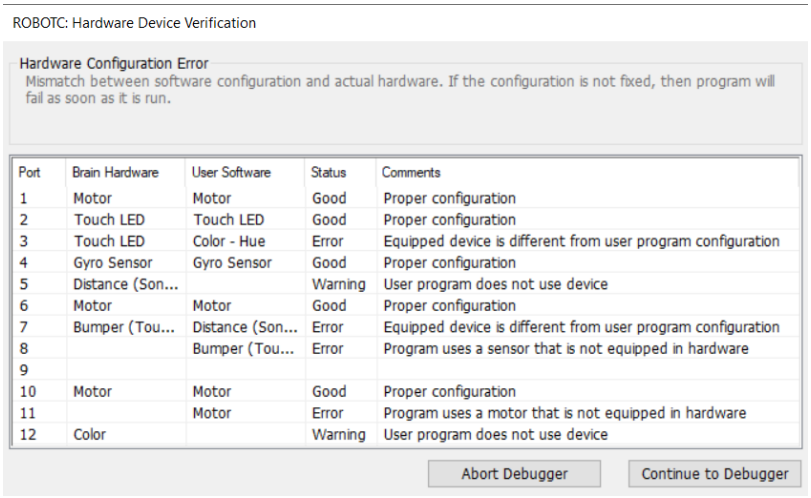
Po úspěšném nahrání se nám objeví buď okno s chybami nastavení robota, nebo se objeví malé **okénko k ovládání programu**. Zde můžeme program spustit tlačítkem ***Start***, pokud je ještě robot připojen kabelem a můžeme zde i program hned zastavit stiskem ***Stop*** na stejném tlačítku.

Program samozřejmě **můžeme spustit i na mozku robota** klávesou Enter  .

Pozn. při této operaci může dojít k hlášení, že **nahrání se nezdařilo**, nebo že selhalo propojení s robotem. Nejčastěji stačí stisknout nahrávání znovu. Může se také stát, že uživatel nezastrčí USB konektor do robota zcela. Popř. je třeba nejprve provést Firmware Download (tlač. vedle).



Pozn. V případě **chybových hlášení** program může fungovat – stiskneme *Continue to Debugger* – to platí pro chybu čidel. Naopak v případě chyby motoru musíme toto nejdřív opravit v *Motor and Sensor Setup*. Chyba zapojení motorů bývá hlášena i na displeji robota.



Status

–> Good – vše v pořádku,

-> Warning – OK, nepoužili jsme senzor v programu

-> Error – na portu 3 je nastaven ColorSensor, ale v mozku je na portu 3 TouchLED.

-> Error – program má nastaven motor na portu 11, ale ten není zapojen

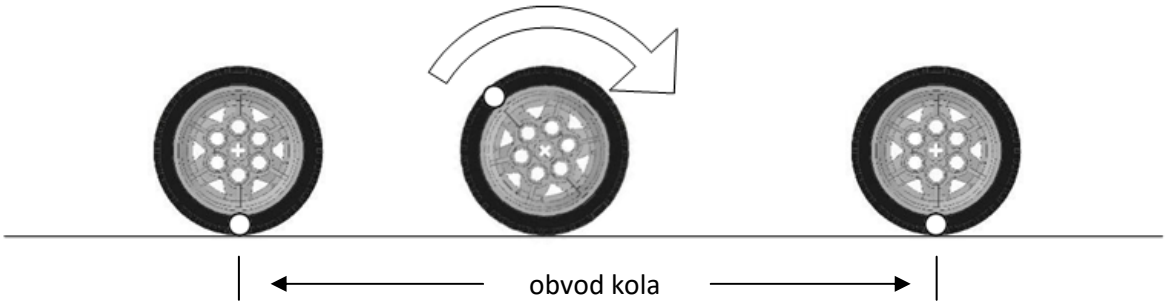
2) Tlačítko Download zároveň provede i kompilaci. Nemusíme tedy mačkat Compile Program.

Co umí náš program?

Popravdě, moc toho není. **Robot popojede o jednu otáčku kol, rychlostí 50**.

```
1 forward ( 1 , rotations , 50 );
2
```

Matematicky vzato jedna otáčka kol odpovídá nějaké vzdálenosti.
Měřením můžeme zjistit, že průměr kol VEXe je 63,7 mm a tedy podle vzorce pro obvod kruhu $o = 2 \cdot \pi \cdot r$ dostaneme, že robot ujel vzdálenost $2 \cdot 3{,}14 \cdot 31{,}85 =$ **200 mm**[3)].



Jak můžeme nastavit **rychlost**? Hodnota **nabývá 0 – 100**.

Kromě hodnoty **rotations** můžeme volit **degrees** (stupně otočení kol), **miliseconds**, **second, minutes**.

```
1 forward ( 1 , rotations , 50 );
2
```
degrees
rotations
milliseconds
seconds
minutes

Zatáčení robota

Určitě vás napadne použít příkaz turnLeft. Ovšem pozor, tímto způsobem se neotáčí celý robot, ale pouze kola. To jak moc se robot otočí, pak u tohoto příkazu řešíme spíše pokusem/omylem, protože to závisí na tom, jak velká kola máme a jak daleko od sebe jsou. Proč? Podívejte se na další straně (viz úkol 5).

```
1 turnLeft ( 1 , rotations , 50 );
2
```

3) Obvod kol je na nich přímo uveden. Existují kola 100, 150, 200 (základní) a 250 mm a 200mm omni.

Několik příkladů k řešení:

1) Kdy dojede robot nejdál? Při nastavení *2 rotations*, *2 degrees* nebo *2 seconds*?

2) Kolik otáček kol je potřeba k ujetí vzdálenosti *45 cm*?

3) Liší se vzdálenost ujetá při *1080 degrees* a *3 rotations*?

4) Dojede robot do výchozí pozice, pokud pojede *vpřed 5 rotations rychlostí 10* a pak *zpět 1800 degrees rychlostí 100*?

```
1 forward ( 5 , rotations , 10 );
2 backward ( 1800 , degrees , 100 );
3
```

5) Otočka, jízda a návrat do výchozí polohy: Otočte se s robotem čelem vzad (např. o dvě otáčky kol doleva, ale vaše hodnota může být jiná, musíte zkoušet).

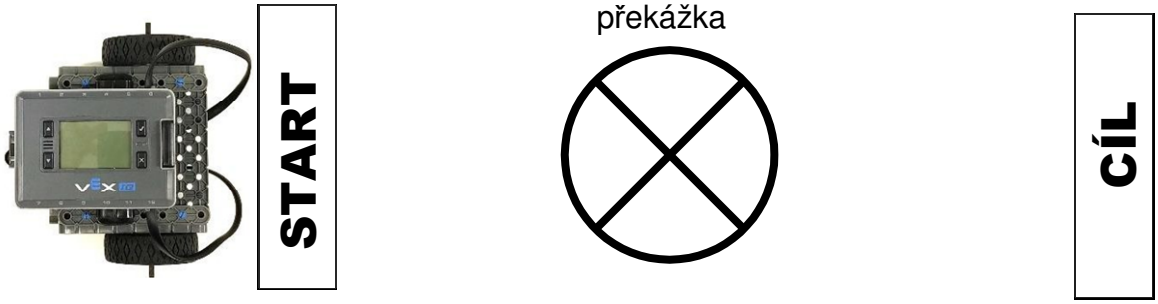
```
1 turnLeft ( 2 , rotations , 50 );
2
```

Nyní popojeďte 500 mm, zde se otočte a jeďte zpět do místa, odkud jste vyjeli.

```
1 forward ( 2.5 , rotations , 50 );
2 turnLeft ( 2 , rotations , 50 );
3 forward ( 2.5 , rotations , 50 );
4
```

Možná vás napadlo, proč jsme jednoduše nezacouvali zpět? Záhy však zjišťujeme, že tím, jak jsme se na začátku otočili, musíme nyní po popojetí udělat znovu otáčku čelem vzad a popojet dopředu, aby byl robot opravdu exaktně ve stejné výchozí pozici.

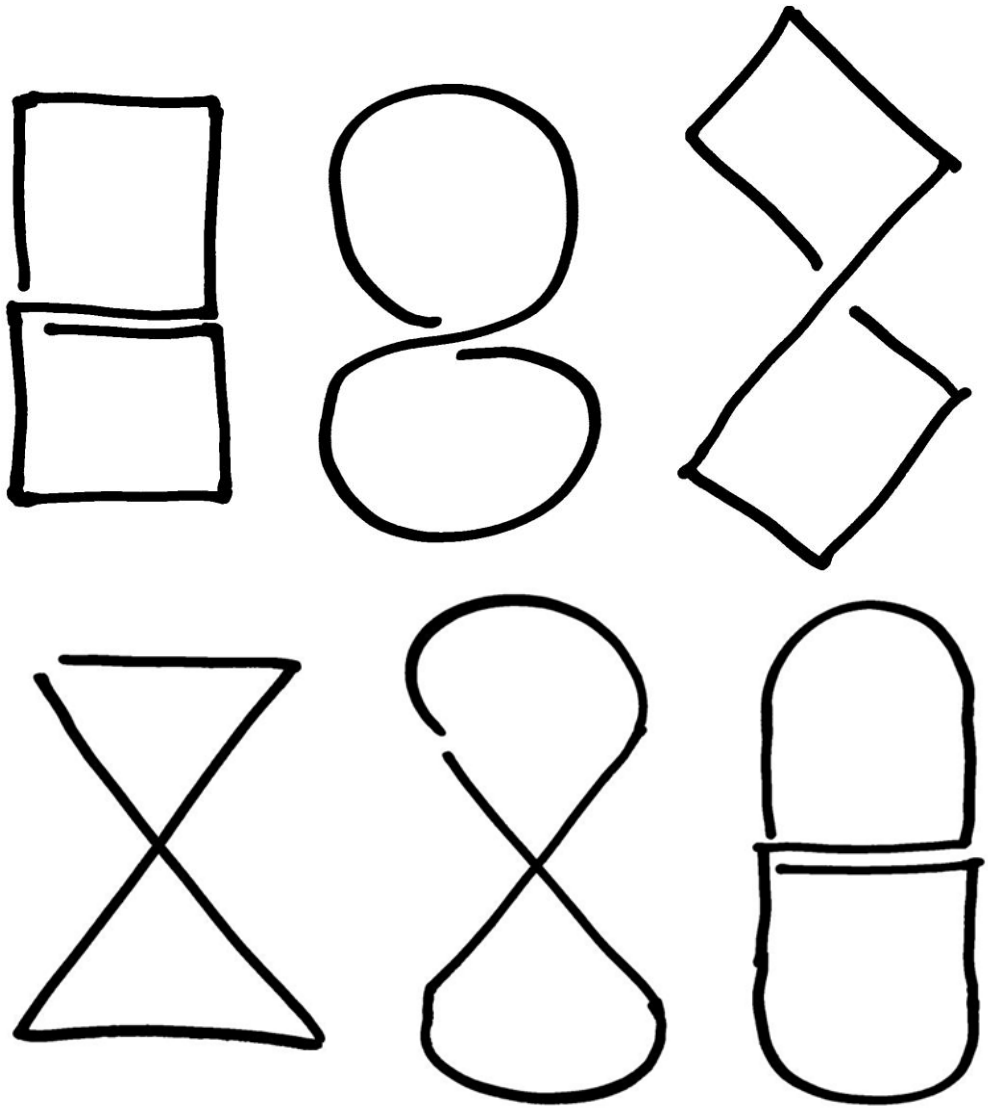
6) Nechte robota objet překážku



Možné řešení je takovéto:

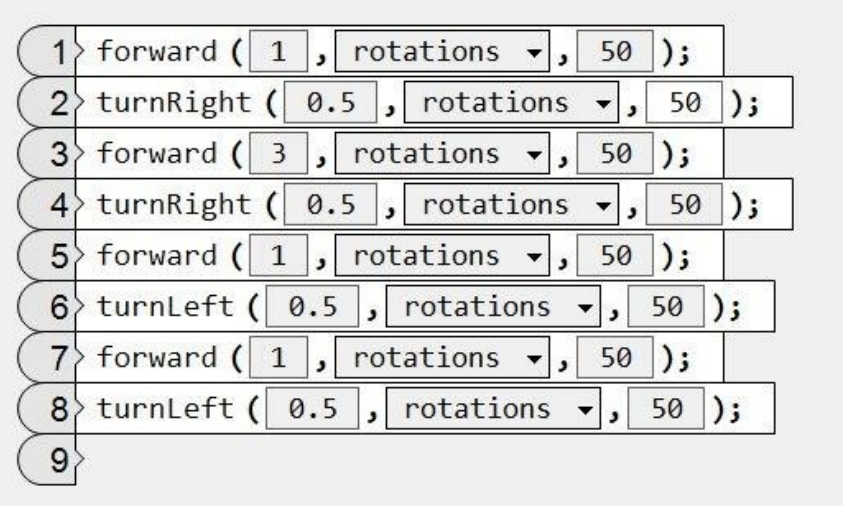
```
1 turnLeft ( 0.5 , rotations , 50 );
2 forward ( 1 , rotations , 50 );
3 turnRight ( .5 , rotations , 50 );
4 forward ( 3 , rotations , 50 );
5 turnRight ( .5 , rotations , 50 );
6 forward ( 1 , rotations , 50 );
7 turnLeft ( .5 , rotations , 50 );
8 forward ( 1 , rotations , 50 );
9
```

7) Nechte robota, aby nakreslil obrazec:



Řešení problémů v programu, hlášení, komentáře

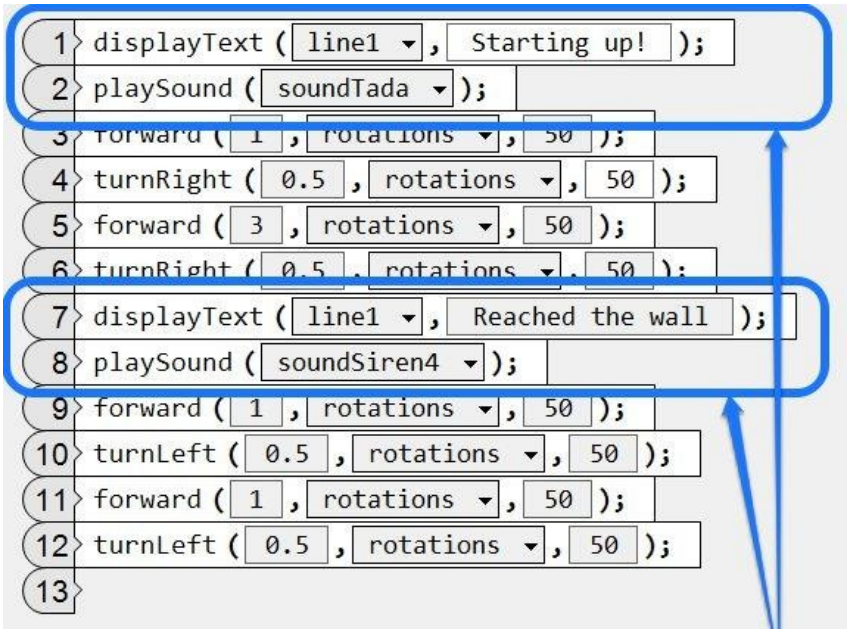
Jak si hrajeme, program nabobtnává a můžeme se v něm snadno ztratit. Je tedy vhodné si začít pomáhat s orientací v něm. Ukážeme si několik možností, jak na to.



Program obsahuje řadu příkazů, které mají často podobný vzhled, a můžeme se v něm ztratit.

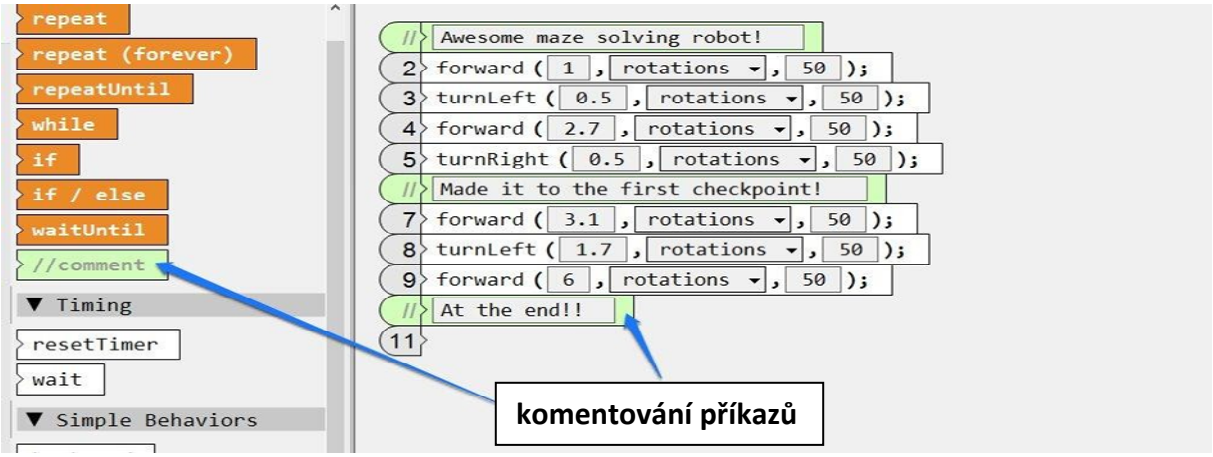
Hlášení na displeji a zvukový doprovod

Jedna ze základních možností jsou pípnutí podle toho, v které části kódu se zrovna robot nachází. Zároveň nás o tom může informovat na displeji.

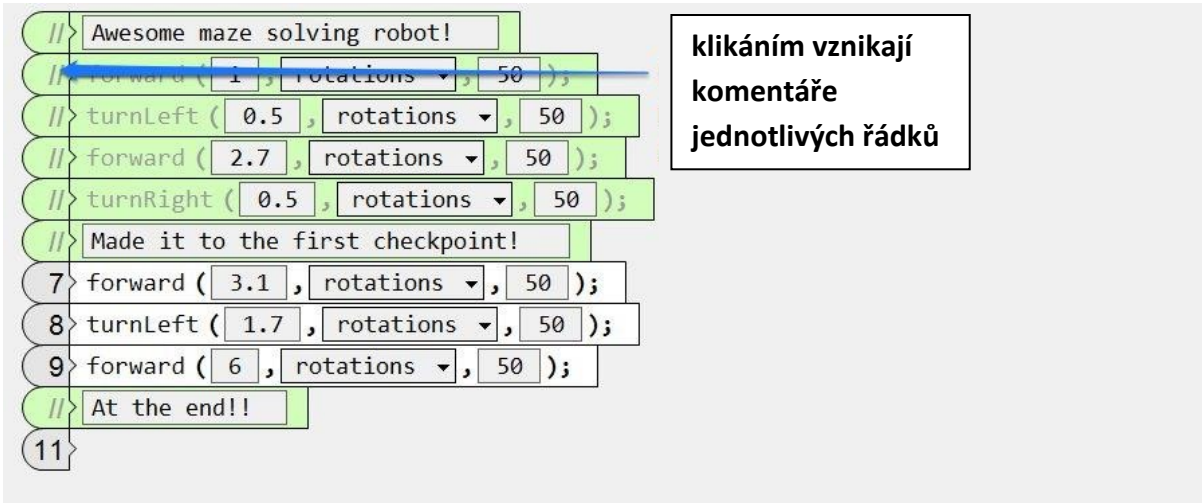


Přidávání komentářů do programu

Pokud programujete, víte, že zakomentovat můžeme každý řádek, nebo určitou část programu. Nabízí nám to i grafické RobotC.

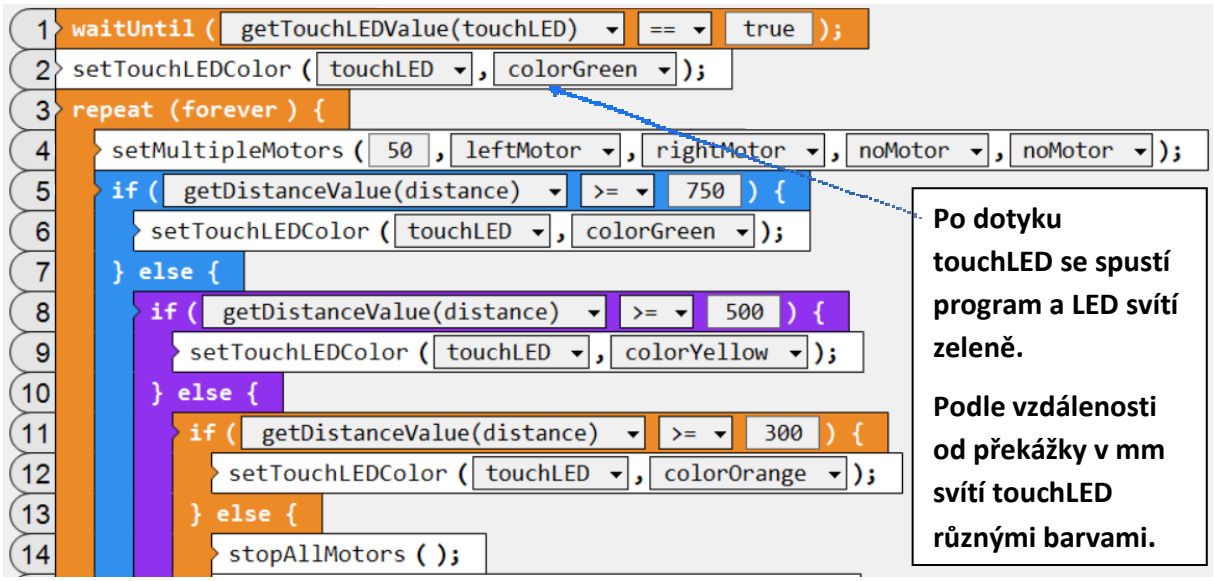


Klikáním na čísla řádků je zakomentujeme:



Signalizace pomocí Touch LED

Místo výpisů na displeji a zvukových signálů můžeme některou část programu signalizovat pomocí dotykové LED, např. spuštění programu nebo podprogramu, signalizace vzdálenosti od překážky:



Vybrané programy s komentáři

Programy odpovídají (není-li uvedeno jinak) výchozímu robotovi Clawbotovi ze stavebnice VEX IQ.

Pohyby

1 – čeká 1 sekundu po spuštění programu

2 – otočí se o 90° (reálně asi o 30°; bez gyroskopu zde postrádá smysl)*

3 – popojede o jednu otáčku motoru dopředu

4 – rozevře kleště (v portu 11 motor kleští)

5 – zvedne držák kleští (v portu 10 motor věže / zvedáku)

```
1 wait ( 1 , seconds );
2 turnRight ( 90 , degrees , 50 );
3 forward ( 1 , rotations , 50 );
4 moveMotor ( motor11 , 1 , rotations , 50 );
5 moveMotor ( motor10 , 1 , rotations , 50 );
6
```

*ukazujeme, co nás může zaskočit v začátcích, níže je vhodnější příkaz

Jízda do čtverce

1 – opakujeme 4× stejné pohyby…

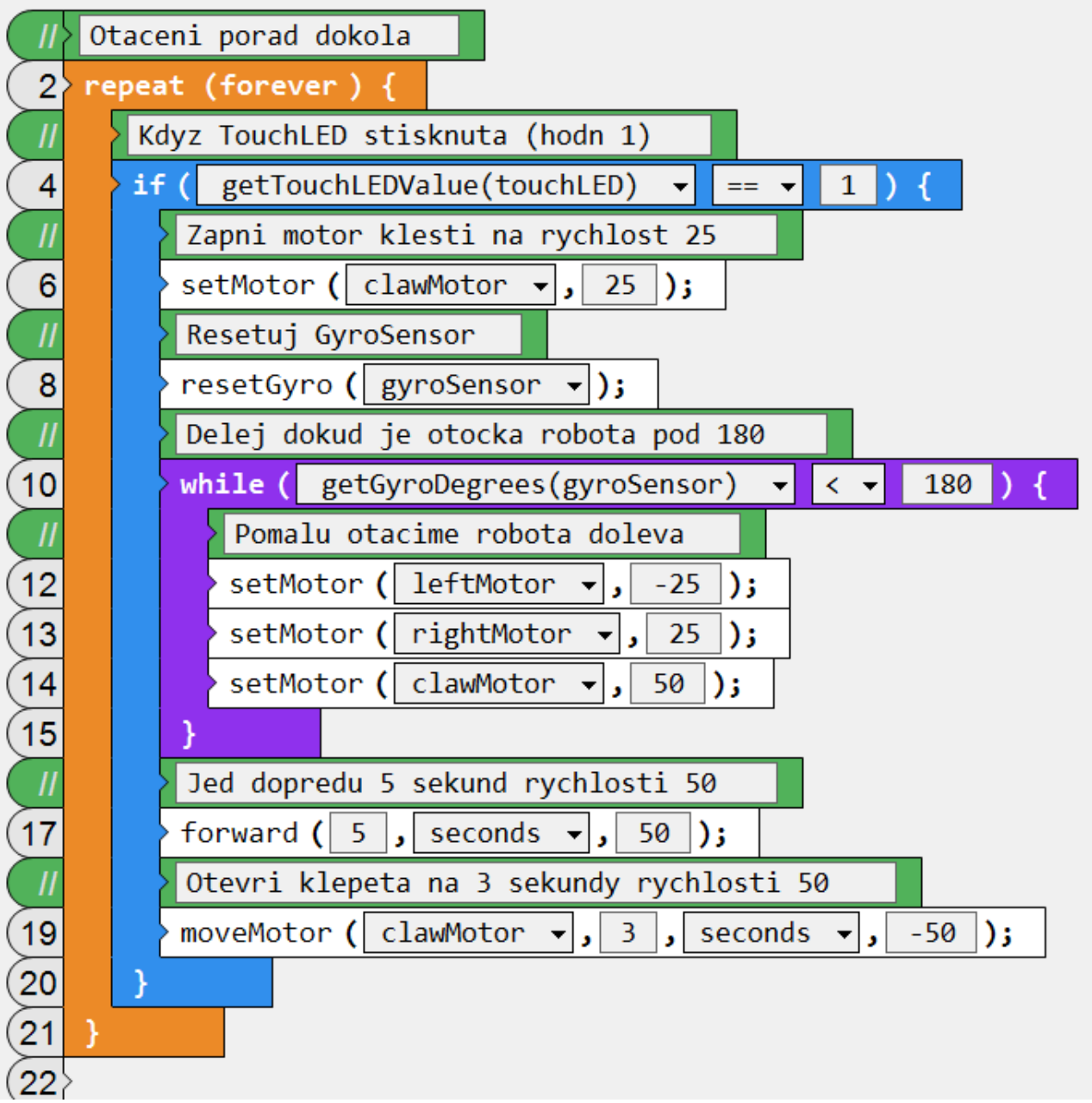
2 – jeď dopředu o jednu otáčku kol

3 – otoč se doprava o 0,7 otáčky kol (u standardní *Drive Base* to odpovídá cca 90°)*

```
1 repeat ( 4 ) {
2     forward ( 1 , rotations , 50 );
3     turnRight ( 0.7 , rotations , 50 );
4 }
5
```

*závisí na rozchodu kol

Příklad programu s vloženými komentáři, ovládání vstupu do programu pomocí dotykové LED

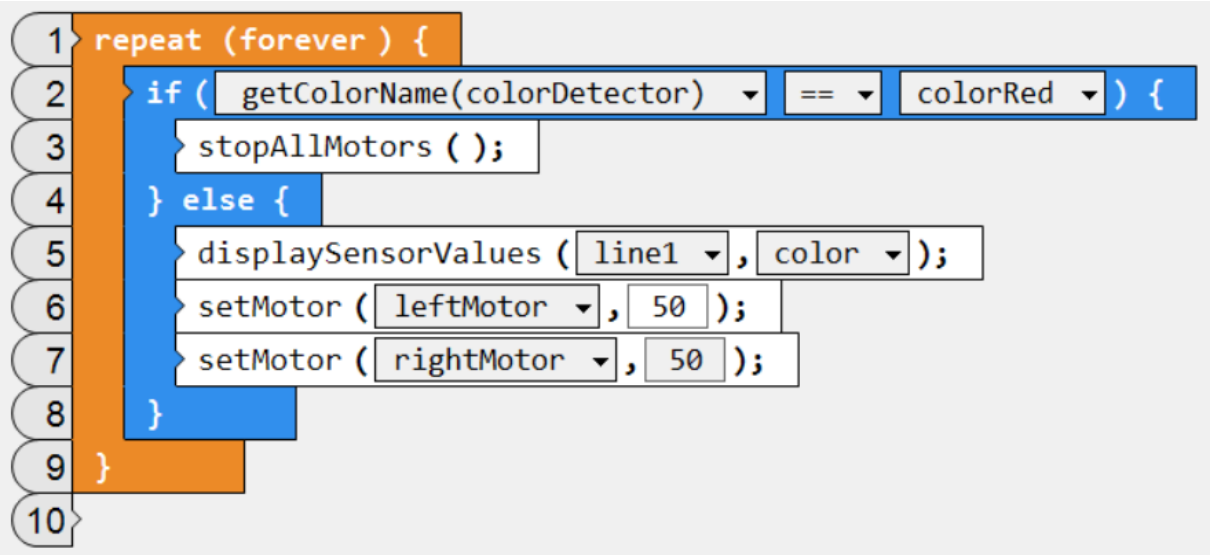


Detekce červené a podle toho zastaví

1 – program běží neustále, a proto je třeba jej přerušit tlačítkem křížku na mozku robota!

2-3 – jestliže barevný detektor vidí červenou, zastaví motory

4-7 – na displeji zobrazuje, jakou vidí barvu (opravdu důležité podle osvětlení) a jede vpřed rychl. 50



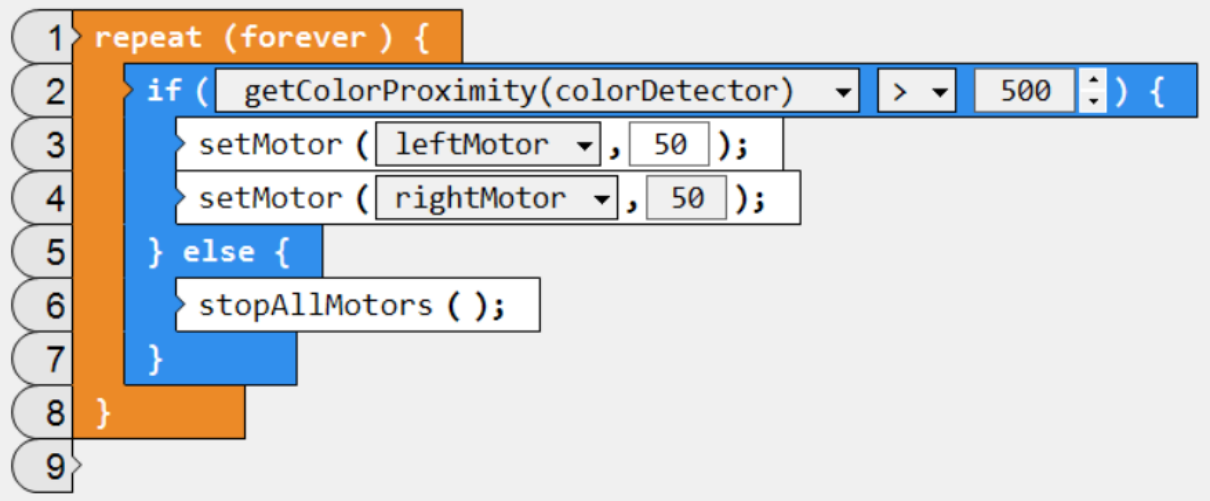
Detekce vzdálenosti barevným senzorem

Program má za úkol pohlídat, že robot nespadne ze stolu. Senzor barev míří dolů a rozsvítí se jeho dvě bílé LED a prostřední infračervená LED měří odražený signál (na škále 0 .. 1023). Když je překážka (např. deska stolu) blízko, množství odraženého infračerveného signálu je velké, naopak od vzdálených překážek, např. od země, se odrazí světla jen málo.

1 – testuj stále dokola

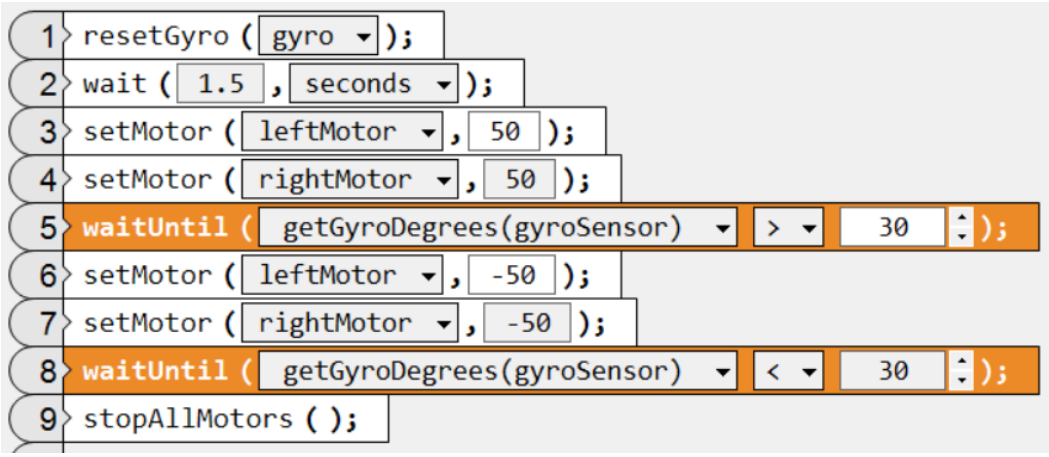
2-4 – když je hodnota z barevného senzoru velká (= překážka je blízko), tak jeď dopředu

5-6 – jinak zastav

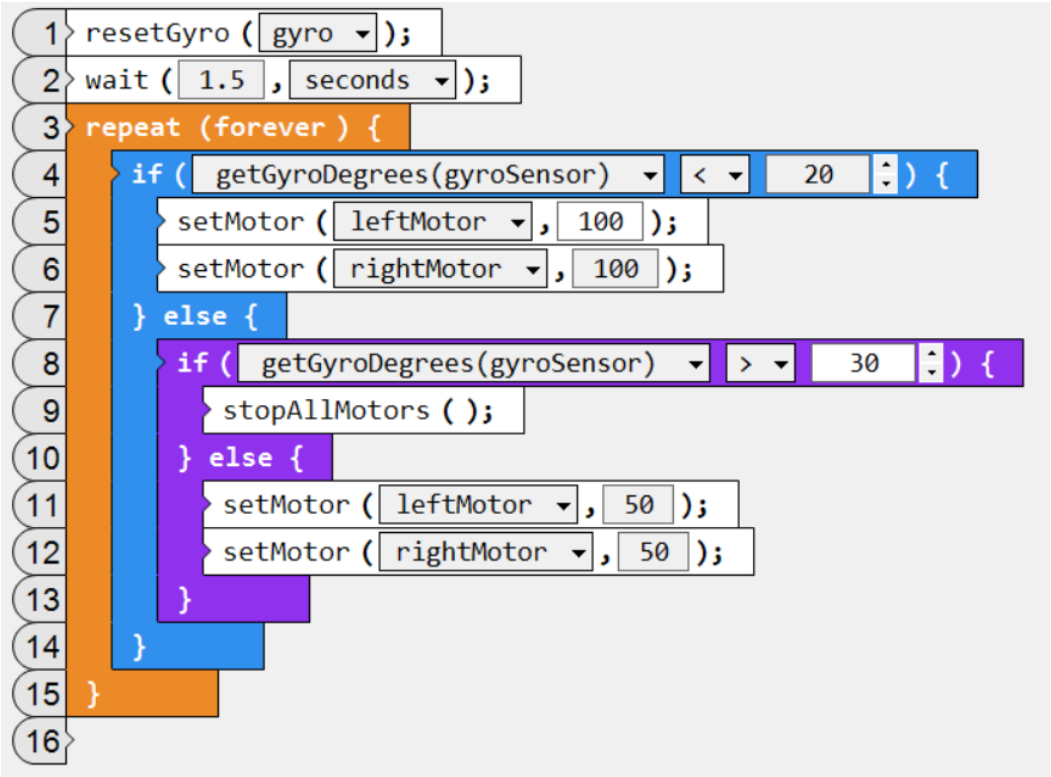


Využití gyroskopu

Při jízdě dopředu detekuj, jestli jsi nenajel na strmý svah. Pokud ano, couvni.

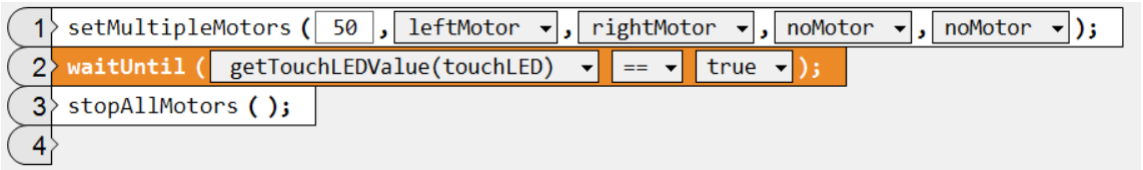


Když je svažitost menší než 20° jeď rychlostí 100, jinak zpomal na 50. Když je sklon nad 30°, zastav.



Dotykový senzor

Při dotyku s překážkou zastav. Hodnota senzoru může být true / false neboli 0 / 1.



Jízda s využitím gyroskopu

1 – vyčkej, dokud se uživatel nedotkne LED

2 – rozsviť dotykovou LED zeleně

3 – jeď dopředu o jednu otáčku kol

4-6 – otáčej se vlevo, dokud gyrosenzor nezaznamená hodnotu 90°
(pro otáčení doprava znaménko mínus, např −90)

7 – zastav otáčení

8-9 – pohni klepety

```
1 waitUntil ( getTouchLEDValue(touchLED) == true );
2 setTouchLEDColor ( touchLED , colorGreen );
3 forward ( 1 , rotations , 50 );
4 repeatUntil ( getGyroDegrees(gyroSensor) == 90 ) {
5     setMotor ( rightMotor , 50 );
6 }
7 stopMotor ( rightMotor );
8 moveMotor ( clawMotor , 0.5 , seconds , 20 );
9
```

Jízda, detekce barvy a podle ní zastavení a rozjetí

1 – program se opakuje stále dokola (testuje barvy, které vidí čislo)

2 – čeká, dokud neuvidí zelenou

3 – jakmile vidí zelenou, rozjede se rychlostí 50

4-5 – jakmile vidí červenou, zastaví

```
1 repeat (forever ) {
2     waitUntil ( getColorName(colorSensor) == colorGreen );
3     setMultipleMotors ( 50 , motor1 , motor6 , noMotor , noMotor );
4     waitUntil ( getColorName(colorSensor) == colorRed );
5     stopMultipleMotors ( motor1 , motor6 , noMotor , noMotor );
6 }
7
```

Jízda, detekce barvy a práce s kleštěmi

1 – robot jede dopředu rychlostí 50 pomocí motoru v portu 1 a portu 6

2 – testuje, zda neuvidí červený objekt

3 – jakmile jej uvidí, zastaví všechny vybrané motory (v našem případě tedy 1 a 6)

4 – rozevře kleště (motor v portu 11 ovládá kleště, záporná hodnota = rozevřít)

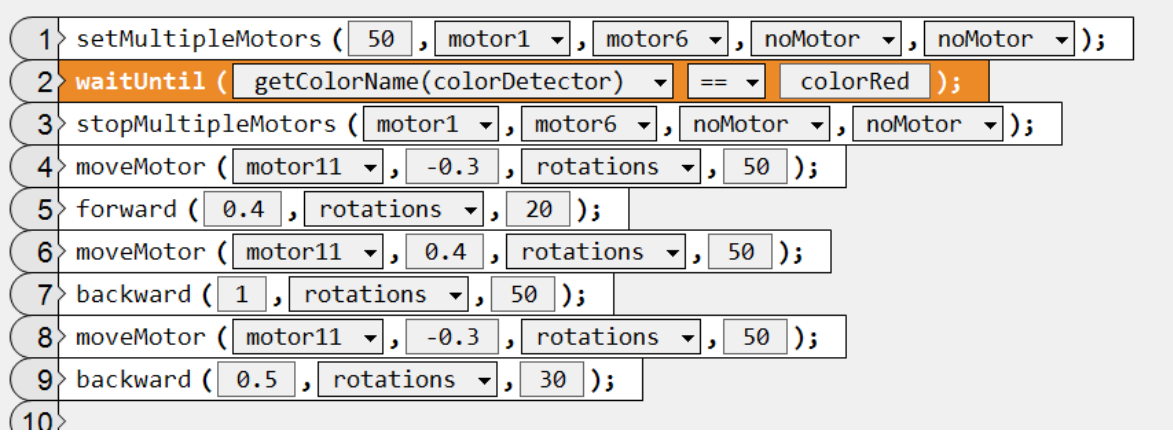
5 – popojede ještě trochu dopředu (rychlost 20, tedy pomaleji)

6 – sevře předmět do kleští (kladná hodnota)

7 – couvne

8 – rozevře kleště

9 – znovu couvne



Autonomní jízda v prostředí s překážkami

1 – opakuje stále dokola (pozor, program nemá konec, nutno vypnout na mozku robota křížkem)

2 – nastaví motory 1 a 6 na couvání (mínus je dozadu) a neustále couvá

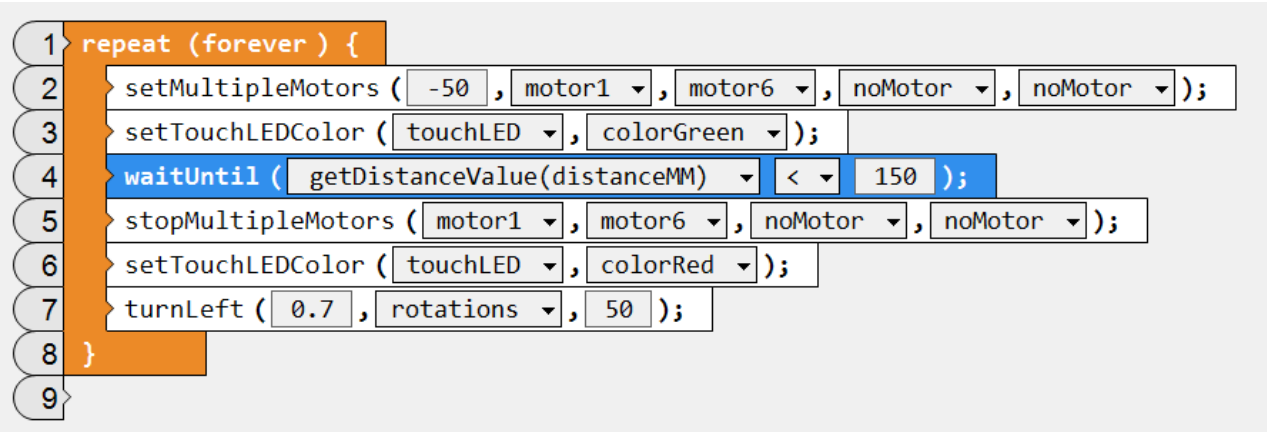
3 – dotyková LED se rozsvítí zeleně

4 – čeká, dokud čidlo vzdálenosti nezaznamená, že méně než 150 mm (15 cm) od něj je překážka

5 – v 15 cm od překážky zastaví motory

6 – nastaví dotykovou LED na červenou

7 – pootočí se doleva a zkouší se opět pohybovat, pokud do 15 cm od něj není překážka



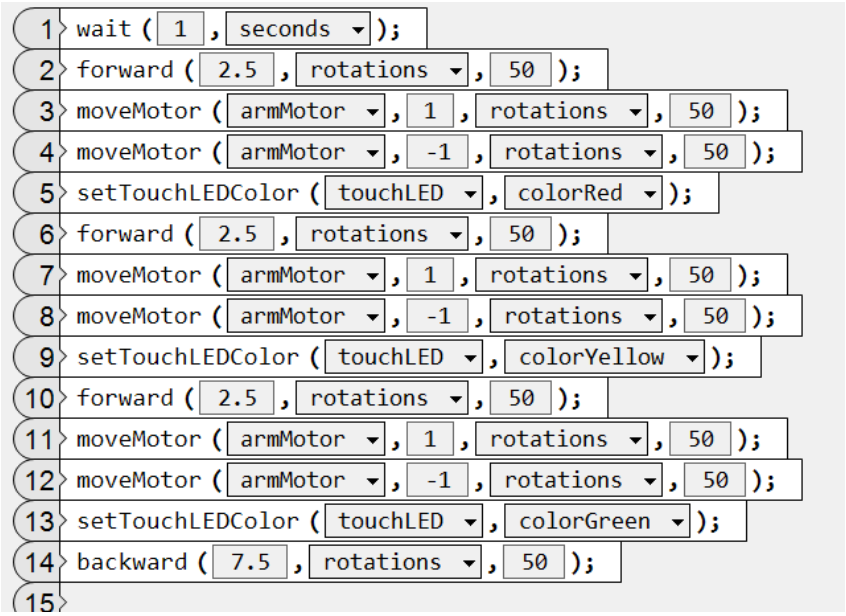
Jízda, zvedání držáku s klepety a signalizace části programu pomocí dotykové LED

1-5 jede dopředu, zvedne držák a položí ho, detekuje tuto část rozsvícením červené na LED

6-9, 10-13 opakuje pouze detekce je barvou žlutou a zelenou

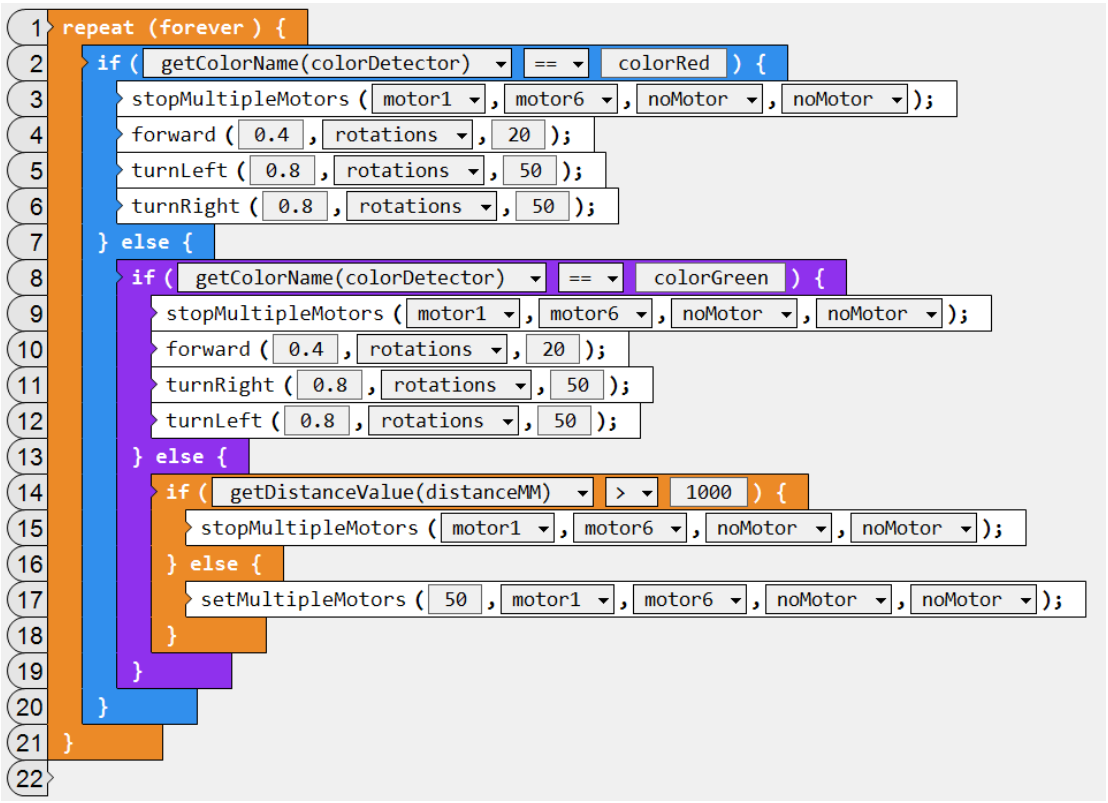
14 vrátí se do výchozího bodu

(Program lze zjednodušit pomocí cyklu, při vynechání barvy LED je to tedy jednoduchá úloha)



Třídička barev

Podle toho, jaký barevný předmět má před sebou, posune jej klepety vpravo nebo vlevo.



Ovladač a Clawbot (v režimu TeleOperated!)

Standardní pohyby motorů v přednastaveném režimu *Driver Control* jsou příliš rychlé. Proto je upravíme pomocí vlastního programu v režimu *TeleOperated*. Na ten je nutné se přepnout v nabídce *Robot – VEX IQ Controller Mode*.

Program nejprve signalizuje, že je v chodu (touchLED svítí zeleně).

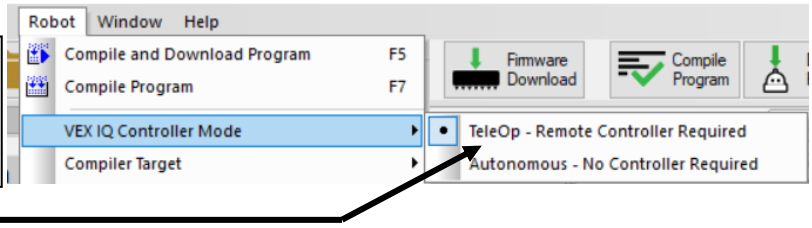
Pohyb paže a klepet ovládáme na předních tlačítkách vlevo a vpravo.

Jízdu dopředu a dozadu zajišťují proměnné *cislo1* a *cislo2*, které jsme si sami zavedli a hodnoty do nich se vyčítají podle vyklopení pák ve směru A a D.
Pokud naklopíme na joysticku ovladač A nebo D jen trošku, hodnota v proměnných cislo 1, 2 bude malá a motory pojedou malou rychlostí. Pokud joysticky naklopíme více, pojedou rychleji.

(Více k umístění tlačítek na ovladači a vysvětlení činnosti je níže a na následující straně).

```
1  setTouchLEDColor ( touchLED , colorGreen );
2  repeat (forever ) {
3      armControl ( armMotor , BtnLUp , BtnLDown , 50 );
4      armControl ( clawMotor , BtnRUp , BtnRDown , 20 );
5      cislo1 = getJoystickValue(ChA) ;
6      cislo2 = getJoystickValue(ChD) ;
7      setMotor ( leftMotor , cislo1 );
8      setMotor ( rightMotor , cislo2 );
9  }
```

POZOR PROGRAM SE NAHRÁVÁ V REŽIMU TELEOP



Přehled ovládacích prvků a tlačítek na dálkovém ovladači

RobotC přijímá data vysílaná dálkovým ovladačem z pákových ovladačů a tlačítek označených písmeny.

Joysticky (pákové ovladače)
A až D nabývají hodnot
-100 až +100

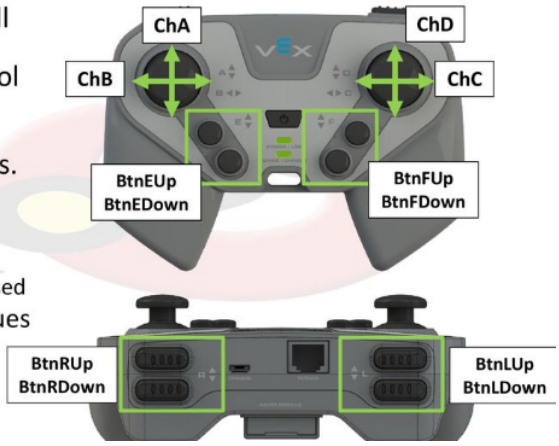
Tlačítka jsou interpretována jako

1 = stisknuto
0 = nestisknuto / puštěno

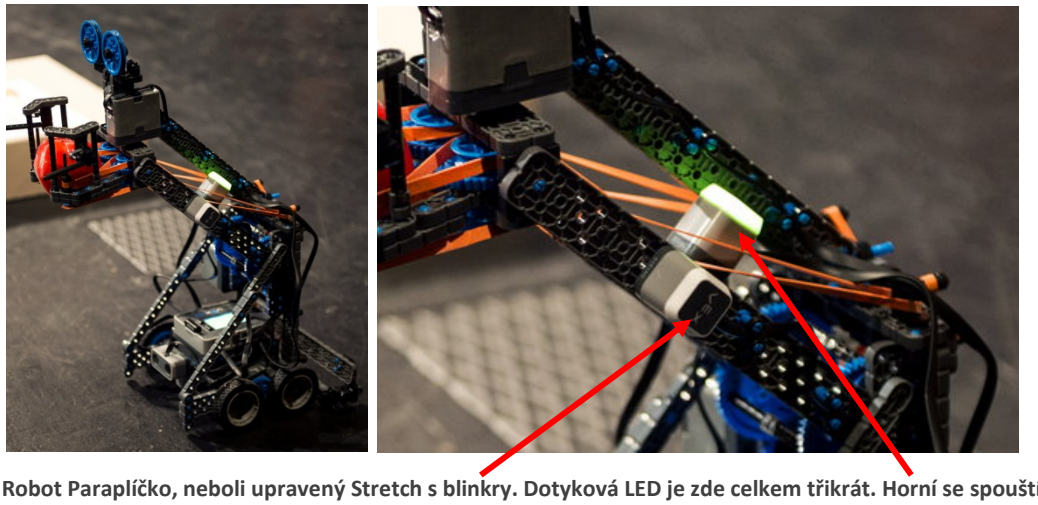
ROBOTC can access all of the data from the VEX IQ Remote Control by referencing the buttons and axes by their described names.

Joystick buttons return values of...
- 1 – Pressed
- 0 – Not Pressed/Released

Joystick Axis return values of...
- -100 to +100 (0 when centered)

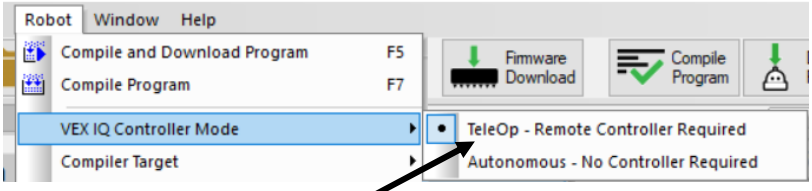


Tento program je vytvořen pro vylepšeného robota Stretch (zde robot „Paraplíčko“ s blinkry :-)



Robot Paraplíčko, neboli upravený Stretch s blinkry. Dotyková LED je zde celkem třikrát. Horní se spouští a signalizují programy a dvě jsou po stránách.

POZOR PROGRAM SE NAHRÁVÁ V REŽIMU TELEOP



Tlačítka vpředu vlevo a vpravo ovládají paži (motor 4) a klepeta (motor 10) robota Stretch.

Jestliže hodnota naklonění kolíku A je větší, než 50, levá dotyková LED se rozsvítí zeleně (u robota na obrázku je to tedy tzv. levý blinkr) a levý motor se rozjede rychlostí 50. Podobně při náklonu dolů (do -50) se motor rozjede dozadu rychlostí -50 (*pro přehlednost zde není programován i motor vpravo*).

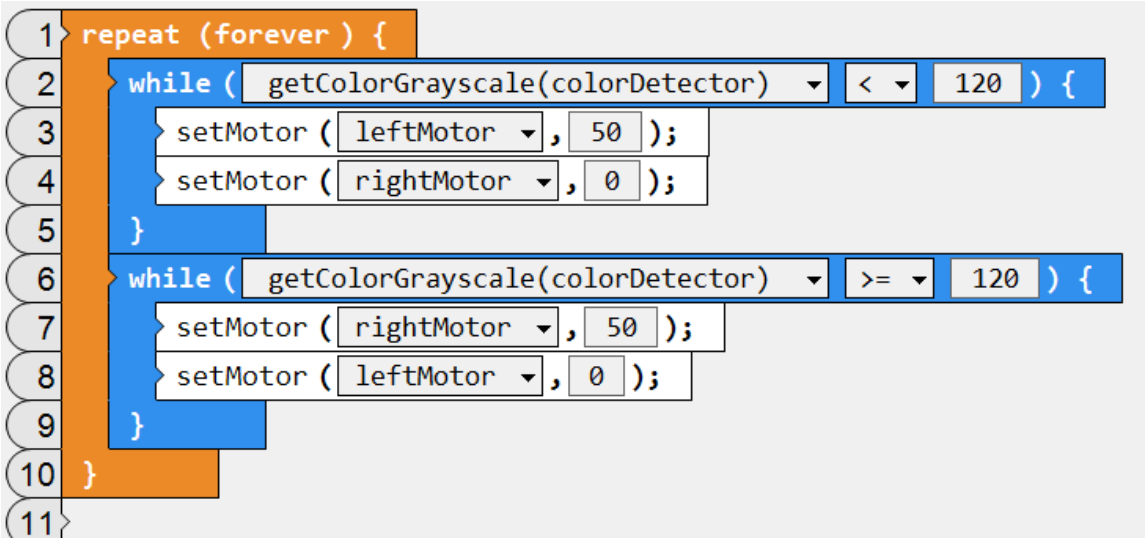
Pokud je kolík v nulové poloze, dotyková LED svítí červeně.

```
repeat (forever ) {
    armControl ( motor4 , BtnLUp , BtnLDown , 75 );
    armControl ( motor10 , BtnRUp , BtnRDown , 25 );
    if ( getJoystickValue(ChA) > 50 ) {
        setTouchLEDColor ( blinkL , colorGreen );
        setMotor ( motor1 , 50 );
    } else {
        if ( getJoystickValue(ChA) < -50 ) {
            setTouchLEDColor ( blinkL , colorGreen );
            setMotor ( motor1 , -50 );
        }
    }
    if ( getJoystickValue(ChA) == 0 ) {
        setTouchLEDColor ( blinkL , colorRed );
        stopMotor ( motor1 );
    }
}
```

Jízda po čáře, základní

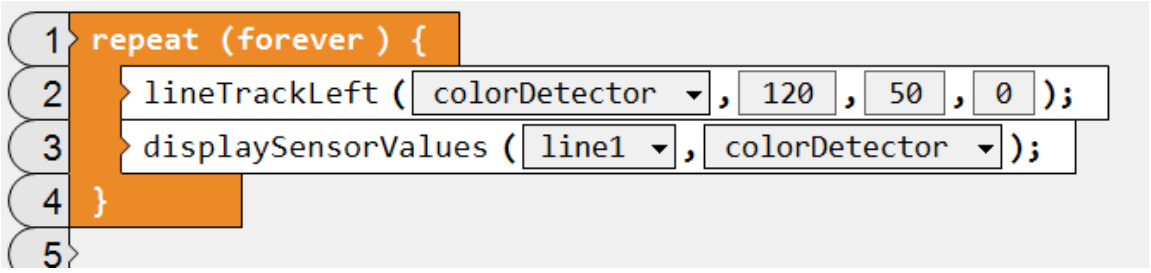
Je třeba předeslat, že zatímco miniroboti typu **ozobot** jsou přímo předurčeni k jízdě po čáře a už se tak „narodili“, tak roboti ze stavebnic typu LEGO nebo VEX to primárně neumí a musí jet podél čáry jakoby vlnivým pohybem, přičemž se drží jednoho z okrajů čáry.

U těchto programů je třeba **zjistit práh detekce černé a bílé** (např. někde mezi 50 a 150) a **nastavit barevný senzor na Greyscale!** (*Motor and Sensor Setup – Devices – Color-greyscale*)



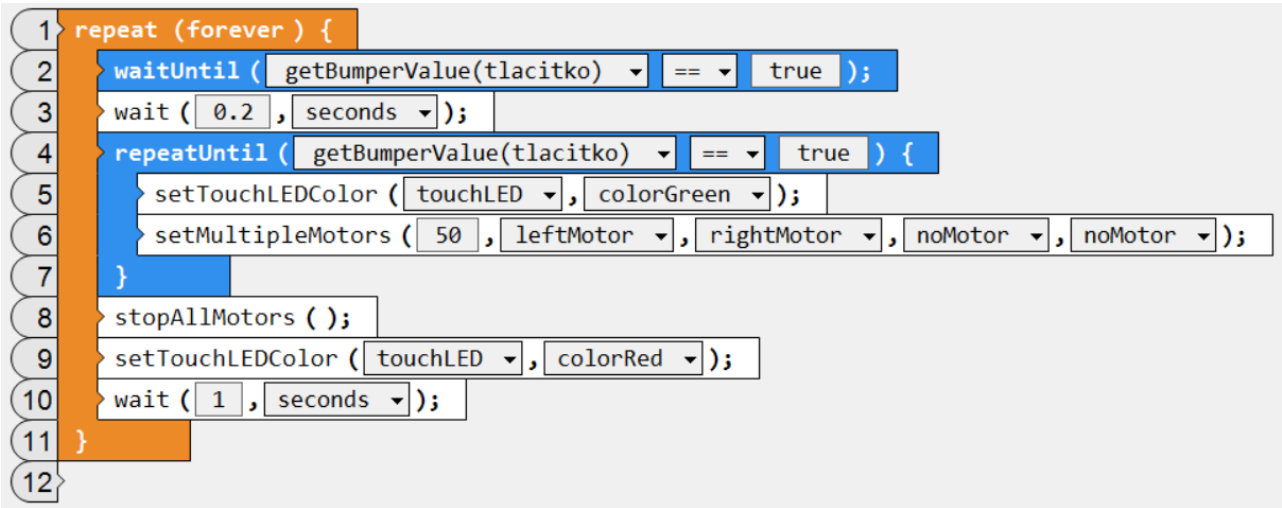
Jízda po čáře s využitím vhodného předpřipraveného příkazu

Pro účely jízdy po čáře je ale přímo připraven příkaz *lineTrackLeft* (nebo *lineTrackRight*).



Opakované spuštění a zastavení programu tlačítkem

Po stisku tlačítka se rozjede a LED svítí zeleně. Další stisk = LED červeně a zastavení. A to pořád dokola.



Obsah

Zdroj inspirace:
Classroom Activities for the Busy Teacher: VEX IQ with ROBOTC Graphical, Damien Kee, 2016
http://www.damienkee.com/books/